SCHOOL OF COMPUTING

UNIVERSITY OF UTAH

2016 GRADUATE COUNCIL PROGRAM REVIEW

2016 School of Computing Graduate Review

1. Program Overview

1.1 Program Mission and Organization

1.1.1 Mission

The School of Computing has three complementary aspects to its mission:

- Research – the development of cutting edge technology – achieved by tackling problems of real-world complexity – with the potential for significant long-term impact on the fields of computer science and multidisciplinary computing;
- Education – providing the State of Utah and the Nation with computer and computational scientists having a core of knowledge that allows them to perform at the highest levels in industry, academics, and government while adapting to rapidly changing technological, social, and professional landscapes, *and* providing students across the University of Utah with access to skills in computing, computer programming, computer science, and data analysis (hereafter referred to as *CS*);
- Service – working with industry, government, educators and the community, at the State and National level, to advance computer science education and research, and to provide these organizations and groups with access to computer science expertise and talent.

1.1.2 Overall Vision

The overall vision of the School is multifold:

- Establish itself as the premier institution in Utah and across the Intermountain West for computing and computer science education and research;
- Maintain and enhance the Schools national and international reputation for innovative, world-class research and exceptional undergraduate and graduate education;
- Become the focal point and at the University and across the State for providing access to high-quality, foundational educational opportunities in computing, computer science, and data science for students in all disciplines and all programs; and
- Provide leadership across the State, throughout the Intermountain West for the region's burgeoning information economy.

1.1.2 Strategic Goals

This overall vision translates into strategic goals for the School:

- Recruit and retain exceptional faculty and staff from around the world;
- Recruit and retain highly qualified undergraduate and graduate students from diverse backgrounds and circumstances and provide them with a world class educational experience;
- Create an environment that fosters innovation, creativity, learning, and strong positive commitments to the School;
- Improve the operational efficiency of the School to better utilize time of faculty and staff, as well as capital and space;
- Improve interactions with the University Administration and other academic leaders/units in order to advocate for and offer CS-related educational opportunities and procure resources necessary to capitalize on the growing opportunities in CS;
- Increase and improve communications with stakeholders, including industry, parents/citizens, legislators, and alumni in order to acquire valuable feedback on the School and its roles, publicize the Schools capabilities, accomplishments, and potential, and garner support for the increased role that the School (and CS, generally) will play at the State and National levels;
- Increase international exposure for the School highlighting its accomplishments and capabilities;
- Diversify the set of CS-related educational offerings into important, related areas, such as digital/interactive media, learning at scale, and data science;
- Establish, with the University Administration, a trajectory for growth/evolution of the School that reflects realistic assumptions about the future role of CS in the State and elsewhere; and
- Procure the resources necessary to achieve this new status and carry out the proposed diversification and increase in activities.

Challenges and Risks for the School

1. <u>Maintaining growth relative to demand:</u> Increasing demand for CS education from both students and employers poses a particular set of challenges for the School. For instance, the median freshman GPA of students admitted to the CS program in 2016 was 3.7. This has important implications for the School and the State, such as lost opportunities (e.g. B

students who cannot become CS majors), the relationships with stakeholders (e.g. legislators, parents), as well as serving as a disincentive for students to come the University and plan to major in CS.

2. Diversification of programs at the University: In response to the State demand for programing skills, other units on campus are offering IT and software related training. For instance, the School of Business now offers an MIS degree and has proposed a degree in data management. The School of Architecture offers programs in design, but is moving toward digital and interface design as specialties. These programs produce different kinds of skills, but have been known to be confused (by outsiders/employers) with the CS degree skill set, and may draw attention and resources from the School of Computing and its mission. The College of Science is exploring new directions/units to address the demand for data science. Alternatively, Math is proposing a MS in Data Science, jointly with the School of Computing.

3. Growth of other programs within the State: Other State schools, such as Utah Valley University (UVU), are investing heavily in CS and outpacing the growth of the School of Computing. Some schools are offering new programs in CS and CE. This trend has the potential to dilute resources and attention at the State level, which could impact growth potential for the School of Computing and its programs.

4. Constraints of resources and space: The School is growing, in faculty, students, and staff. However, growth is constrained by several factors. While the trajectory of computer science is distinctly upward relative to other STEM disciplines (even other engineering disciplines), the allocation of resources for new faculty hiring takes place within a context of limited College and University resources. In these circumstances, successfully making the case for *growth relative to demand* faces significant challenges. Meanwhile available space for offices and student laboratories in the School has grown very little over the last five years, and faculty space needs are currently not consistently met by School resources. The faculty is currently in a mode of "doing more with less" in their use of space.

5. Retention of BS degree seekers: The School estimates that approximately 10% of each class per year fail to continue in the program. About half of this attrition is due to probation-related unenrollments, and the other half is for reasons that are unclear. The School will need to work to improve retention and graduate rates.

6. Faculty recruitment and retention: While recruiting of School faculty has been very successful over the last 10 year, the continued successful recruitment of faculty will likely become more difficult as other programs around the Nation grow in response to overall workforce demands. Likewise, the extremely productive faculty at the School have not gone unnoticed, and with increasing competition faculty are being recruited to go elsewhere. This problem is aggravated by a salary/raise model that does not keep up

with trends in the field and the limited availability of good quality research space (as mentioned above).

7. Ethnic and gender diversity: The School has made progress on the recruitment of women (four new female hires in the last 7 years). The percentage of undergraduate women has improved, but is still below national averages (11% vs 16%). There is a growing awareness at the State level that in the face of increasing demand, the under tapped talent pool of women for computer science represents a significant loss. Unique cultural aspects of Utah (lower percentages of two income families relative to the nation) also aggravate this problem. Meanwhile, Hispanics are significantly underrepresented at the University and especially within the School. The School of Computing must address these issues in order to fully access available talent pools (and meet workforce demands) and offset institutionalized biases associated with systematic underrepresentation.
8. Responding to the changing field: The field of computer science/computing is changing rapidly and, sometimes, unpredictably. The subject matter, the expectations of our students and their employers, and the methods of instruction are all moving targets, and the School will need to demonstrate a larger degree of agility in how it defines its curricula and how it uses modern methods and technologies for instruction.

The above challenges are not all unique to Utah, and the ability of the School to meet these challenges will define its identity and level of success in the coming decades.

1.1.3 Overview of the School and Programs

The School of Computing at the University of Utah began as a Division within the College of Engineering in 1965, and become a Department in 1973. In 2000, the Department of Computer Science made a transition to become the School of Computing. The goal of the transition to a School was motivated by the dramatic growth in the field of CS circa 2000 and was meant to broaden the School's mission to better serve a diverse set of intellectual and societal issues, to foster collaboration and interdisciplinary education and research, and to provide a possible on-ramp for the School to become a separate College. In 2004 the School offered graduate students the additional option of a Computing degree (in addition to CS), which allows for students to follow *tracks*, which are curriculum options that are managed in a flexible manner by small faculty committees and *Track Directors*.

The School offers several degrees and options, some jointly with other programs:

- BS in Computer Science: Students are admitted into the program (at our discretion) at the end of freshman year. Total enrollment is approximately 500. We currently graduate approximately 100 students per year, and have been increasing our admissions by about 10% per year, with the latest incoming class, of approximately 200 students.
- BS in CS with EAE emphasis: Students in the BS in CS program take a set of electives

and obtain a certificate in "Entertainment Art and Engineering". About 20 students per year take this option. EAE is now also an independent program (offering a master's in EAE), which collaborates with the School on this undergraduate emphasis.

- BS/MS in Computer Science: This is a five year degree option for which highly qualified students may apply. Approximately 20 students per year enter into this program.
- BS in Computer Engineering: This joint program with the ECE Department graduates approximately 20 students per year.
- MS in Computer Science/Computing: Students are directly admitted into the MS program and may choose among course only, project, and thesis options. MS degrees are awarded per year range from approximately 45 to 85.
- PhD in Computer Science/Computing: Students choose between the conventional CS curriculum or among the tracks in the Computing degree (about 50% currently). Approximately 20 students per year are awarded the PhD in either CS or Computing.

	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015	2015 – 2016
MS Enrollment	101	129	129	129	152	168	131
MS Degrees Awarded	43	51	51	57	51	88	75
PhD Enrollment	122	126	121	124	130	120	137
PhD Degrees Awards	16	12	15	19	19	21	16
Declared Pre-CS Majors	316	387	446	486	605	772	814
Declared CS Majors	352	393	406	443	450	450	551
CS BS Degrees Awarded	71	76	77	87	92	101	98
Declared CE Majors		68	71	76	92	99	
CE BS Degrees Awarded		23	12	18	20	16	15

Table 1-1 – Enrollment and Degrees Awarded

Above master's degree numbers do not include Computational Engineering and Science (CES) students, these are included in Table 1-6.
Above Bachelor CS degree numbers do not include Computer Engineering (CE) students, these are included in Table 1-6.

The School maintains a strong, active research program with highly productive faculty, students, and staff. Several quantitative measures such as paper output and research expenditures confirm this. For instance, research expenditures in the School have doubled in the last 7 years to approximately $20 million per year, which averages to over $500,000 per faculty member, per year. Other measures of research productivity are consistent with this. The faculty/students published approximately 234, 198, and 214 papers over the years 2013, 2014, and 2015, respectively. The faculty are/have been on 25 editorial boards and 4 served as editors-in-chief of journals in the past three years. In the last three years, faculty members have served as conference committee chairs 107 times. The School's faculty have served on numerous national boards, including the CCC, the CRA, and PCAST, as well as serving on many national and international task forces and review boards, including several under the auspices of the NSF, the

NIH, DoE, ACM, and IEEE. Since the past review, several faculty members have been honored as IEEE Fellows, ACM Distinguished Scientists, and AIMBE fellows.

While the expertise of the faculty virtually spans the field of computer science, the faculty are loosely organized into the following areas of research activity:

- Algorithms and Theory
- Computer Architecture
- Data Science and Information Management
- Programming Languages and Formal Methods
- Graphics and Animation
- Image Analysis
- Human Computer Interaction (HCI)
- Machine Learning and Natural Language Processing
- Networking, Embedded Systems, and Operating Systems
- Robotics
- Scientific Computing
- Security & Privacy
- Visualization

In addition to this informal organization, there are several more formal research entities that bring international visibility to the School and the University:

- Scientific Computing and Imaging Institute – http:/www.sci.utah.edu
- The Utah Robotics Center – http:/robotics.coe.utah.edu
- The Flux Group – https://www.flux.utah.edu
- Center for Parallel Computing – http://www.parallel.utah.edu

Tenure line faculty salaries and most full time career-line teaching faculty and support staff are funded by the State of Utah. The School receives additional income, mostly from productivity based incentives. This includes 15% of the research overhead generated by the School (to support one-time hiring costs, facility improvements/renovations, and research-related infrastructure) , differential tuition for engineering classes, and additional teaching related incentive funds that are based on a combination of number of declared majors and student credit hours. A more detailed accounting is given in Section 6.1. Overall, from an immediate cash-flow perspective, the School is financially sound. This raises questions about how to best utilize this "soft" money, which cannot be readily applied to supporting salaries for permanent hires.

Regarding faculty salaries, the compensation compared to CS departments in the CRA-Taubee survey show that the School is competitive with departments in other research universities (Table 1-2). (The cost of living in Salt Lake City is about at the national average, and therefore at a gross level these numbers should be relevant).

Rank	Taulbee CS Depts	Taulbee CS Public	Taulbee CS Private	SoC Tenure Line
Assistant	$96,055	$95,199	$105,060	$95,509
Associate	$109,633	$108,929	$117,563	$118,451
Professor	$149,036	$145,267	$170,963	$150,067

Table 1-2: The median salaries by rank in the School of Computing compare favorably with median salaries at research universities nation wide.

Past strategies for hiring, growth, and allocation of resources have been to build on the School's strengths in niche research areas. The School has historical, internationally recognized strengths in graphics, and more recently visualization (SCI Institute), systems (Flux group), and several specific, applied areas such as scientific computing, medical image analysis, and robotics. Very recently the strategy has been to build a more diverse set of strengths that take advantage of the School's culture of collaborative and applied research. This is motivated by several factors. First, is the general growth of the School and the availability of resources to grow the faculty and students, allowing for substantial resource allocations (e.g. critical mass) in more diverse specialties. A second motivation is the demand from students and employers for training/expertise in growing or emerging specialties, such as data science, HCI, and security. A third motivation is the evolution of the field of CS itself, where many of the opportunities for greatest impact (and national/international recognition) lie in new areas.

In light of this, the School's strategy includes expansion into areas that have not been traditional strengths but synergize with existing expertise and interests of the faculty. Thus, recent hires in HCI build on expertise in data visualization, security hires build on strengths in systems/networking and programming languages, computer vision builds on medical image analysis and robotics, etc. Of course, this strategy must respect our underlying principles in hiring: hiring faculty of excellent quality whose dispositions will foster collaboration and enhance the productivity of their colleagues.

1.2 Program Planning

1.2.1 Educational programs

Programs are planned via the Director's office, through the Directors of Undergraduate Studies (Jim de St. Germain) and Graduate Studies (Feifei Li). There is an Undergraduate Curriculum Committee, which is appointed by the Director and a Graduate Curriculum Committee, which consists of the computing/CS Track Directors and the Director of Graduate Studies. The faculty participate in changes to curricula and programs by discussions at faculty meetings (and email) and by votes that formally communicate their views to the Directors. Changes and new programs are formally reviewed by the College Curriculum Committee, and subsequent reviews at the

University level take place for new programs.

Here we give some general trends in the programs and some plans underway for these programs.

1.2.1.a Evolution of the graduate computing degrees and tracks

In 2004, the Utah Board of Regents approved a new graduate Computing degree program in the School of Computing. Under this program tracks are created locally and require only School of Computing faculty to agree to run the track, and the approval of the Director of the School. The intent is to have tracks in areas that complement the Computer Science graduate degree. Tracks may be multidisciplinary or interdisciplinary, consist of track faculty from one, two or more departments and are administered by these faculty. These tracks are a direct and anticipated result of broadening the scope of research in the School. Currently there are six track areas, with a seventh in the planning stages. The specific requirements of each track may be found in Section 4B.1. The current tracks for the Computing Degree are:

- Computer Engineering – MS and PhD; Erik Brunvand is the track director; joint
- track with the Department of Electrical and Computer Engineering.
- Data Management and Analysis – MS and PhD; Jeff Phillips is the track director. (See also the certificate program in Big Data Analytics)
- Graphics and Visualization – MS and PhD; Chuck Hansen is the track director.
- Image Analysis – MS and PhD; Tom Fletcher is the track director.
- Networked Systems -– MS and PhD; Sneha Kumar Kasera is the track director.
- Robotics – MS and PhD; John Hollerbach is the track director, joint track with Department of Mechanical Engineering.
- Scientific Computing – MS and PhD; Hari Sundar is the track director.

The Computing Tracks have been successful in allowing our graduate students to specialize in terms of courses and other requirements. At present approximately 40% of our MS students and 39% of our PhD students are enrolled in a Computing track rather than in the Computer Science degree track.

While the Computing Degree concept, with tracks, has been successful at attracting students and allowing them to complete the program while achieving very high levels of research in their areas of expertise, the naming "Computing" combined with the offering of CS graduate degrees has created some unnecessary complication, and there is an active discussion to retire the MS and PhD degrees in *computing*, and offer very similar tracks within the *computer science* degree options.

1.2.1.b Leveling of the MS Program

During the years 2009-2015, the MS program grew substantially, almost doubling in size. This growth served several purposes.

- It helped meet the growing demand for software engineers in Utah and around the Nation.
- It created student credit hours and majors, which generate income for the School.
- It helped buoy the enrollments in graduate classes, helping to justify the teaching of the wide range of classes needed to support the various tracking in computing.
- It served as a recruiting tool for the PhD program.

In 2014, a decision was made to level the number of admissions to the MS program(s). This was done because it was observed that many graduate classes are of an appropriate size (some are too big) and, with increasing applications, a leveling of admissions may help to improve the quality of MS students (who share many classes with PhD students). This strategy dovetails with plans to offer alternative master's-level degrees, as described below.

1.2.1.c BS/MS Program

In fall 2005 the School instituted a combined BS/MS program to attract high-achieving undergraduate students into the graduate program. This program allows an undergraduate to begin to take courses for their MS degree while still an undergraduate. Because many undergraduates have more credit hours than strictly required for the BS degree, this overlap allows most BS/MS students to finish both their BS and MS degrees in five years (according to plan). Students are considered undergraduates until they have two semesters remaining to finish, at which time they are considered graduate students. At the completion of their MS, students receive both BS and MS degrees simultaneously. This program has been very successful, and attracts very high caliber undergraduates. Currently there are 32 students in our BS/MS program at various stages of completion.

1.2.1.d Entertainment Arts and Engineering Program

In 2007 the School initiated a certificate program that offers an *Emphasis* in Entertainment Arts and Engineering within the Computer Science BS degree program. The EAE Emphasis was developed jointly with the Division of Film Studies (College of Fine Arts), which also offers the emphasis. The purpose of this program is to provide an undergraduate, interdisciplinary academic path for those students that wish to have careers in the digital entertainment industry (video games, digital animation, computer-generated special effects, etc.). Several specialized EAE offering count as electives in the CS program, and those classes are popular (enrollments of about 40-50), and approximately 20 CS students per year receive the Emphasis in EAE (with their BS degrees).

Since the last review the EAE program, headed by CS Professor Robert Kessler has broken off from the School to become a separate instructional unit. The EAE program offers their own master’s degree (terminal, professional) in EAE and they have recently proposed a BS in Games

(BSG). They hire their own instructional faculty. Recently there was a joint hire of a highly qualified “games+CS” Professor, Michael Young from NC State, and there are plans for more joint hires as opportunities arise. There are ongoing discussions about the reporting structure of EAE (which currently reports to two Deans and the VPs office) relative to Engineering and the School of Computing.

1.2.1.e Computer Engineering Program

The Computer Engineering program is a separate degree program offering a BS in Computer Engineering. This program is jointly administered by a faculty committee made up of faculty from the School of Computing and the Department of Electrical and Computer Engineering. The directorship of this program alternates between the two departments every three years. The current Director (as of July 2015) is Dr. Erik Brunvand from the School of Computing.

Computer Engineering, as defined by this program, includes the design, implementation, and programming of digital computers and computer-controlled electronic systems. Computer Engineering is a software and hardware-oriented degree whose requirements include courses offered by the School of Computing and/or the Department of Electrical and Computer Engineering. The Computer Engineering program is accredited by ABET, and has undergone an ABET review in 2015. They currently have their own curriculum planning (committee as above) and admissions criteria, and approximately 20 students per year graduate with a BS in CE.

1.2.1.f New programs and directions

To address some of the challenges in Sections 5A.2 and 5B.2, the School is initiating several new programs and course directions.

The first new program is the proposed Master of Software Engineering (MSE). The demand for software engineers is considered a bottleneck for many of Utah’s technology and nontechnology companies, and there is a growing gap in demand between student who have computer skills and those who do not. Meanwhile there is a growing consensus that there is a moderately large, untapped workforce of students who have the raw talent to write software, but who have not had sufficient exposure. To address this need the School is developing a Master of Software Engineering (MSE) degree program that will educate students with Bachelor’s Degrees in various non-technical backgrounds in computer and software fundamentals, so that they can be technically proficient in software engineering, and participate in this growing workforce. Thus, by reaching out to non-CS majors, the MSE program will recruit an untapped demographic to help meet local and national demand for software talent. Our MSE curriculum stresses significant hands-on teaching and an immersive learning environment. It uses project-oriented approaches to equip students with tools and perspectives for problem solving while honing their critical thinking skills that transcend specific software languages or applications. The program will be mostly taught by a dedicated team of instructional faculty, who will be supported from the revenue the program generates. Tenured faculty manage the MSE program, and thereby establish the pedagogical vision and standards, and swiftly accommodate trends in computer science and software engineering. The duration of the MSE program is 18 months. The MSE program will admit non-CS majors who can demonstrate problem solving skills and the ability to reason mathematically and logically. The proposed program is modeled loosely on similar programs at the University of Pennsylvania and the University of Chicago (and plans, as far as we know, at other schools, such as University of Illinois Urbana-Champaign). The proposed program has been approved by the University's Board of Trustees and is awaiting final approval by the State Reagents.

Another new direction for the School is the development of alternative on-ramp courses for computer programming. This past year the Director formed an ad-hoc committee to examine questions relating to undergraduate students' first exposure to CS on campus and to relate this to what has been done at other universities. This committee has made several important observations. One observation is that our current CS1&2 offerings (CS1410 and CS2420) are considered very difficult classes, whose grades become essential to admission into the program (at the end of freshman year – see Undergraduate Section 3A.1.3). This creates a competitive atmosphere, in which not all potentially qualified students feel comfortable. Second, these classes have some very specific learning objectives (e.g. data structures and software design paradigms) that are considered essential for acceptable progress through the subsequent parts of the CS curriculum. Third, because these classes are used to evaluate the fitness of freshman for the CS program, the grading policies in these classes are enforced in a way that ensures that there is an adequate separation between mediocre and very promising students. Therefore, these introductory classes are not particularly well suited to students who want to learn computer programming but are not (yet) interested in a CS degree. While other universities are seeing booming enrollments from non-majors (or people not intending to major), we have not seen this at the University of Utah (and have not actively promoted it).

To address this issue of increasing interest in CS-skills among nonmajors, the School is planning a set of alternative, introductory programming classes for non-majors. These classes would be designed around projects and hands-on experience, and would not face the specific, challenging learning objectives of our current on-ramp. The projects would focus on domain-relevant applications such as digital media, life sciences, and data science. We call these classes, informally, CP1&2. The goal would be to try to entice departments from all over campus to recommend these classes to their majors. The plan is also to take a subset of our junior/senior level electives that do not have a strict requirement of conventional/broad CS content (such as visualization or data science) and make them available to students who have passed CP1&2. We anticipate that students who do particularly well in CP1&2 could be offered a "back door" into the CS program. These classes will become required in the new BS in Games proposed by EAE. We believe this strategy will address several of the challenges above, helping to meet the growing demand across campus for computer skills as well as making parts of our program more accessible to a greater diversity of students. We have a tentative plan to begin offering these classes in the Fall of 2017.

1.2.2 Outreach

The School is engaged in a variety of strategies to better connect with potential students, alumni, academic communities, industry, and the Utah community at large.

1.2.2.a K-12 student interactions

The School is engaged in a large number of interactions with K-12 students in order to promote CS in Utah and to publicize the presence and activities of the School.

GREAT Summer Youth Camps: The School offers technology-oriented camps for students in 4th-12th grades. These GREAT (Graphics and Robotics Exploration with Amazing Technology) camps provide intensive and fun instruction in computer programming, robotics, graphics and games for 700 students a summer in weeklong camps. The GREAT camps provide targeted scholarships to recruit these students. The camps offer such scholarships to roughly 50 students a year. As a large outreach program, the camps have also been able to offer customized camps for different groups. Recently, the GREAT camps have:

- a summer bridge program for African refugee students preparing to attend the University;
- a high-school mentoring program for refugee students;
- two camps for military kids in the western US; and
- a camp partnering with researchers in Biology exploring ecological simulation.

Each summer, the GREAT camps hire 20 or more instructors. In addition, the GREAT camps have a high school intern program with around a dozen interns each summer. The summer camp teaching role has been a valuable means of mentoring high quality students and helping them feel a part of the School. An important aspect of the GREAT camps is that the large number of topics for different age groups allows students to return to the University of Utah campus for many summers and to build up skills in a variety of computer science topics and to eventually take on leadership roles as interns. The GREAT camp attendees are well prepared for future STEM courses and majors at the University and elsewhere.

Engineering Days: The School provides CS activities for the College of Engineering's yearly Engineer Day, where over 500 high school students try out different mini-courses. Last year, the School ran three sessions where students engaged in a quick, hands-on activity programming an interactive graphics application.

Hi-GEAR: The College of Engineering hosts a week-long camp for 25-30 high school young women. Each day is devoted to different engineering topics. The School has provided day-long and half-day long sessions on different topics ranging from image processing to robotics.

School Field-Trips: The School hosts school field trips during the year. Last year the School provided tours, demonstrations, and hands-on activities covering security, visualization, and robotics.

Besides these activities that are a number of faculty who independently participate in outreach activities to the community as part of their professional service. These activities include, for instance:

- judging at local science competition, such as the Salt Lake Valley Science & Engineering Fair;
- running/judging the Utah Regional First Robotics Competition;
- workshops for Salt Lake City K-12 educators in art and technology; and
- mentoring of high-school students interested in CS-related topics.

1.2.2.b Industrial interactions

In 2008, to improve our relationship to local technology companies and to better understand how our programs fit with their needs and expectations, we reformed a (then) dormant *Industrial Advisory Board* (IAB). Dr. Matthew Might headed this effort and held the first meeting of the new IAB in May 2009. In May of 2014 Dr. Miriah Meyer took over this role.

Currently, each annual IAB meeting is designed around a theme — in 2015 the meeting focused on growth of CS and the resulting broadening of skills necessary in our local industry; this year we will focus on ideas for building a closer relationship between the School research and local companies through lab-lette style endeavors. These meetings consist of a mixture of brainstorming and discussion around the theme, as well as yearly updates from the School Director and other faculty about changes within the School. The agenda and brainstorming materials from the 2015 meeting are included in Appendix A.

The current IAB consists of 15-20 members. These members are predominately executives from local technology companies, but also include several alumni that work in other western states. The current members are: John LaLonde (CEO, Abstrax), Rob Nelson (VP of Technology, Disney Interactive), Galen Murdock (CEO, Veracity), Steve Townsend (VP of Engineering, Instructure), Jon Morrey (Technical Research Manager, FamilySearch), Robert Palmer (senior engineer, Tableau), Michelle Kolbe (consultant, Red Pill Analytics), Mark Sharrock (VP, GoldmanSachs), Karl Sun (CEO, LucidChart), John Hatfield (Distinguished Engineer, TaskEasy), Jeff Pinkston (Director of Software Development, Microsoft), and Chachi Kruel (CTO, Experticity).

Since 2014, Director Whitaker has been active in the Utah Technology Council (UTC), and has given presentations and talks at various public policy forums and closed meetings associated with this group. The UTC is an advocacy group consisting of over 5000 member companies, with a heavy representation of software/IT companies. The UTC is the primary lobbying organization responsible for the College's *Engineering Initiative*, which has resulted in increased budgets for faculty hiring and sustained growth (student production) across all departments in the College of Engineering. Plans are underway to sponsor an additional appropriation for the Engineering Initiate during this year's legislative session (winter 2016/2017).

1.3 Previous Review and Actions

Here we recapitulate recommendations from the 2009 review and describe actions and progress.

Recommendation 1: Strategic plan. The School should produce a strategic plan that, after examining trends in computing and unique strengths of the School, provides ways for improving and growing the School. In particular, the plan should delineate a strategy for a steady but controlled growth in faculty and graduate students, especially PhDs.

The strategic plan is an ongoing discussion between the Director's office and the faculty in the School. These discussions have taken place at our annual retreats, which are generally set aside for this purpose. These discussions also take place at regular faculty meetings as decisions become timely and the faculty are giving input on specific issues, such as hiring, growth, etc. The current set of strategic goals are described in Section 1.1.2. With respect to growth, the School has acted, in the last five years according to a particular, agreed-upon plan, which is steady, controlled growth in the undergraduate program (approximately 10% per year), building a stronger MS program to strengthen the graduate class offerings (which is now in place, at 50-70 new MS students per year), and growing the PhD program in proportion to faculty/research needs, while maintaining quality. This is also in place, and for the last two year, incoming PhD classes were 25 students (and an additional 5 or so admitted off schedule).

Recommendation 2: Faculty. The School should make sure that junior faculty are given clear guidance on what standards they must meet in order to be successful in their evaluations. The College and School should work together to create a clear policy regarding the treatment of clinical (lecturing) faculty members and a more stable, career-oriented track for them.

Since this review, the Dean has launched a new faculty-mentoring program at the College level and, due to its success, it has been institutionalized. Junior faculty all have assigned mentors. All junior faculty receive formal, annual, written feedback on their progress from a full faculty committee. This includes a one-on-one meeting with the Director to discuss this feedback. Prior to the tenure review, a junior faculty member can expect to have no less than five, full, written evaluations, and dozens of meetings with their mentor and the Director focusing on their performance. More recently, the junior faculty have initiated a Slack channel to discuss issues that are pertinent to their jobs, including performance. All of this feedback exists in the context of the School's written policies for tenure and promotion, which state the criteria that will be evaluated gives specific examples of the data that will be used in making this evaluation.

Regarding career-line and auxiliary faculty, the College has adopted formal policies and guidelines for *nonregular* (that term is no longer in use) faculty hiring, review and promotion. The School has implemented its own working version of that policy, which is given in Appendix B. Some aspects of these policies have raised confusion (e.g. "scholarship" requirements for teaching faculty, and teaching requirements for research faculty), and the Director/Associate Director of the School have had meetings with these faculty to resolve this confusion and to discuss how the policies will be implemented in the face of these apparent contradictions. The

Dean's office has been made aware of these contradictions in the College policy (which the School has been told are not pre-emptible). Despite these minor confusions, the current written policy and its implementation have been clearly described to the appropriate faculty, and since the last review, there have been several successful promotions of career-line faculty.

Recommendation 3: Students: The School should encourage first-year students to engage in research projects as RAs, for example through appropriate incentives to faculty, and should shift TA responsibilities to more experienced students. The School should minimize the number of courses with mixed undergraduate and graduate content, and should increase the number of intellectually challenging and technically deep graduate-only courses. The School should institute programs beyond the coursework to improve student excitement for computer science.

In 2014, after a great deal of planning, discussion, and analysis, the School implemented a new *fellowship program* for (virtually all) incoming PhD students. The program provides a full paid graduate stipend (and tuition waiver and health benefits) for the first year of PhD study. During this time students are required to participate in 4 hours of *research rotation*, which includes independent study and research seminars. Students are required to enroll in 4 hours of *teaching mentorship* (TM) (unpaid, simultaneous with RA) by their third year in the program.

This program was designed to simultaneously address several issues. First, it serves as a recruiting tool. Based on interviews with perspective students we found that fellowships are valued more highly than promises of TAs. Second, it exposes students to a wider range of research areas in their first year and provides opportunities for lighter weight (i.e. nonpaid) exploration of potential research. Finally, it also provides a pool of experienced students to help with classes. Thus, a typical class may have assigned a TA for grading (and undergrad or MS student) and a TM to help with office hours, assignments, lecturing, etc.

We consider the plan to be an experiment, but the feedback so far is positive. PhD enrollments have risen and students and faculty have been generally support of this plan and its outcomes so far. There have been some objections to the TM requirement by students and faculty. Early data suggests that the program is financially solvent (the TMs strategy almost offsets the cost of the fellowships).

Regarding courses with mixed offerings (graduate and undergraduate), the School has been systematically phasing these out, there are relatively few classes that have this intentionally mixed student body. Also, now that the tracks are more established and the overall graduate program is bigger, there are more offerings of truly advanced topics, such as Advanced Image Processing, Topological Data Analysis, and Advanced Algorithms.

The issue of "programs beyond coursework" remains open and ongoing. At the undergraduate level, there are a significant number of summer internship possibilities and REUs (see Graduate Section 3B.1 and Undergraduate Section 4A.4), and there is now a student ACM chapter. There are also undergraduate seminars that expose students to research topics and industrial experiences (outside speakers). We have begun (this current year) appointing a dedicated advisor to the Student Advisory Committee, and there are active discussions about how student groups could become involved in promoting activities outside of the classroom. Finally, the School has been actively fielding teams for programming competitions (we sent a team to the international finals in Thailand in 2016) and has hosted several local hack-a-thons.

At the graduate level, there has begun a fairly broaden discussion about *student life*, with an eye towards improving the overall experiences of graduate students and help alleviate the risk of isolation associated with the high-pressure environment that the program creates. There is now an advisor to the Graduate Student Advisory Committee, and the current instantiation of that committee is organizing social activities and planning to formulate a *mission statement* for this committee. We have also developed a significant *Distinguished Lecture Series*, sponsored by Goldman Sachs, to bring in high-profile visitors and expose our student body (and others) to state of the art topics in computer science.

Recommendation 4: Diversity. The School should formulate and implement efforts to recruit minority and female faculty and students to achieve appropriate diversity among its body. The Office of the Associate Vice President for Equity and Diversity is committed to this goal and may provide useful ideas and strategies in this regard. The use of annual progress reports to the Graduate School should be considered as a way to encourage the School to work effectively towards this goal.

Diversity is a consistent topic of discussion among faculty, and the awareness and importance of this issue has been raised significantly in the last 5-10 years. The goal is to increase the presence of underrepresented groups in our faculty and student body. There are several measures that we have taken to achieve this, including active recruitment and increased mentoring. Much of the explicit activity has focused on women, because they are so severely underrepresented, and they represent the greatest potential for improvement. The results are positive, we have significantly better representation of women in our undergraduate and graduate populations, and we have added three new female faculty members (and promoted one other) since our last review. Percentages of other underrepresented groups have also improved. More details and data are given in Section 2.

1.4 Department Profile

	2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015
Full Time Tenured Faculty	22	25	24	22	22	24	23
Full Time Tenure Track	10	6	5	8	11	9	10
Full Time Career Line	6	5	7	9	9	9	8
Part Time Tenure/Tenure Track			2	3	1	1	3
Total	38	36	38	42	43	43	44

Table 1-3: Faculty Headcount - Source OBIA

	2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015
Undergraduate Pre-Majors	154	184	204	272	353	471	597
Undergraduate Majors	294	313	366	414	421	453	466
Enrolled in Masters Program	86	100	127	128	131	151	165
Enrolled in Doctoral Program	123	123	127	121	121	129	122
Total	657	720	824	935	1026	1204	1350

Table 1-4: Enrolled Majors - Source OBIA

		2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014
SCH	Lower Division	4,053	4,608	4,743	5,259	5,436	6,458
	Upper Division	5,115	5,301	5,747	6,810	7,299	8,596
	Basic Graduate	2,340	2,461	2,736	3,123	3,389	4,731
	Advanced Graduate	1,843	1,848	1,901	1,634	1,518	1,621
FTE	Lower Division	135	154	158	175	181	215
	Upper Division	171	177	192	227	243	287
	Basic Graduate	117	123	137	156	169	237
	Advanced Graduate	92	92	95	82	76	81
FTE/FTE	LD FTE per Total Faculty FTE	3	4	3	3	4	5
	UD FTE per Total Faculty FTE	4	4	4	4	5	6
	BG FTE per Total Faculty FTE	3	3	3	3	3	5
	AG FTE per Total Faculty FTE	2	2	2	2	1	2

Table 1-5: Student Credit Hours and FTE - Source OBIA

	2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015
Undergraduate Certificate							
Graduate Certificate					1		
Bachelors	82	82	92	90	108	114	117
Masters	40	44	51	51	60	56	92
Doctorate	13	16	12	15	19	19	21

Table 1-6: Degrees Awarded - Source OBIA (BS degrees includes ½ of CE degrees granted)

	2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015
Total Grants	$5,209,953	$5,727,917	$5,917,936	$6,109,168	$7,586,835	$8,781,791	$10,745,237
State Appropriated Funds	$5,869,202	$5,405,025	$ 5,500,259	$5,699,661	$5,803,257	$6,128,556	$7,566,343
Teaching Grants	$540,565	$453,083	$550,790	$579,145	$807,239	$190,417	$319
Special Legislative Appropriation							
Differential Tuition			$596,416	$494,983	$465,897	$556,300	$660,600
Total	$11,619,720	$11,586,025	$12,565,401	$12,882,957	$14,663,228	$15,657,064	$18,972,499

Table 1-7: Funding Source OBIA

	2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015
Expenditures	$11,224,010	$17,657,863	$14,174,188	$16,484,777	$17,214,144	$18,492,294	$19,251,560

Table 1-8: Research Spending - Source Dean's Office

	2008 – 2009	2009 – 2010	2010 – 2011	2011 – 2012	2012 – 2013	2013 – 2014	2014 – 2015
Direct Instructional Expenditures	$5,816,921	$5,971,781	$5,981,659	$6,850,875	$7,585,059	$8,452,161	$9,301,806
Cost Per Student FTE	$11,302	$10,943	$10,287	$10,703	$11,325	$10,315	$12,556

Table 1-9: Cost Study - Source OBIA

2. Faculty

2.1. Faculty Profile

As of Fall 2016 the School of Computing has 53 non-auxiliary faculty members (40 in 2009). The number of tenure-line (regular) faculty has gone from 32 to 40. The number of career-line faculty has gone from 4 to 6. Table 2-1 lists the regular faculty and their primary areas of expertise. Table 2-2 gives the numbers of faculty (by appointment type) for the current and previous review period. Tables 2-3 through 2-4 give data on faculty who have left the School and who have been hired since the last review.

Name	Area
	Tenure-Track Assistant Professors
Aditya Bhaskara	Theoretical computer science and machine learning
Mahdi Bojnordi	Computer architecture, new memory technologies
Tammy Denning	Security and privacy, human-centric computing
Tucker Hermans	Autonomous learning and perception in robots
Ladislav Kavan	Computer graphics and animation
Alexander Lex	Interactive data visualization, applied to molecular biology and pharmacology
Miriah Meyer	Information visualization, human-centered visualization
Jeff Phillips	Geometric data analysis, algorithms for big data
Zvonimir Rakamaric	Formal methods, software reliability, software resilience
Vivek Srikumar	Machine learning and natural language processing
Ryan Stutsman	Formal methods, software reliability, software resilience
Hari Sundar	Parallel algorithms, scientific computing, image analysis
Bei Wang- Phillips	Topological data analysis, scientific visualization, information visualization
Jason Wiese	Personal data, HCI
Cem Yuksel	Computer graphics
	Tenured Associate Professors
Erik Brunvand	Computer architecture and VLSI systems
Tom Fletcher	Medical image analysis and computer vision
Feifei Li	Database, Big data analytics, large-scale data management systems
Matt Might	Security, parallelism and optimization via program analysis
Kobus Van der Merwe	Networking systems
Suresh Venkatasubramanian	High dimensional geometry, clustering, kernels, large data models
	Tenured Professors
Rajeev Balasubramonian	Computer architecture, cutting-edge memory systems
Martin Berzins	Adaptive numerical methods, scalable parallel computing
Richard Brown	VLSI, microprocessor design, biomedical electronics, sensors
Elaine Cohen	Modeling, graphics, and visualization, geometric computation and analysis
Matthew Flatt	Extensible programming languages
Ganesh Gopalakrishnan	Rigorous correctness checking of software and hardware
Mary Hall	Parallel computing, compiler optimization, performance tuning
Chuck Hansen	Scientific visualization, GPU algorithms
Tom Henderson	Autonomous systems, cognitive robotics, smart sensor networks
John Hollerbach	Robotics, teleoperation, virtual reality, and human motor control
Chris Johnson	Visualization, scientific computing, image analysis
Sneha Kasera	Networks and systems
Bob Kessler	Video games for health and software engineering
Mike Kirby	Scientific computing and visualization
Valerio Pascucci	Computer graphics, computational geometry, geometric programming
John Regehr	Software testing and reliability, compilers, embedded systems, operating systems
Ellen Riloff	Natural language processing, information retrieval, and artificial intelligence
William Thompson	Vision science, spatial organization, perception and graphics
Ross Whitaker	Image and geometric processing, data and medical image analysis, visualization

Table 2-1: Summary of current School faculty

Title	2009	2016
Assistant Professor (TT)	5	15
Assistant Professor Lecturer (CL)	5	1
Associate Professor (T)	10	6
Associate Professor Lecturer (CL)	0	3
Distinguished Professor (T)	0	1
Professor (T)	16	19
Professor Lecturer (CL)	0	1
Research Assistant Professor (CL)	4	3
Total	40	49

Table 2-2: Tenure and Rank.

For reference: tenure track (TT), career line (CL), tenured (T)

Name	Position	Gender	Race	Reason	Departed
Bargteil, Adam	Assistant Professor	Male	White	Not Tenured	Summer 2015
Daume, Harold	Assistant Professor	Male	White	Resigned	Spring 2010
Davis, Alan	Professor	Male	White	Retired	Spring 2015
Gerig, Guido	Professor	Male	White	Resigned	Spring 2015
Hollaar, Lee	Professor	Male	White	Retired	Spring 2014
Sikorski, Christopher	Professor	Male	White	Deceased	Summer 2012
Silva, Claudio	Professor	Male	Hispanic	Resigned	Spring 2011
Silva, Juliana Freire	Professor	Female	Hispanic	Resigned	Spring 2011
Van den Berg, Jur	Assistant Professor	Male	White	Resigned	Fall 2013

Table 2-3: Faculty no longer with the School

Name	Gender	Race
Bhaskra, Aditya	Male	Asian
Denning, Tamara	Female	White
Hermans, Tucker	Male	White
Kavan, Ladislav	Male	White
Lex, Alexander	Male	White
Li, Feifei	Male	Asian
Meyer, Miriah	Female	White
Nazm Bojnordi, Mahdi	Male	White
Phillips, Jeff	Male	White
Rakamaric, Zvonimir	Male	White
Srikumar, Vivek	Male	Asian
Stutsman, Ryan	Male	White
Sundar, Hari	Male	Asian
Van der Merwe, Kobus	Male	White
Wang, Bei	Female	Asian
Wiese, Jason	Male	White
Young, Michael	Male	White
Yuksel, Cem	Male	White

Table 2-4: New faculty

Title	Average Age	Minimum Age	Maximum Age	SDev of Age
Assistant Professor (TT)	36	29	50	5
Assistant Professor Lecturer (CL)	53	48	57	6
Associate Professor (T)	44	35	56	9
Associate Professor Lecturer (CL)	41	32	48	8
Distinguished Professor (T)	56	56	56	0
Professor (T)	56	39	71	10
Professor Lecturer (CL)	58	58	58	0
Research Assistant Professor (CL)	38	34	48	6

Table 2-5: Faculty age statistics

2.2. Faculty Diversity

The largest demographic in our department's faculty is white males at 71%. In general, for a breakdown of our demographics, we have 81.6% whom have identified as white, 9.2% Asian, 7.9% did not identify and 1.3% Hispanic. With respect to gender, 11.8% are female, and 88.2% are male. Tables 2-5 through 2-8 give the relevant demographic data on the faculty. For faculty no longer with the School, or hired into the School since 2009, see Tables 2-3 and 2-4 in the 2-1 Faculty Profile Section.

During the last seven years, in net, we have added eight additional faculty as a consequence of 19 new hires. The summary breakdown of the new faculty, also shown in Table 2-4 – three females, sixteen males, fourteen identified as white, and five as Asian.

University policy provides a well-defined structure for recruitment and hiring involving minority advertisements and oversight of internal procedures and applicants. As part of this system, the School regularly advertises faculty job opportunities in the following venues: Hispanic Outlook Magazine, Diverse: Issues for Higher Education. The School has also advertised in SWE magazine.

Title	Female	Male
Assistant Professor (TT)	2	13
Assistant Professor Lecturer (CL)		1
Associate Professor (T)		6
Associate Professor Lecturer (CL)	1	2
Distinguished Professor (T)		1
Professor (T)	3	16
Professor Lecturer (CL)		1
Research Assistant Professor (CL)		3
Total	6	43

Table 2-6: Title/tenure by gender

Title	Asian	Hispanic	Not Identified	White
Assistant Professor (TT)	3	0	1	11
Assistant Professor Lecturer (CL)	0	0	0	1
Associate Professor (T)	0	0	3	3
Associate Professor Lecturer (CL)	0	0	0	3
Distinguished Professor (T)	0	0	0	1
Professor (T)	3	0	2	14
Professor Lecturer (CL)	0	0	0	1
Research Assistant Professor (CL)	0	0	0	3
Total	6	0	6	37

Table 2-7: Title/tenure status by race

	Asian	Not Identified	White
International	7	1	6
Domestic	0	1	34
Total	7	2	40

Table 2-8: Distinguishing faculty race by domestic or international

The 2015 Taulbee Survey, shows the School is slightly lower then the national trend in its hiring practices of females. Since 2009, 15.8% of our new hires have been female; the CRA report shows an average of 20.3% for tenure track positions (2014-2015 survey). The same 2015 Taulbee Survey also shows how we follow the trends in regards to race. Of our new hires 16% have identified as Asian and 83% as white. The Taulbee survey indicates 44.8% whites in tenure track positions and 27.6% for Asians. In the past three years, tenure-track offers were made to three other female candidates who did not accept our offers.

	Tenure-Track		Teaching		Research		Postdoc		Total	
Male	255	79.7%	127	75.1%	39	75.0%	108	80.6%	529	78.4%
Female	65	23.0%	42	24.9%	13	25.0%	26	19.4%	146	
Unknown	0		1		1		14		16	
Total	320		170		53		148		691	

Table 2-9: 2015 Taulbee Survey: Gender of newly hired faculty

	Tenure-Track		Teaching		Research		Postdoc		Total	
Nonresident Alien	46	15.9%	12	7.8%	14	26.9%	67	53.2%	139	22.3%
American Indian/Alaskan Native	0	0.3%	0	0.0%	0	0.0%	0	0.0%	0	0.2%
Asian	80	27.6%	23	14.9%	14	26.9%	28	22.2%	145	23.3%
Black or African American	9	3.1%	2	1.3%	0	0.0%	2	1.6%	13	2.1%
Native Hawaiian/Pacific Islander	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
White	130	44.8%	112	72.7%	22	42.3%	25	19.8%	289	46.5%
Multiracial, not Hispanic	2	0.7%	0	0.0%	0	0.0%	1	0.8%	3	0.5%
Hispanic, any race	9	3.1%	2	0.3%	0	0.0%	1	0.8%	12	1.9%
Resident, race/ethnic unknown	13	4.5%	3	0.9%	2	3.8%	2	1.6%	20	3.2%
Total Known Residency	290		154		52		126		622	
Residency Unknown	30		16		1		22		69	
Total	320		170		53		148		691	

Table 2-10: 2015 Taulbee Survey: Ethnicity of newly hired faculty

2.3. Faculty Teaching

The faculty teaching record has remained strong, based on several sources of data. Table 2-12 through 2-14 shows the Schools average for student teaching evaluations in response the questions about the course compared against the College and the University. These results suggest that student teaching evaluations for the School compare favorably. We can also see, from Table 2-12) that very few classes each year earn teaching evaluations that are below the 4.0 level.

Every semester the Dean's office recognizes faculty in the College who are among the top 15% (among all College faculty) in their student teaching evaluations (sometimes called the "Dean's list"). The School typically has from 12-17 faculty (out of a total of 35–40, 25–30%) who earn this distinction (Table 2-15). Likewise, the School faculty have won several College and University level teaching awards since the last review (Table 2-17).

The School promotes effective teaching in several ways. First, peer teaching evaluations are conducted (and reported) for all informal and formal reviews (every year for tenure track, for every promotion, and every five years after tenure). The Student Advisory Committees (graduate and undergraduate) evaluate and vote on every promotion, and they are invited to give input on every proposed hire. The School also recognizes an outstanding professor/teacher every year (Table 2-16).

The School hires graduate students, post docs, and adjunct professors to teach on occasion. However, the policy is to try to cover all classes with tenure-line or instructional faculty. Typically 2-3 classes per year are taught in this fashion. A significant number of credit hours (approximately 1/3 of all classroom hours) are taught by instructional/lecturing faculty.

Year	Average GRP
2009	5.57
2010	5.51
2011	5.56
2012	5.50
2013	5.58
2014	5.56
2015	5.50
2016	5.55
Grand Total	5.54

Table 2-11: Department average for Question Group (GRP), based by course

GRP Range	2014	2015	2016
< 3.2 or (blank)	0	1	3
3.2 – 3.5	1	0	0
3.5 – 3.8	2	2	0
3.8 – 4.1	2	3	2
4.1 – 4.4	3	7	7
4.4 – 4.7	7	16	7
4.7 – 5.0	18	16	15
5.0 – 5.3	34	49	29
5.3 – 5.6	65	85	34
5.6 – 5.9	48	55	23
> 5.9	103	107	90
Grand Total	283	341	210

Table 2-12: Department GRP count by range for the last three years

Semester	Instructor	CS	College	University
Fall 2013	4.94	5.24	5.18	5.28
Spring 2014	5.13	5.1	5.12	5.31
Summer 2014	5.32	5.34	5.21	5.22
Fall 2014	5.22	5.24	5.25	5.27
Spring 2015	5.12	5.21	5.18	5.3
Summer 2015	4.75	5.12	5.05	5.16
Fall 2015	5.44	5.15	5.2	5.3
Spring 2016	5.12	5.23	5.11	5.09
Summer 2016	5.71	5.82	5.18	5.26

Table 2-13: GRP by instructor at different institutional levels

Semester	Course	CS	College	University
Fall 2013	5.54	5.12	5.08	5.15
Spring 2014	4.81	5.04	5.05	5.18
Summer 2014	5.31	5.3	5.21	5.22
Fall 2014	5.38	5.2	5.09	5.12
Spring 2015	4.99	5.09	5.08	5.16
Summer 2015	4.63	5.14	5.05	5.16
Fall 2015	5.5	5.0	5.08	5.14
Spring 2016	5.04	5.1	5.11	5.09
Summer 2016	5.96	5.82	5.18	5.26

Table 2-14: GRP by course at different institutional levels

	2010	2011	2012	2013	2014	2015
Rajeev Balasubramonian				X		X
Martin Berzins						
Erik Brunvand	X		X	X	X	X
Elaine Cohen						
Tammy Denning						
Jim de St. Germain	X		X	X	X	
Matthew Flatt	X	X		X		X
Thomas Fletcher	X	X				
Ganesh Gopalakrishnan						
Mary Hall			X			
Chuck Hansen	X			X		X
Tom Henderson						
Tucker Hermans						
John Hollerbach						
Peter Jensen	X	X	X	X	X	X
Christopher Johnson						X
David Johnson			X	X	X	X
Sneha Kasera						
Bob Kessler	X		X	X		
Mike Kirby				X	X	
Alexander Lex						
Feifei Li				X	X	
Matt Might			X	X	X	
Miriah Meyer				X		
Erin Parker	X	X	X	X		X
Valerio Pascucci	X	X				
Jeff Phillips			X			
Zvonimir Rakamaric				X		X
John Regehr	X				X	
Ellen Riloff				X	X	
Vivek Srikumar						
Ryan Stutsman						X
Hari Sundar						
William Thompson	X	X			X	
Kobus Van der Merwe					X	X
Suresh Venkatasubramanian	X					
Ross Whitaker		X	X	X		
Cem Yuksel				X		X
Joe Zachary						X

Table 2-15: College of Engineering recognition of teachers ("Dean's list") for the School

Year	Recipient
2010	Joe Zachary
2011	Erin Parker and Peter Jensen
2012	Peter Jensen and Erik Brunvand
2013	Matthew Might
2014	John Regehr
2015	Miriah Meyer

Table 2-16: School of Computing Outstanding Teaching Awards

Recipient	Award
Erin Parker	2011 College of Engineering Outstanding Teaching Award
Joe Zachary	2015 College of Engineering Outstanding Teaching Award
Valerio Pascucci	2016 University of Utah Mentoring Award

Table 2-17: Teaching awards to the School faculty

2.4. Faculty Scholarship

The faculty in the School are extremely productive in research and scholarship. This is evident in the publications and research expenditures, as compared against other units in the University and against other CS institutions around the country. Table 2-18 shows research expenditures since the last review period, which have almost doubled (with a 25% increase in number of faculty). It also shows numbers of publications (articles with a faculty author, co-authorships counted once). Research activity continues to increase—the total value of new awards in (fiscal) 2016 is $29,644,364.

Visibility with the field also suggests a highly productive faculty. The faculty/students published approximately 234, 198, and 214 papers over the years 2013, 2014, and 2015, respectively. The faculty are/have been on 25 editorial boards and four have served as editors-in-chief of journals in the past three years. In the last three years, faculty members have served as conference committee chairs 107 times (see Table 2.19). SoC faculty have served on numerous national boards, including the CCC, the CRA, and PCAST, as well as serving on many national and international task forces and review boards, including several under the auspices of the NSF, the NIH, ACM, and IEEE. Since the past review, several faculty have been honored as IEEE Fellows, ACM Distinguished Scientists, and AIMBE fellows.

Fiscal Year	Total
2010	$10,483,257
2011	$14,174,188
2012	$16,484,777
2013	$17,214,144
2014	$18,492,294
2015	$19,127,939
2016	$21,294,593

Table 2-18: Research expenditures

Fiscal Year	Total
2016	$29,644,364

Table 2-19: Fiscal year data vs. 2-17 calendar year data

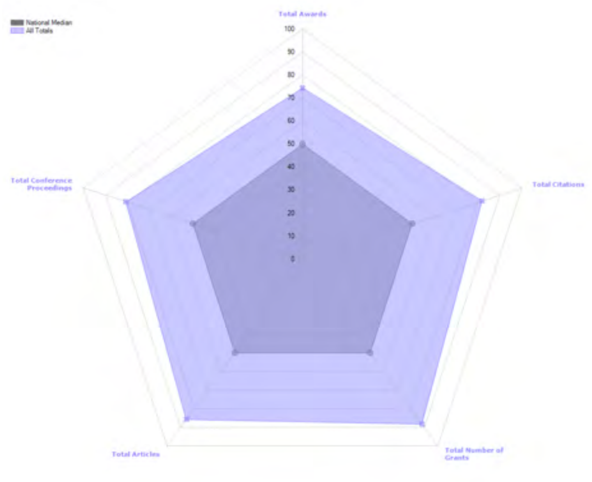
Director of Outreach	Office of the Director
Education Outreach Coordinator	Peter Jensen
Industrial Liaison	Miriah Meyer
School RPT	Ellen Riloff
Regular Colloquia	Zvonimir Rakamaric
Organick Lecture Series	Ganesh Gopalakrishnan
Distinguished Lecture Series	Suresh Venkatasubramanian
Space	Office of the Director
Facilities Liaison	John Regehr
Research Grants	Office of the Director
CES Program Director	Martin Berzins
Diversity Committee Director	Mary Hall (chair)
	David Johnson
	Miriah Meyer
	Bill Thompson
Faculty Awards Liaison	Chris Strong
Dept. Safety Officer	Chris Strong

Table 2-20: 2015 - 2016 Department committee assignments

	2013	2014	2015
Chairs	39	34	34
Editors/Editorial Boards	29	24	23
Boards	22	28	24

Table 2-21: Editorial/advisory activities (number of instances) by year

The data shows that the research productivity of the School compares favorably against other CS institutions nationally. Table 2-22 shows a summary from an Academic Analytics (http://www.academicanalytics.com) evaluation of the productivity of the School. Several trends are worth noticing. First, the School typically ranks near the 80th percentile across most performance indices. Second, there are some general trends that confirm several qualitative observations about the School. For instance, with regard to research expenditures (grants), the School is among the top institutions in the nation (near 90%). Publications and citations are very strong (80–90%). The weakest category is awards, where the School ranks in the 70-80% range. This suggests an opportunity. The productivity of the faculty is somewhat above what is generally acknowledged by the field as a whole, and thus appropriately publicizing the status of the School could reap noticeable benefits.



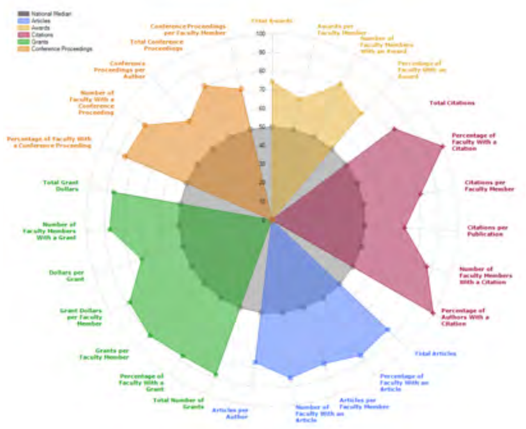


Table 2-22: Graphics from *Academic Analytics* show quantified scholarship performance relative to other research institutions nationally

2.5. Faculty Service

The faculty are engaged in service for both the University (including the School/College) and the field. Service is evaluated during all formal and informal faculty reviews. Within the School, service is assigned by the Director's office in a subjective manner to balance needs, capabilities, and workload. Exceptional levels of service within the School are accounted for qualitatively and recognized during promotion and through raises (as determined by the Director). Since 2013, Directors Whitaker and Kirby have compiled a comprehensive list of internal (School) service assignments and organized service committees in a hierarchical manner (e.g. all things relating to the graduate program report to the Director of Graduate Studies).

Director	Ross Whitaker
Associate Director	Mike Kirby
Executive Committee	Mary Hall
	Feifei Li
	Suresh Venkatasubramanian
	James de St. Germain
	Sneha Kasera
	Mike Kirby
	Ross Whitaker (Chair)
Director of Graduate Studies	Feifei Li
Chair of Graduate Admissions Committee	Matthew Flatt
Track Directors	
Computer Engineering	Erik Brunvand
Data Management	Jeff Phillips
Digital Media	Erik Brunvand
Graphics & Visualization	Chuck Hansen
Image Analysis	Tom Fletcher
Networked Systems	Sneha Kasera
Robotics	John Hollerbach
Scientific Computing	Hari Sundar
BS/MS	Cem Yuksel
Student Awards/Fellowship Liaison	Elaine Cohen
Director of Undergrad Studies	James de St. Germain
Undergraduate Studies Committee	Zvonimir Rakamaric (chair)
	James de St. Germain
	Bob Kessler
	Erik Brunvand
	Tammy Denning
	Erin Parker
	Tom Henderson
	Matthew Flatt

Table 2-23: 2015 - 2016 Department committee assignments

Director of Outreach	Office of the Director
Education Outreach Coordinator	Peter Jensen
Industrial Liaison	Miriah Meyer
School RPT	Ellen Riloff
Regular Colloquia	Zvonimir Rakamaric
Organick Lecture Series	Ganesh Gopalakrishnan
Distinguished Lecture Series	Suresh Venkatasubramanian
Space	Office of the Director
Facilities Liaison	John Regehr
Research Grants	Office of the Director
CES Program Director	Martin Berzins
Diversity Committee Director	Mary Hall (chair)
	David Johnson
	Miriah Meyer
	Bill Thompson
Faculty Awards Liaison	Chris Strong
Dept. Safety Officer	Chris Strong

Table 2-24: 2015 - 2016 Department committee assignments

Academic Appeals and Misconduct Committee Chair	Feifei Li
Women in Engineering Faculty Council	Mary Hall
College Scholarship Committee	Erin Parker

Table 2-25: 2015 - 2016 College assignments

Academic Senate	Rajeev Balasubramonian
Cyber Infrastructure Council	Kobus Van der Merwe
Presidential Commission on the Status of Women	Mary Hall
UPTAC	Chuck Hansen
Grad Council	Chuck Hansen
Graduate School Admissions Committee	Chuck Hansen
Senate Faculty IT Ad Hoc Committee	Peter Jensen
University Teaching Committee	Erik Brunvand
Director, Scientific Computing and Imaging Institute	Christopher Johnson
Vice President for Research Search Committee	Christopher Johnson
Distinguished Professor Advisory Committee	Christopher Johnson
Dean of Dentistry Search Committee	Christopher Johnson
Neuroscience Initiative Executive Committee	Christopher Johnson
Chair of Radiology Search Committee	Christopher Johnson
Biodiversity Cluster Search Committee	Christopher Johnson
Entrepreneurial Faculty Advisor	Christopher Johnson

Table 2-26: 2015 - 2016 University assignments

2.6 Retention, Promotion and Tenure (RPT)

The School follows RPT policies that are consistent with the policies of the University and the College (organized hierarchically). The policy for the School (adopted April 2, 2008) is located on the WWW: http://www.cs.utah.edu/docs/Faculty/RPT2008.pdf, and is included in this report as Appendix C.

To summarize, RPT evaluations are done by the RPT committee of faculty, which consists of all faculty of the appropriate rank (excluding those on leave, sabbatical, etc.). This committee meets and discusses each individual case and assigns a subcommittee to draft a report of the proceedings. This report is then discussed and finalized (usually by email in the final stages). The School's Director then makes a recommendation in light of that report. For formal cases, these evaluations/recommendations then move upstream through the University Administration.

The procedures for tenure-track faculty are thorough and well documented. There is an annual informal report, which is prepared by a subcommittee after a meeting of the full RPT committee. The Director then writes an accompanying letter and meets with each tenure-track candidate *every* year. These annual reports are generally very thorough, with explicit statements of accomplishments relative to expectations and recommendations for improvement. There is also a mid-term formal review (at year three), which entails a procedure that is virtually identical to an tenure evaluation, including letters from outside experts and an formal evaluation of individual performance criteria (teaching, scholarship, service). This tenure-track, mid-term review includes a formal vote on *retention*, with the option of recommending that a faculty member be removed from the tenure-track appointment.

Post-tenure reviews are every five years, with the option of requesting consideration for promotion to Professor (*Full* Professor). These regular, post-tenure reviews are conducted by a pair of faculty (appointed by the Director for each case), who evaluate CV, date from annual Faculty Activity Reports, teaching evaluations (and peer evaluations), and write a report summarizing a faculty member's (subject) performance/productivity relative to expectations. This pair of faculty meet with the subject to discuss this findings. The Director writes a letter summarizing findings and evaluating those relative to the School's expectations, and making recommendations for improvements, etc. The Director meets with the subject about these findings.

Table 2-27 summarizes the outcomes of various formal RPT actions since the period of the last review.

Academic Year	Name	Action	Outcome
2009– 2010	None		
2010 – 2011	Valerio Pascucci	R, P & T	Tenured, Promoted to Professor
	Suresh Venkatasubramanian	Retention	Retained
	Thomas Fletcher	Retention	Retained
2011–2012	Mary Hall	Promotion to Professor	Promoted to Professor
	Matthew Might	Retention	Retained
2012–2013	Feifei Li	R, P & T	Tenured, Promoted to Assoc. Professor
	Suresh Venkatasubramanian	R, P & T	Tenured, Promoted to Assoc. Professor
2013–2014	Matthew Flatt	Promotion to Professor	Promoted to Professor
	Ellen Riloff	Promotion to Professor	Promoted to Professor
	R. Michael Kirby	Promotion to Professor	Promoted to Professor
	Matthew Might	R, P & T	Tenured, Promoted to Assoc. Professor
	Thomas Fletcher	R, P & T	Tenure, Promoted to Assoc. Professor
	Jeff Phillips	Retention	Retained
	Miriah Meyer	Retention	Retained
2014–2015	Sneha Kasera	Promotion to Professor	Promoted to Professor
	Rajeev Balasbramonian	Promotion to Professor	Promoted to Professor
	John Regehr	Promotion to Professor	Promoted to Professor
	Jakobus Van Der Merwe	Retention and Tenure	Tenured
	Cem Yuksel	Retention	Retained
	Zvonimir Rakamaric	Retention	Retained
2015–2016	None		

Table 2-27: Retention, Promotion & Tenure (RP&T) Summary

2.7 Faculty Vitae

See Appendix D.

3A. Undergraduate Students

3A.1 Undergraduate Student Recruitment

3A.1.1 Recruiting undergraduate students

The School of Computing undergraduate advisor participates in many recruiting events alongside other advisors in the College of Engineering. These events include presentations to nearby universities, hosting tables at campus wide major exploration events and participating in College of Engineering Day which invites high school and transfer students from across the state to preview the College. Last year, the School participated in the "Friday Afternoon in Engineering" days, hosting one Friday presentation to recruit current, transfer and prospective students.

For educational outreach, the School at the University of Utah is committed to exposing K-12 students to the exciting and far-reaching field of computer science. The School offers outreach programs that encourage students to pursue careers in computer science and engineering. Several programs are specifically designed to interest women and other underrepresented groups in computer science.

Current programs directly utilizing the School staff include:

- GREAT (Graphics & Robotic Explorations with Amazing Technology) summer camp for 8-10 graders.
- EAE (Entertainment Arts & Engineering) summer camp for high schoolers – Working with YouthEd to offer over 25 courses ranging from game design, to scratch, to 3D modeling – for a wide range of ages.
- Hi-GEAR engineering summer camp for high school females, EYH (Expanding Your Horizons) conference workshop for girls in grades 6-9.
- Provide support to FIRST LEGO League (where teams of ages 9-14 compete)
- Girl Scouts Engineering Night for girls in grades 3-6.
- Various school visits.

<u>College of Engineering Recruiting Activities:</u>

The College is very active in both recruitment and K-12 outreach, and it sponsors a number of programs to increase participation of students in STEM activities. Some programs offered through the College include:

- Elementary engineering days, where the Academic Affairs team travels to elementary schools to provide hands-on engineering activities for each grade.

- A lunchtime speaker series offered to high schools sending members of the Engineering Alumni Association to discuss the career paths of practicing engineers as a method for helping students explore career opportunities in engineering,
- Presentations offered in junior high and high school science and math classes.
- A teacher lending library that includes lesson plans, videos and hands-on engineering activities that tie directly into the curriculum and used as a way to augment classroom learning.
- Engineering Day in the fall.
- "Meet an Inventor Night" in the spring.
- Two summer camps: a HI-GEAR camp offered only to high school aged girls and an Exploring Engineering Camp open to all high school students.
- A mentoring program (College of Engineering Ambassadors) made up of students representing each department in the College and trained to go out into high schools to bring hands on engineering activities that help to connect engineering to the high school curriculum.
- Working with our Partner Schools, Frost Elementary, Western Hills Elementary, Valley Jr. High, Kearns Jr. High, Kearns High School, Granger High School, teams of students hired from the membership of College diversity clubs, Society of Women Engineers (SWE), Science, Technology, Engineering Programs (STEP), Society of Professional Hispanic Engineers (SHPE), and National Society of Black Engineers (NSBE), we have established engineering clubs to engage elementary, junior high, middle school and high school students in engineering activities. They plan and facilitate two on campus activities for the junior high and high school students.
- The College also participates in travel to meet and recruit prospective students both transfer and new freshmen sponsored by the University Office of Admissions. This travel is out-of- state in the fall and six events around the state of Utah in the spring.
- The College participates with other campus partners in encouraging students to explore engineering and science through on-campus activities. (Red, White and U Day in the spring for admitted students and Connecting U Days in the fall and spring for high school juniors and seniors)
- The College hosts monthly information sessions for prospective students and work with student groups and classes to facilitate campus visits.

3A.1.2 Retaining undergraduate students

Advisors have a mandatory first year meeting with all freshman students in their first semester to help them register for their second semester and go over all program requirements. Advisors also meet with students for a second year mandatory meeting to check on their progress and encourage them to return the following semester. We seek to retain students of merit by awarding academic scholarships each year as well as placing them in undergraduate TA positions.

The School also has a strict probation policy, where students who have low GPAs are advised on the criteria necessary to maintain status in the program. We have raised the continuing GPA from 2.3 to 2.5 to try and catch students more quickly and provide them the resources to succeed. For those students who do go on probation (or who are removed from the program due to probation violations) we work to make individualized plans to help the student improve (or if necessary, re-attain Full Major Status (FMS) – see the section on admitting Undergraduate Students).

From 2009-2014, the CS Bachelor's degree has graduated 65% of its students who were admitted to FMS. The University average graduation rate is 58%.

3A.1.3 Evaluating and admitting undergraduate students

The School admits students into the program after their first year, based on successfully completing a sequence of "pre-major" courses (CS 1410 - OO Programming, CS 2420 – Data Structures and Algorithms, Calc I, and Calc II). The first year courses, especially CS 2420, are considered strong predictors of college success overall and CS success in particular. It is unfortunately the case that many students are not prepared for the rigor of computer science coursework. In general, 30% of CS 1410 students do not complete the course at a level allowing progression to CS 2420, and 30% of CS 2420 students do not complete the course at a level allowing progression into the major. This trend has been observed regardless of professor and pedagogy; that being said, the School is aware that what it means to be a "Computer Scientist" is changing, and we are discussing what, if any, changes might be necessary. There is a strong sentiment that those students who earn a CS degree are very skilled and marketable; there is a related sentiment that there are other students who might not be able to complete a CS degree, but could be trained to be programmers perhaps through some sort of "computing" degree (i.e., a less rigorous, more application based degree). For example, not all students need to be able to create a new machine learning algorithm, but there is growing evidence that many industries would like to be able to apply machine learning to their data to find trends.

Below, we discuss the formal definitions of our major statuses, as well as the requirements to transition between them. It should be noted that we are in the midst (as of 2016) of two large transitions: 1) first year admission (of potential students) will now go through the College, and 2) we are looking to allow more students into the CS major based on a more holistic analysis of their likelihood of success.

Pre-Major and Major Status: Prior to 2016, any student could become a computer science pre-major by informing the University Registrar or the School Academic Advisor. Moving forward, new students will be admitted to the College, and at that point declare an interest, such as Computer Science. Regardless of "pre-major" status, students must complete the pre-majors courses (CS 1410 and 2420, Calc 1 and 2), and then apply for full major status (FMS).

Students apply on the School website following the semester in which they complete the final pre-major requirements. Applications are accepted in both spring and fall semesters. Note: most upper-division classes in Computer Science are restricted to full majors (or minors).

Evaluating Applicants (policy from 2009-2016): Once grades for the prior semester have been posted, a committee of instructional faculty review all applicants. Applicants with University and pre-major GPAs of 3.5 or higher who have adhered to the policy on repeating courses are assured admission to the computer science major.

Applicants with University and pre-major GPAs of 3.0 or higher who have adhered to the policy on repeating courses and have one or both GPAs below 3.5 are carefully considered. Due to space constraints, not all applicants in this category may be admitted. The goal of the Admissions Committee is to select those students with the most promise for success in our challenging degree program.

Below are some of the criteria used to evaluate an applicant's potential for success:

- Applicants with an A or A- grade in CS 2420 are very promising; applicants with a C+ or C grade have very little chance of success in upper-level CS courses.
- Applicants on an upward trajectory, with grades that steadily improve with each CS and MATH course taken, show promise. Applicants on a downward trajectory have very little chance of success in upper-level CS courses.
- Applicants who have repeated one or zero courses show great potential to complete the degree requirements directly and efficiently. Applicants who have repeated multiple courses show a concerning pattern that may continue.
- If other CS, math, physics, and engineering courses have been taken by the applicant, those grades are considered. In particular, A or A- grades in Discrete Structures, Calc 3, and Linear Algebra demonstrate the ability to perform well in upper-level courses.

Academic Statement
Applicants may optionally submit a statement to support their application. The purpose of this statement is to guide the Admissions Committee in recognizing factors in one's academic performance that may not be directly evident from the University and pre-major GPA. Applicants who have previously been denied admission are encouraged to use the statement to point out significant improvement since the last application.

Results of Application policy: The highly competitive nature of this process, including **strict requirements and limited seats** for FMS positions has resulted in **high caliber** potential students (average first year GPAs being over 3.6) but may also have **discouraged students** from continuing to pursue a career in CS (e.g., it may discourage students from starting the first year, completing the first year once started, or even applying once the first year is completed). To address this possibility, we are modifying our admissions process (see below).

*Evaluating Applicants (**policy post 2016**):* The School is looking to carefully expand the number of new majors by tweaking the previous admission criteria. The goal of the admissions process is, as always, to try and predict likely measures of success. In the future, we will continue to look at the GPA of students, but will stress it less; we hope this will encourage applications from a larger and more diverse population. We will also be stressing that students with an "alternative" background may petition the Director of Undergraduate Studies to talk directly about obstacles they have overcome and why they would make a good (non-traditional) admit to the School.

Discussion of new policy consequences: Historically, once admitted to the program (FMS), 30% of our students do not graduate. Of this group, two-thirds do not do so because of overall poor grades or the inability to pass one or more required upper division CS courses. Thus 20% of the School students (those who were able to successfully be considered for FMS) are unable to meet the rigor, high expectations, and time commitments necessary to succeed in the CS Bachelor's degree.

Because of the need of more trained computer scientists, as well as more diverse computer scientists, we are hoping to find a way to get more students into the major while ensuring a high probability for success. As an example, students who earn an A in CS 2420 are 65% likely to earn an A- or above in CS 3500 (the next required course). Students who earn a B in CS 2420 are 16% likely to earn above an A- or above in CS 3500. Thus while we are sensitive to expanding our enrollments and diversity, we need to find a way to make sure the outcomes of CS 2420 are met (at the excellent level) by a larger and more diverse population of students, giving hope that they will succeed throughout the program.

3A.2 Undergraduate Student Diversity

The following table shows the percentage of undergraduate women, underrepresented minorities (URM) and nonresident alien (NRA) students from enrollment data, from Fall 2009 to Spring 2016. For underrepresented minorities, only the following categories are included: Hispanic/Latino, American Indian or Alaska Native, Black or African American, Native Hawaiian or Other Pacific Islander, and Two or More Races. The data was obtained from the Office of Budget and Institutional Analysis (OBIA), http://www.obia.utah.edu.

	2009 – 2010		2010 – 2011		2011 – 2012		2012 – 2013		2013 – 2014		2014 – 2015		2015 – 2016	
	#	%	#	%	#	%	#	%	#	%	#	%	#	%
females	17	5.2	20	5.7	33	7.8	41	9.4	45	10.3	60	10.8	61	10.8
URM	15	4.6	19	5.2	30	7.1	29	6.7	31	7.1	45	8.1	55	9.7
NRA	14	4.3	15	4.1	22	5.2	29	6.7	23	5.3	35	6.3	36	6.4

Table 3A-1: Percentage of undergraduate women, URM, and NRA

As can be seen from the table, the percentage of women in the undergraduate program has grown from 5.2% to 10.8% during this time period, at the same time that the number of enrolled majors has increased significantly, from 326 to 565. The percentage of underrepresented minorities has similarly grown from 4.6% to 9.7%. The percentage of nonresident alien students has ranged from 4.3% to 6.7%, but has been relatively flat for several years. By comparison, according to the 2015 Computing Research Association (CRA) Taulbee Survey (http://cra.org/resources/taulbee-survey/), the percentage of undergraduate CS degrees awarded was 15.7% for females, 13.4% underrepresented minorities in the same categories, and 8.8% nonresident alien students. Therefore, in spite of encouraging trends towards increasing the diversity of our undergraduate student population over the past seven years, we are below the national average across all categories. Further, unlike other universities that are continuing to see an increase in the percentage of women, ours has plateaued for the last few years.

To calibrate this data, we provide the same numbers for the entire undergraduate student body at University. Among the 15,038 undergraduates in 2016, 44.6% are women, 16.6% are underrepresented minorities in the same groups, and 5.7% are nonresident aliens. Our major has a disproportionately high percentage of nonresident alien students. A partial explanation for lagging behind other institutions in representation of women and underrepresented minorities is related to the demographics of the University, and increases in recruiting diverse students to the University would likely be beneficial to diversity of our program.

To gain a better understanding of the reasons behind the lower percentage of women and underrepresented minorities, we have gathered some additional data looking at the pipeline of students as they progress through the required undergraduate courses. We have also participated in the CRA's Booming Enrollment survey in 2015, which required collecting demographic data for classes. In the Fall 2015 offering of the introductory CS 1410 course, only 17% of the students were women, 25% were minorities (but this number includes people who declined to provide race and ethnicity so is not directly comparable to the previous statistics), and 13% were nonresident alien. At each level we see a decline among these percentages with the Fall 2015 required CS 3500 having 15.5% women, 16% minority and 9% nonresident alien students; and Fall 2015 upper level required CS 4400 having 9% women, 15% minority, and 7% nonresident alien students. Additional analysis has shown that at every course level, the percentage of women and minority students drops. In conclusion, it seems that our classes are not sufficiently diverse from the first course, and then there is a retention problem throughout the program.

New this year, a process to admit entering freshman into the CS program as part of college admissions could potentially increase the diversity of the incoming students. We looked at the gender data for the direct admits from this year's class; we do not have other demographic data. Nine of the 53 students admitted, or 17%, were women. Of the 19 students that accepted their admission offer, only one was female, or 5%. Direct admission may be a path in the future to recruit more diverse students, but it was not effective towards this goal this year.

To better understand our students' experiences, we participated in the 2015 CRA Data Buddies survey of undergraduate CS students, which resulted in a collection of survey results for all participants, and one with just the University of Utah responses. While overall the Utah students' responses compared favorably to those of the entire group, the one area where we were below the norm was in environment issues and mentoring. Perhaps the most troubling statistic is that, when asked who the students consider as a mentor, 46% of the Utah students responded

"No one" (compared to 36% for the entire group). To balance this, their responses were above average when asked about getting support from other students and their families.

Regarding diversity activities in improving recruitment of women and underrepresented minorities, the School runs a summer camp for K-12 students, with significant efforts to recruit female and minority counselors as well as campers (https://www.cs.utah.edu/~dejohnso/GREAT/). Our faculty participates in the College's HiGEAR (https://www.coe.utah.edu/2016/06/17/shifting-into-hi-gear/) summer program for high school girls. Beginning in 2011, we have two new scholarships for women, the *Ariana LaLonde Scholarship for Excellence in Computer Science* and the *Undergrad Excellence Scholarship Program.*

For students already at University, we have made significant effort to recruit and retain women, and to a lesser extent, underrepresented minorities.

- A fundamental curriculum change was the introduction of what is now called CS 1030, a gentle onramp course introducing students to computer science who have no previous background. This course uses a forgiving programming language and is at a slower pace than our traditional CS 1 course.
- Starting in 2009, we worked with National Council on Women in Information Technology (NCWIT), who surveyed our undergraduates and visited our campus to interview students, faculty and advisors. The survey resulted in a report, and the interviewing led to guidelines for a strategic plan.
- Every year since 2011, the School has sent a group of students and a faculty mentor to the Grace Hopper Celebration of Women in Computing. In 2013 and 2015, we sent students to the Tapia Celebration of Diversity in Computing. In 2014 and 2016, we have participated in the Rocky Mountain Celebration of Women in Computing conference, sending students and speakers, and serving on the organizing committee. The students have found these conference trips to be very rewarding, and have provided anecdotal evidence of attendance being valuable for retention.
- Starting in 2011, we began having regular once-a-semester faculty lunches with the undergraduate women students to get their opinions on how to improve our program or the learning environment, and identify ways to support the students in their careers.
- Our faculty participates in the College of Engineering Women in Engineering program as council members, mentors, speakers, and attendees.

3A.3 Not Applicable for Undergraduates

3A.4 Undergraduate Student Support

The School offers tuition waiver and merit-based scholarships to pre- and full-computer science and computer engineering majors who are currently enrolled at the University of Utah. Awards are available for full-time students for fall and spring semesters only. CS/CE scholarship applicants are also eligible for scholarships awarded through the College of Engineering. Some College scholarships require additional essays, but only one application is required for all School and College scholarships. The scholarship application deadline is February 15 each year, for the following Fall and Spring semesters.

To be eligible for most scholarships, students must take at least 12 credit hours per semester and have a high cumulative and/or CS GPA.

Incoming transfer students and entering freshmen can also apply for University Scholarships or College of Engineering Scholarships.

BS/ MS students are eligible to apply for School scholarships, but are only eligible to receive undergraduate scholarship funding while they are in the undergraduate portion of the program.

Departmental Tuition Waiver Scholarships: These awards are available to matriculated (degree-seeking) students majoring in computer science or computer engineering who are residents of the state of Utah. The award covers up to $5,000 of resident tuition for fall and spring semester.

School Scholarships: These awards are available to all computer science and computer engineering majors. They range in value from $500 to $6,000, and are made possible by generous donations from the School faculty, alumni and companies. (See list below).

College of Engineering Scholarships: The College awards several scholarships to the top students in the college. CS students may apply for Collage scholarships simply by filling out the online application for a CS scholarship. The College also has scholarships available for financial need.

The following includes the undergraduate scholarships awarded for 2016-17. Availability and amounts may vary each year:

- Elvin D. Asay Endowed Scholarship
- Igor Best-Devereus Scholarship
- Richard & Brenda Brown Scholarship
- Computer Science Research Experience for Undergraduates
- Curl avery.io # scholarship
- Al Davis and Julian D’Amore Scholarship

- Howard J. & Joan P. de St. Germain Endowed Scholarship
- Disney Scholarship
- Wilford and Dana Druk Scholarship
- EBay Data Center Engineering Scholarship
- EMC Data Center Engineering Scholarship
- Joseph and Phyllis Everton Memorial Scholarship
- David H. Hanscom Undergraduate Scholarship
- Janette and Pierre Haren Scholarship
- Grace Murray Hopper Memorial Scholarship
- IGERT Fellowship
- InsideSales.Com, Inc Scholarship for Women in Computer Science
- Yury Izrailevsky Scholarship
- Robert R. Johnson Innovation Scholarship
- Kessler Family Scholarship
- Arianna M. LaLonde Scholarship for Women of Excellence in Computer Science
- Marvin and Tami Martin Scholarship
- Paul G. & Alison R. Mayfield Undergraduate Scholarship
- Shane V. & Robin S. Robinson Endowed Scholarship
- School Department Fellowship
- School Faculty Scholarship
- School Research Undergraduate Experience (REU)
- School Tuition Award Scholarship
- School Undergraduate Scholarship
- School Women's Scholarship
- Abraham Stephens Scholarship
- Undergrad Excellence in Computing Scholarship
- Kiri Wagstaff AI/ML Scholarship
- Walton Family Scholarship
- James Waters Scholarship
- Women in Computer Science Scholarship
- Kim Worsencrost & Dennis McEvoy Family Undergraduate Scholarship

School Undergraduate Teaching Assistant Positions
In addition to scholarships, the School employs undergraduate students as teaching assistants (TAs). Over the past several years this number has increased from under 20 to over 40 per semester. Pay for top TAs has also been increased to reward continued participation. See Section 3A.6 for further information.

3A.5 Undergraduate Student Advising

The School employees 1.5 staff advisors who handle ~800 prospective and pre-major students, and ~600 FMS students. On average, students are able to schedule an appointment with, and see an advisor within two weeks (mostly within a week except during mandatory advising periods). Our advisors have a total of 12 years of experience. Staff advisors are responsible for a variety of duties associated with FMS advising (e.g., degree path planning, "on-track" checks, career planning, graduation clearance, emotional counseling, etc.), as well as transfer student advising, new student recruitment, permission codes, pre-req checking, scheduling, etc. In general our advisors feel their workload is manageable, though some thought could be put into taking the administrative load (e.g., pre-req checking and permission codes) off of them.

The University requires mandatory undergraduate advising checkpoints during the first semester of freshman year (or the transfer year), the end of the second year, and one semester prior to graduation. Students are encouraged to meet with the academic or faculty advisor once every semester and are sent email announcements throughout the year. Majors, Pre-majors, minors, and prospective students meet with the undergraduate advisor either in-person, over the phone, or through email communication.

Undergraduate pre-engineering and undecided students can meet with Ms. April Vrtis, the Academic Advisor for the College. She helps students navigate the College, explore their interests, and assists students with finding the best ways to achieve their goals.

The School is planning to assign undergraduate students directly to faculty mentors. The idea is that each faculty member will give professional development advice to approximate 10-15 students. Specific degree bureaucracy will still be handled by staff advisors, but we hope that students will be able to get more career advice (e.g., how to get into grad school) from the faculty.

Appeals Process
Students who have concerns with any aspect of the program (e.g., grade disputes, conflicts with professors, desires to waive/change degree requirements, etc.) are encouraged to bring their concerns to the Director of Undergraduate Studies and the School Undergraduate Committee. Students who have appeals denied are always allowed to talk to the next level of the chain of command: Director of the School, Dean of Engineering, University Vice Presidents.

3A.6 Undergraduate Teaching Assistant (TA) Training

The School selects and tracks TA performance via a new online system (created by undergraduates in our Web Software Course, and maintained by faculty). This system allows faculty/instructors to evaluate each TA at both the midpoint, and at the end of the course. For the most part, only TAs who achieve very good or above ratings are retained for the coming year.

Further, the School has moved to a new model where TAs are classified into groups: undergraduate TAs, master's level TAs, and PhD teaching mentorship students. In general, courses taught at the undergraduate level are staffed by undergraduates. Graduate courses are staffed by PhD students being trained via the teaching mentorship (TM) program. When there are not qualified undergraduates or when there is not an appropriate "TM", other graduate students (usually Masters) are hired to cover courses.

TAs are paid at a level corresponding to their experience. First year undergraduates are paid $12 an hour (for usually between 10 and 20 hours a week) and receive a raise each semester up till their fourth, where they jump to $16 an hour. We find this payscale to be below what students make while working (or via internships) but not so far below as to greatly reduce our candidate pool. Furthermore, TAs receive non-tangible benefits, such as recommendations and research offers.

3A.6.1 Teaching introductory Computer Science

The School has been teaching a course to instruct new TAs on proper teaching methods (as well as to train them on various special situations, such as FERPA and appropriate student interactions). Prior to Fall 2015, this course was offered to both graduate and undergraduate students, and took place throughout the early part of the semester.

In Fall 2015, the CS 5040 course was updated to better train and support our undergraduate TAs (with our graduate TAs receiving training from their PhD advisors). The most significant change was to give new TAs the majority of their training *before the semester begins*. Class meetings moved from once a week throughout the semester to a 9-hour workshop (across three days) the week before classes begin and two 2-hour follow-up meetings after the first and second thirds of the semester.

Another notable alteration to CS 5040 was to restrict enrollment to undergraduate student TAs and focus instruction on handling large introductory-level courses. The concerns of teaching such courses are different from those of teaching small-to-medium advanced undergraduate and graduate level courses, which are the typical assignments of graduate student TAs.

The material covered in CS 5040 was brought up-to-date and expanded somewhat. Topics include: classroom demeanor, communication and presentation skills, time management, helping students on a variety of platforms (in lab sections, in TA help hours, in email, via forum posts), being the TA of near peers, effective grading, observing other TAs, and staying motivated.

Related to the improvements in CS 5040, the School has increased support for a community among the undergraduate student TAs in the following ways:

- Each semester, the School sponsors a party to show appreciation for the undergraduate TAs.

- Each year, undergraduate TAs receive special apparel (t-shirts, jackets, etc.) from the School, which fosters unity among the TAs and also advertises the opportunity for TA experience to other students.
- Each year that a student continues as a TA, he/she receives a $1/hour raise.
- Beginning Fall 2016, a lab in MEB will be designated for use by the undergraduate TAs to further encourage collaboration among TAs.

This effort has been especially important as the number of undergraduate student TAs continues to grow (we currently employ over 40 undergraduate students as teaching assistants).

3B. Graduate Students

3B.1 Graduate Student Recruitment

The School recruits graduate students both domestically and internationally. The department's domestic recruiting efforts comprise of the following practices:

- Recruitment from the department's own undergraduate population:

  The department has an active and growing REU (Research Experience for Undergraduates) programs. Many faculty members actively engage senior undergraduate students in their research. At their senior years, undergraduate students have the option of doing an undergraduate thesis, which helps them develop interest in research and also motivates them to apply for graduate schools (either here at Utah or elsewhere).

  The School helps undergraduate students apply for various fellowships and grants to support their graduate studies. In particular, a seminar has been developed specifically for guiding students to apply for NSF's Graduate Fellowship Program, and it has proven quite effective; for example, two undergraduate students were awarded the highly competitive NSF Graduate Fellowship in the 2015-2016 season.

- Recruitment from (geographically) nearby schools.

  The School has developed ties and relationships with colleges and universities from the Intermountain-West region, which allows the School to attract more applicants to its graduate program from these student populations. For example, the School has participated in the annual UCUR conference (Utah Conference on Undergraduate Research). The UCUR conference is an annual event organized by different universities from the State of Utah, that is open to all undergraduate students from the State to report their research activities. It attracts hundreds of student participants each year, and is a great venue to promote the School's graduate program. The School has extended its outreach to nearby states such as Idaho, Nevada, Colorado, etc. by sending flyers and giving talks at schools from those states.

- Recruitment from domestic students nationally.

  The School also engages in various recruitment efforts from the pool of domestic undergraduate student population nation-wide. This is often carried out by asking faculty members to show a brief overview of the School when they visit and give talks at different universities. Individual faculty's reputation and the impact made by his/her research program also play a key role in attracting more applications from domestic students nation-wide. Having said that, this is an area the School needs the invest continuous effort and improvement in order to increase the domestic graduate student population in its graduate program, at both the MS and the PhD levels.

- Recruitment from domestic students internationally.

  A large portion of the School's graduate students is international students. This is the case for both the MS and PhD programs. Many of the School's faculty members come from an international background and they often still maintain a strong connection with their home countries' top institutions, which helps the School recruit talented students from those countries. Faculty members are asked to promote the School's graduate program whenever they visit a foreign institution. This is often carried out by asking faculty members to show a brief overview of the School when they visit and give talks at different universities. Individual faculty's reputation and impacts made by his/her research program also play a key role in attracting more applications from domestic students nation-wide. Having said that, this is an area the School needs continuous efforts and improvement in order to increase the domestic graduate student population in its graduate program, at both the MS and the PhD levels

For admitted PhD students, the department organizes a grad visit day – typically in late Febuary or early March – that brings these students to Salt Lake City and enables them to interact with faculty members and visit the department in person before making their final decisions. This is a very effective tool in convincing top quality students to come to Utah when they are faced with multiple offers.

Figure 3B-1 shows the number of applications to the School's graduate program at the MS level in the last five years, and a clear upward trend is observed for the master's program. The number of applications to the School's graduate program at the PhD level in the last five years, as shown in Figure 3B-2, however, shows little variation.

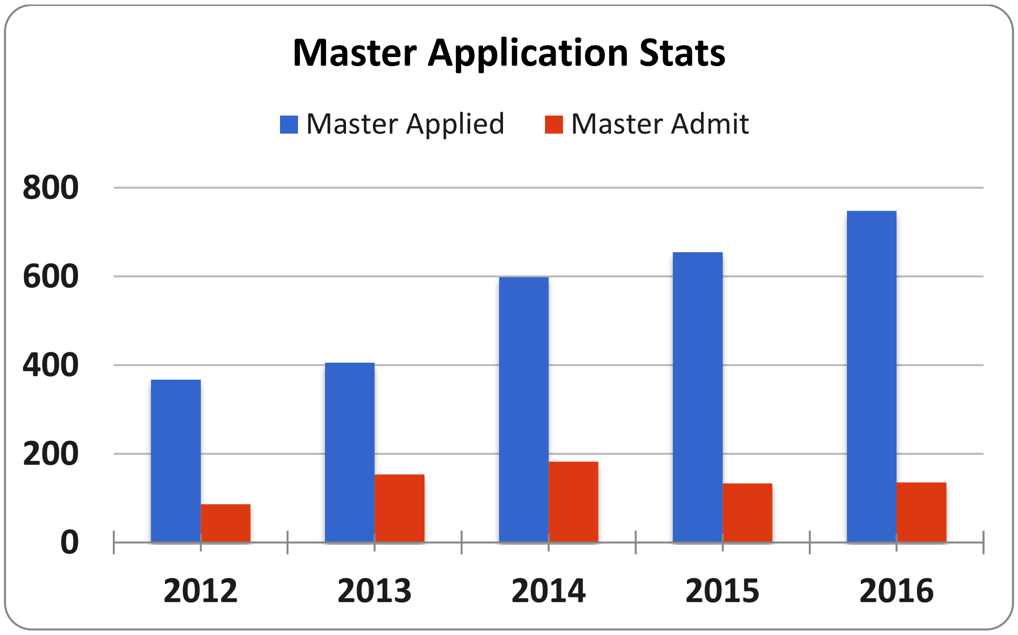


Figure 3B-1: Application and admission numbers for the MS program

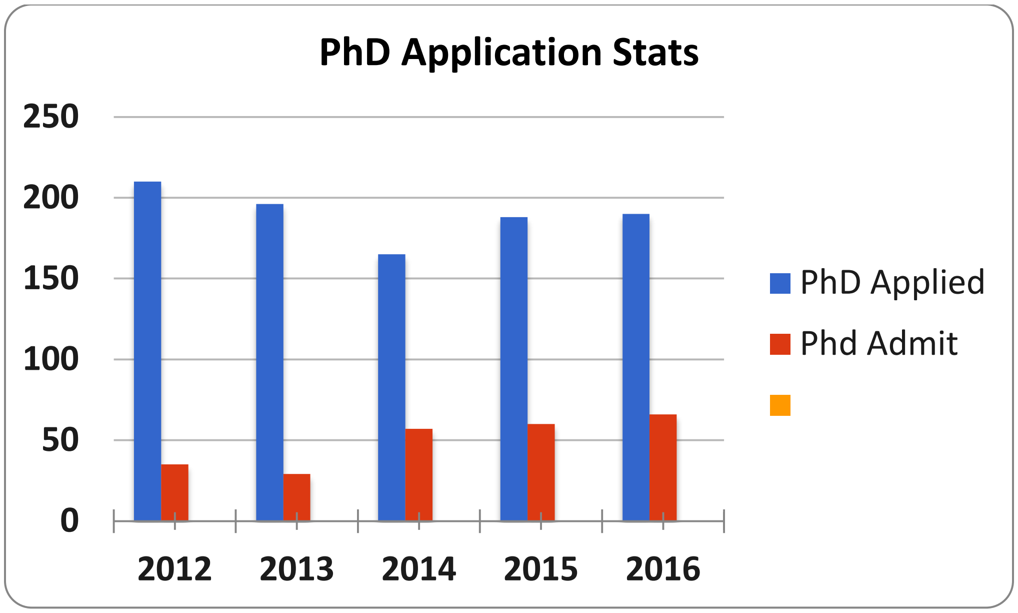


Figure 3B-2: Application and admission numbers for the PhD program

3B.2 Graduate Student Diversity

The table below shows the demographic composition of our graduate students, separated into MS and PhD, using the same categories as used for the undergraduates. From 2009 to present, the percentage of women in the MS program has increased from 8.6% to 28.3%, while the numbers of master’s students have ranged from 93 in 2010 to a peak of 147 in 2015. In the PhD program, the percentage of women has grown from 13.5% to 22.4%, while the number of PhD students has grown from 111 to 125. Nationwide, according to the 2015 Taulbee report, the percentage of MS and PhD degrees awarded to women was 25% and 18%, respectively. Therefore, our enrollment of women has grown significantly and exceeds the national percentage of degrees awarded to women. Regarding underrepresented minorities, we have seen some growth in our master’s program but almost no growth in the PhD program, and are usually below the national average of 4% for both degrees. Our nonresident alien national students comprise 74.3% and 55.2% of our MS and PhD programs, as compared to 68% and 60.7%, respectively, for the national average. Therefore, the main diversity issue in our graduate student pipeline is the lack of underrepresented minorities. For the university as a whole: 10.5% of master’s and 8.3% of PhD students are underrepresented minorities in the same groups.

	2009-2010		2010-2011		2011-2012		2012-2013		2013-2014		2014-15		2015-16	
	#	%	#	%	#	%	#	%	#	%	#	%	#	%
Females MS	8	8.6	9	8.1	14	11.9	19	16.5	28	19.7	32	21.8	32	28.3
Females PhD	15	13.5	17	14.0	18	14.8	17	14.8	27	22.7	26	22.4	28	22.4
URM MS	2	2.2	1	0.9	3	2.5	6	5.2	8	5.6	4	2.7	2	1.8
URM PhD	0	0.0	0	0.0	0	0.0	1	0.9	0	0.0	0	0.0	1	0.8
NRA MS	49	52.7	67	60.4	62	52.5	68	59.1	101	71.1	116	78.9	84	74.3
NRA PhD	70	63.1	68	56.2	70	57.4	68	59.1	66	55.5	65	56.0	69	55.2

Table 3B-1: % and count of females, URM and NRA by program

- There is an annual dinner for all female graduate students early in the Fall semester. Over the past two years, we have had additional dinners with just PhD students once or twice a year. The students find these dinners beneficial as they help establish a community among the students, and also provide an opportunity to confer with female faculty and raise issues of concern.
- The School and individual faculty send a few graduate students to the Grace Hopper Conference, along with the undergraduate group.
- CRA-W Grad Cohort has provided an opportunity for a few of our PhD students to meet other PhD students across the country and focused mentoring.

3B.3 Graduate Student Admissions

The quality of the School's graduate students has witnessed a steady increase in recent years. This is clearly reflected by Table 3B-2 which shows that the acceptance rate to the master 's program has been reduced from 23.6% in Year 2012 to 18.1% in Year 2016. A larger pool of applications also means that the School enjoys more flexibility to be more selective (even if the acceptance rate were to stay the same). This observation is reinforced by the numbers shown in Table 3B-2, which indicate that the average GPA of students being admitted has seen a clear increase in recent years.

With regard to the PhD, even though there is no significant change to the acceptance rate (and some earlier years even exhibit a lower acceptance rate), the quality of those students who had applied and eventually got accepted into the School's PhD program has increased. On contrary to the analysis used for MS students, it is difficult to quantify the quality of PhD students (through simple measures such as GPA value), however, across the boards, faculty members have reported that more and more PhD students are from high quality undergraduate programs (e.g., top universities in their home countries for international students as well as more domestic out-of-state students) and many new PhD students come with extensive undergraduate research experiences as well.

An indication of this is that the School's research expenditure is on a steady upward trajectory (see the analysis in Section 3B.4), showing a significant increase of more research outputs and higher research activities, which of course is due to many factors, such as the increase of faculty body; but better graduate student quality, especially at the PhD level, is definitely a contributing factor.

Academic Year	Number of Students who Applied to Graduate Programs	Average Overall GPA	Average Overall GRE Score	Number of Graduate Students Admitted
2015-16	828	3.68	625/157	77
2014-15	735	3.55	672/157	84
2013-14	558	3.52	655/157	105
2012-13	566	3.60	627/156	70
2011-12	578	3.76	633	79
2010-11	554	3.44	623	68
2009-10	517	3.40	622	50

Table 3B-2: Graduate admissions data

Lastly, the department has developed an admission system that allows interactive reviewing and commenting for each application. The system is very user-friendly and has increased the productivity of the School's admission process significantly.

3B.4 Graduate Student Support

The School is committed to providing financial support to its graduate students. Almost all of the School's PhD students are supported through graduate fellowships, research assistantships, and teaching assistantships. A small portion of the School's MS students also receive funding support in the form of research assistantships and teaching assistantships. As long as a student is supported at 0.5 FTE rate (20 hours per week), he/she enjoys full tuition benefit, and the student only needs to pay what is called differential tuition which is only $700-800 per semester for nine credit hours. The student also receives significant subsidy to purchase a health insurance coverage (up to 80% is covered).

Graduate research assistantship and fellowship provide a monthly stipend of $2,300, whereas a graduate teaching assistantship provides a monthly stipend of $1,900. As research productivity and research activities continue to rise, the annual research expenditure for the School has seen a steady increase in the last few years, as shown in Figure 3B-3. This means that the School is able to support more graduate students on research assistantships.

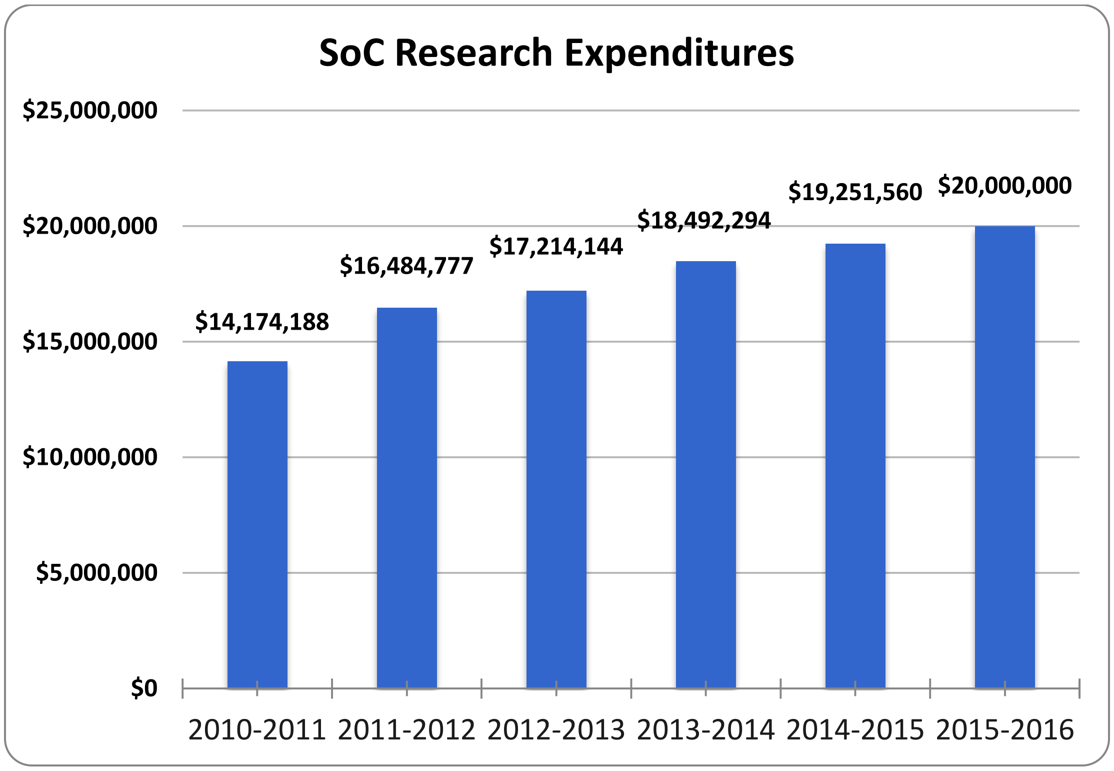


Figure 3B-3: Annual research expenditure

Furthermore, the increasing enrollment of the computer science undergraduate program also brings more SCH (student credit hour) and return overhead dollars to the department. This allows the department to recruit and support more students under teaching assistantships.

The department also implemented a new funding model for first year PhD student since 2014. Most of new PhD students are supported by the department through graduate fellowships, which allows them to explore different research directions and research programs within the department, without the obligation to work with any one particular faculty in their first year (in contrast to a student who is supported by a research assistantship), or to work as a TA. In return, a fellowship student is required to serve as a "teaching mentor" (see details below in Section 3B.6) in his/her second or third year, when he/she is supported by a faculty member using the faculty's research dollars. This enables the fellowship funding model to be self-sustained.

As shown in Figure 3B-4, most of new incoming PhD students are now supported by the department's fellowships. Almost all fellowship students are converted to a RA by the start of the second year of their PhD program.

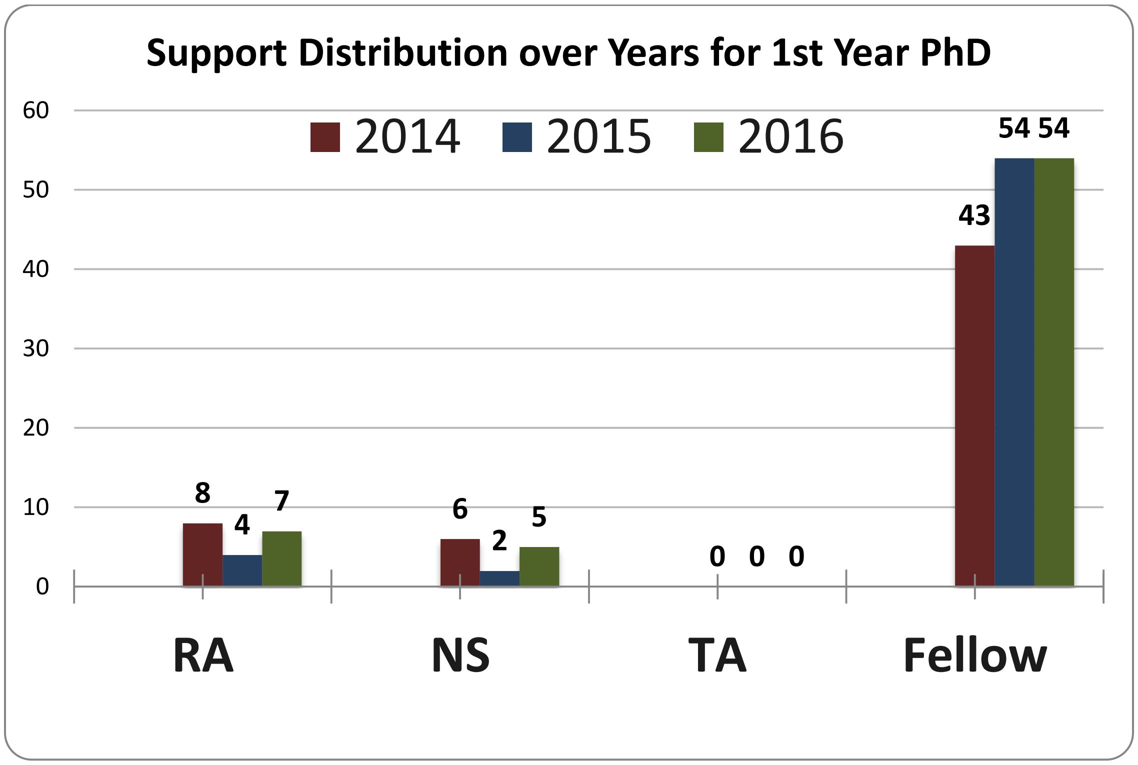


Figure 3B-4: Types of support for 1st year PhD students since the implementation of the department fellowship model (NS stand for No Support)

In contrast, the School offers few funding supports for new incoming MS students as seen in Figure 3B-5. However, a good portion of them are able to secure funding supports in their second or third semesters in the program, through research assistantships (many are from other departments and schools from the University) and teaching assistantships.

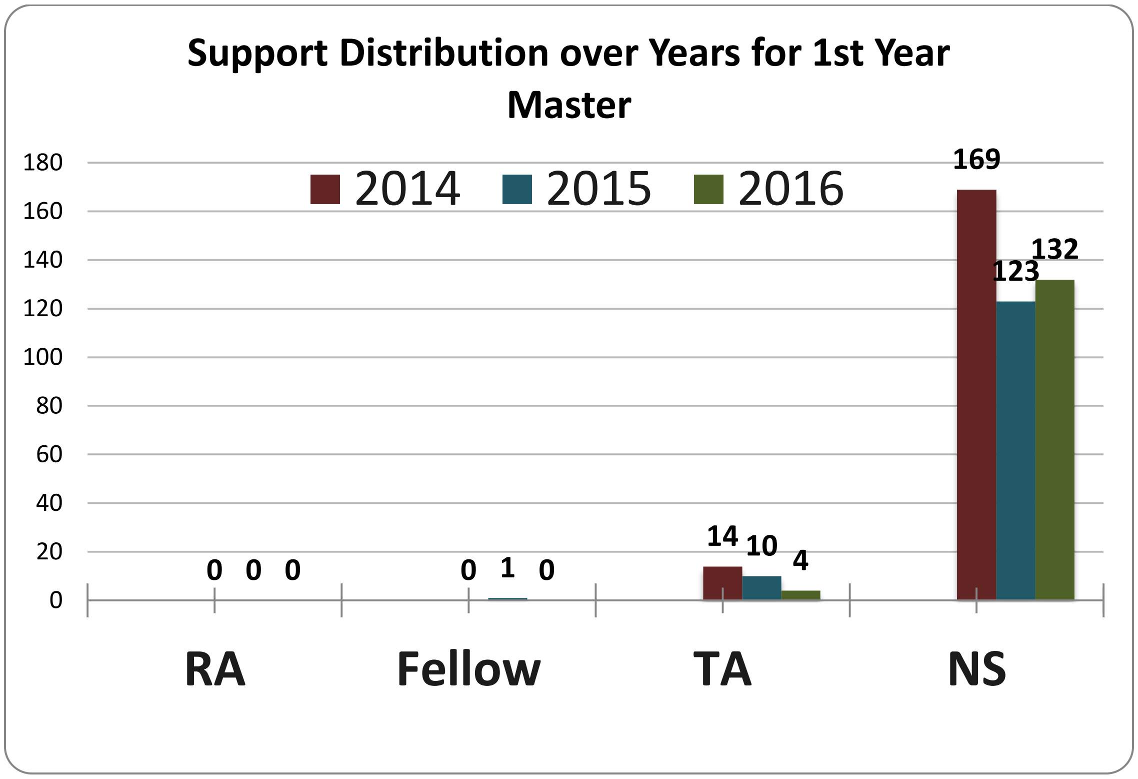


Figure 3B-5: Types of support for 1st year master's students since the implementation of the department fellowship model

PhD students also apply for and receive support through University fellowship programs. On average 1-2 students will receive a University fellowship from a given year, which offers full support that is roughly equivalent to a research assistantship for a year.

The School also strongly encourages PhD students to apply for various graduate student fellowships from external sources, such as the NSF graduate fellowship, and fellowship awards from industry that are offered by various companies; see the list below that summarizes the recipients in recent years.

Student Name	Years Received Award	External Support	Advisor
Vakkalanka, Sarvani	2009-2010	Graduate Fellowship	Gopalakrishnan, Ganesh
Li, Peng	2012	NVIDIA Fellowship $25k	Gopalakrishnan, Ganesh
Chiang, Wei-Fan	2013	NVIDIA Fellowship $25k	Gopalakrishnan/Rakamaric
Erickson, Phillip	2014-2016	ARCS Scholarship $15k	Henderson, Thomas
McCurdy, Nina	2015	NSF GRF $100k	Meyer, Miriah
Sun, Weibin	2011-2012	NVIDIA Fellowship $25k	Ricci, Robert
Swan, Hannah	2015	NSF Graduate Fellowship $138k	Yuksel, Cem

Table 3B-3: PhD students who have received external support

Students Name	Year Completed	Degree	University	Title(s)	Advisor
Yu Sun	2007	Ph.D.	University South Florida	Associate Professor	John Hollerbach
Eric Eide	2009	Ph.D.	University of Utah	Research Assistant Professor	Matthew Flatt
Geoffrey Draper	2009	Ph.D.	Brigham Young University Hawaii	Associate Professor	Rich Riesenfeld
Mark Van Langeveld	2009	Ph.D.	University of Utah	Associate(EAE) Assistant(SoC) Professor	Robert Kessler
Brian Rague	2010	Ph.D.	Weber State University	Department Chair - Professor	Joe Zachary
Cecily Heiner	2010	Ph.D.	Southern Utah University	Assistant Professor	Joe Zachary
Kevin Tew	2012	Ph.D.	Brigham Young University	Assistant Professor	Matthew Flatt
Subodh Sharma	2012	Ph.D.	Indian Institute Of Technology Delhi	Assistant Professor	Ganesh Gopalakrishnan
Jared Plumb	2013	MS	Weber State	Adjunct Professor	Robert Kessler
Ruihong Huang	2014	Ph.D.	Texas A&M University	Assistant Professor	Ellen Riloff
Varun Shankar	2014	Ph.D.	University of Utah Department of Math	Assistant Professor/Lecturer position	Mike Kirby
Daniel Kopta	2015	Ph.D.	University of Utah	Assistant Professor	Cem Yuksel
Paul Rosen	2015	Ph.D.	University of South Florida	Assistant Professor	Chris Johnson
Linda DuHadway	2016	Ph.D.	Weber State University	Assistant Professor	Thomas C. Henderson

Table 3B-4: Students who have moved into academia after graduation

Lastly, at the undergraduate level, the department offers funding support through REU grants and teaching assistantships. Undergraduate students can also apply for various scholarships that are made available through the College of Engineering. There are many different types of scholarships that come with different application and selection criteria, as well as different level of funding support. Please refer to the following website https://www.coe.utah.edu/scholarships for a complete list of available scholarships from the College of Engineering to the undergraduate students in the department.

3B.5 Graduate Student Advising

The department has two graduate student advisors who are responsible for answering questions from prospective graduate students. They are well trained with various policies and requirements in the department's graduate program. They are both experienced with advising graduate students, and by working closely with the director of graduate studies (DGS) and various faculty members, they are able to provide timely and accurate response to address any questions that prospective graduate students may have.

The graduate student advisors are also responsible for advising graduate students that are currently enrolled in the program. They have access to individual student's record and process various forms that students need to complete throughout the program. They also remind students about upcoming milestone deadlines that students need to meet. The graduate student advisors work closely with the DGS to constantly review student profiles and suggest ways to improve the operations of the graduate program.

The School maintains a comprehensive handbook called the Graduate Handbook that details the degree requirements and timelines for different degree options, for both MS and PhD studies. The grad handbook is reviewed and updated annually (during the summer), and the latest version is released at the beginning of each Fall semester. Students can always refer to the latest Handbook online at http://www.cs.utah.edu/docs/Graduate/gradhandbook_2016-17.pdf

Note that a student is **not** subject to new changes and requirements that are introduced to the graduate handbook after he/she has entered into the program. In other words, a student is only binding to the version of the graduate handbook in the Fall semester when he/she enters into the program. The department does keep all previous versions of graduate handbook in an archive, so that students may refer to the correct version that applies to him or her degree requirements.

All MS students need to complete a program of study form which list the courses a student has taken and the grades received for each course, as well as the summary for the project or thesis the student is working on if he/she is in the MS project-option or MS thesis-option. The program of study form lists the committee members for an MS student (the department requires at least three CS faculty members for an MS committee; this is required even for master's students who are in the master's by course-only option). Students need to present the program of study form to their committee members and obtain approvals from each member, which offer an opportunity for committee members to review the progress of an individual student and provide feedback and comments to the student. The DGS acts as the last gatekeeper and does the final sign-off for a student's program of study form.

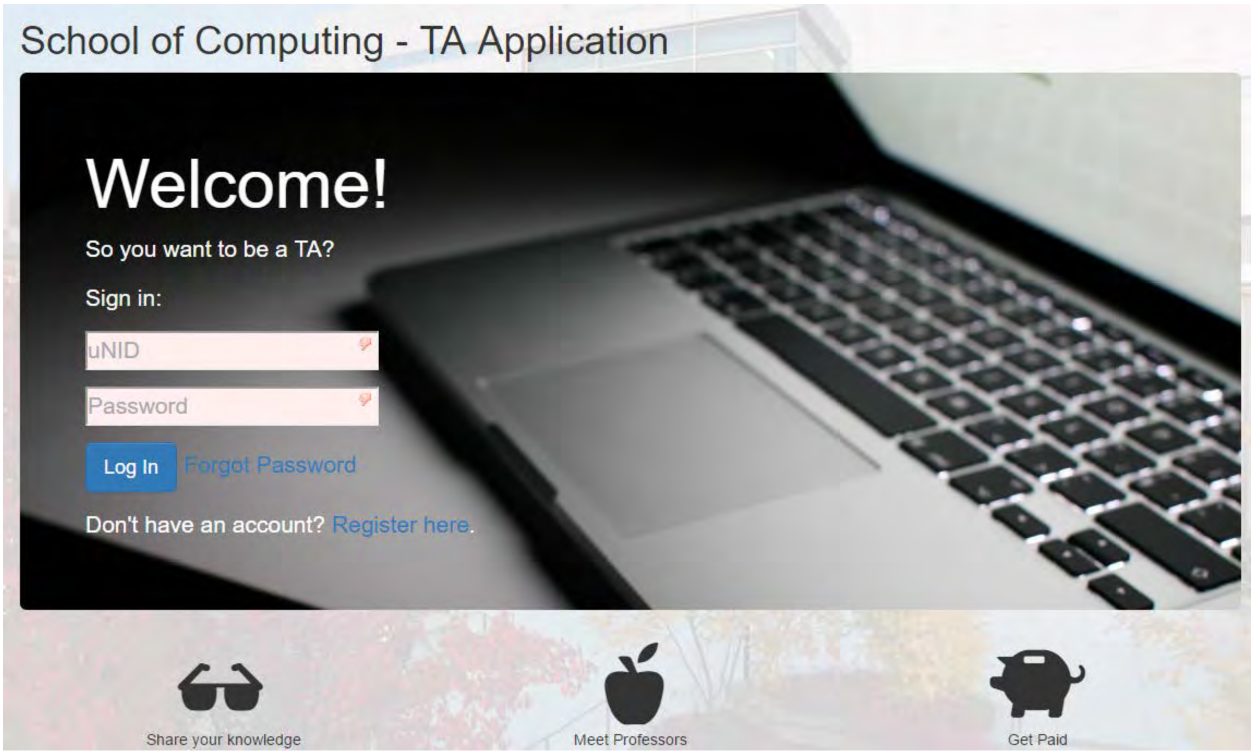
All PhD students are subject to a more stringent advising process. In particular, every Fall semester each PhD student needs to fill out a due progress form which has laid out specific milestones that a PhD student needs to go through in the program, and the expected timeframe of completion for each milestone. These due progress forms are then reviewed by the student's advisor, the graduate student advisors, and finally the DGS to identify excellent, acceptable, and questionable cases (based on the progress made by the student with respect to the target timeframe for each milestone). A faculty meeting is schedule at the end of the Fall semester in which the DGS will report the review results of all due progress forms and discuss with the faculty to form plans for different students, especially those whose progress is questionable.

For handling student appeals we rely on the University's Policy 6-400: Student Code (http://regulations.utah.edu/academics/6-400.php). This may be found in Appendix E.

Lastly, the graduate handbook details the policy regarding leave, change of degree, and other related matters that may expect in the course of a student's graduate study.

3B.6 Graduate teaching assistant (TA) training

The School has developed its own TA application and evaluation system, which is available online at https://ta.cs.utah.edu; see a screenshot of the system in Figure 3B-6. This system enables faculty to easily keep track of TA applications and provide TA evaluations and feedback during and after a semester. These data provide useful insights in designing and developing plans to train the graduate students to be more effective TAs and become a more effective teacher.


School of Computing - TA Application

Welcome!

So you want to be a TA?

Sign in:

Don't have an account? Register here.

Figure 3B-6: Teaching assistant application page

The department organizes a TA orientation at the beginning of each semester. Professor James de St. Germain leads this effort. In addition, the College provides a TA training session to which the department always sends the TAs.

Lastly, since 2014, all PhD students are required to fulfill a teaching mentorship (TM) as a part of their degree requirement. A TM serves similar role to that a TA plays, but often involves more in-depth teaching assignments, such as leading a lecture, designing part of the course, rather than just holding office hours and grading. The funding for TM is provided by a student's advisor, and typically a TM will work with his/her advisor as a teaching assistant for the advisor's course. This means that a TM is assigned with course that falls in the area of his/her research expertise, thus, a TM will be effective in engaging in those more in-depth teaching assignments, and gain valuable teaching skills in the process.

Lastly, all international TAs are required to take a communication skill test organized by the University. If an international TA didn't get a passing grade in that test, he/she must register for a language and communication skill course offered by the English department and the Communication School, in order to serve as a TA.

4A. Undergraduate Curriculum and Programs of Study

4A.1 Undergraduate Degree and Certificate Requirements

The School offers a Bachelor of Science degree in Computer Science, a joint BS degree (with the ECE Department) in Computer Engineering, and an undergraduate minor in CS. In addition to the standard BS in CS degree, students may earn an emphasis in Entertainment Arts & Engineering; Regent's approval, August 27, 2010.

4A.1.1 BS Degree requirements

The Bachelor of Science in CS track is a software-oriented degree that includes 19 CS courses, including required core, theory, and elective courses. A student must be admitted as a CS major by the School in order to take upper-division courses and pursue the CS degree. CS pre-majors take five beginning math and CS courses before applying to the major (i.e., CS 1030, CS 1410, CS 2420, Calc I, Calc II).

4A.1.1.a BS in Computer Science

This program begins with a set of three courses that give students a significant, in-depth exposure to computing topics while exposing them to the breadth of issues that arise in CS, followed by a background in object-oriented programming, and then an introduction to algorithms and data structures. Students then take seven core courses in discrete mathematics, software engineering, computer organization, more advanced algorithms and data structures, software systems, and theory. They build on this background by choosing seven electives from the breadth of the School's course offerings (which includes advanced courses in theoretical CS, scientific computing, artificial intelligence, databases, operating systems, computer networks, programming languages, graphics, computer architecture, and digital design). Each student's undergraduate program is capped with a *senior capstone project*. Along with an in-depth study of computing, the curriculum encompasses a general education in mathematics, science, and the humanities.

If desired, students may take courses options from within a set of *tracks*, to complete the seven required electives (see Appendix F). Students may apply to receive a certificate of completion upon graduation. Track areas include: (1) Software Development, (2) Web & Mobile Development, (3) Computer Systems, (4) Programming Languages, (5) Robotics, (6) Artificial Intelligence, (7) Information, (8) Theory, (9) Visual Computing, (10) Computer Organization, (11) Embedded Systems, and (12) CAD for Digital Systems.

See Appendix G: BS requirements sheet:
http://www.cs.utah.edu/docs/Undergraduate/CSMajor_2016-17.pdf

4A.1.1.b BS in CS, EAE Emphasis

The *EAE Emphasis* includes the same pre-major, major, math and theory requirements as the standard CS degree. Instead of choosing seven electives as students do with the standard degree, the EAE students are given five required electives, and choose two on their own. The senior project also differs on the EAE track. It is offered as a distinct class listing; with a larger multi-disciplinary teams (both engineers and artists) working to produce a video game.

This EAE Emphasis has a companion emphasis offered through the College of Fine Arts (for Film Majors). A key characteristic of the EAE emphasis is its interdisciplinary nature. As the digital entertainment industry continues to grow, employers are focusing more on students who understand both sides of the industry, whether it is CS students with additional fine arts skills or fine arts students with computing skills. The EAE specialization offers cutting edge courses designed for undergraduate students interested in pursuing careers in the digital entertainment industry, and expressing themselves using digital media, including courses covering video game design and development, 3D animation, and computer-generated special effects.

The undergraduate EAE program has been recognized by the Princeton Review as the #1 program for video game development in 2016 and 2013, and #2 in 2015 and 2014.

See Appendix H: BS-EAE requirements sheet:
http://www.cs.utah.edu/docs/Undergraduate/EAE_2016-17.pdf

Semester	Total Graduates	Female	EAE	Honors
(Fall 08)	15	0	N/A	N/A
(Spring 09)	36	3	N/A	N/A
(Sum 09)	10	0	N/A	N/A
2008-09	61	3	-	-
Semester	Total Graduates	Female	EAE	Honors
(Fall 09)	19	1	N/A	N/A
(Spring 10)	42	3	N/A	N/A
(Sum 10)	10	1	N/A	N/A
2009-10	71	5	-	-
Semester	Total Graduates	Female	EAE	Honors
(Fall 10)	20	0	0	0
(Spring 11)	43	2	4	0
(Sum 11)	13	0	2	1
2010-11	76	2	6	1
Semester	Total Graduates	Female	EAE	Honors
(Fall 11)	24	1	5	1
(Spring 12)	41	1	6	0
(Sum 12)	12	1	2	0
2011-12	77	3	13	1
Semester	Total Graduates	Female	EAE	Honors
(Fall 12)	26	2	1	0
(Spring 13)	52	2	7	0
(Sum 13)	9	0	1	0
2012-13	87	4	9	0
Semester	Total Graduates	Female	EAE	Honors
(Fall 13)	15	1	0	0
(Spring 14)	70	10	14	1
(Sum 14)	7	0	1	0
2013-14	92	1	1	15 1
Semester	Total Graduates	Female	EAE	Honors
(Fall 14)	15	1	4	0
(Spring 15)	79	8	21	0
(Sum 15)	7	1	3	0
2014-15	101	10	28	0
Semester	Total Graduates	Female	EAE	Honors
(Fall 15)	10	1	4	0
(Spring 16)	81	10	14	4
(Sum 16)	5	2	0	1
2015-16	96	13	18	5

Table 4A-1: Bachelors of Science CS graduation numbers. Female, EAE and Honors numbers are not included in the Total Graduates column.

4A.1.2 BS in Computer Engineering

Computer Engineering includes the design, implementation, and programming of digital computers and computer-controlled electronic systems. The School and the Department of Electrical jointly offer a Bachelor of Science degree in CE. The CE curriculum provides students with a sufficient background in mathematics, CS, and engineering sciences to analyze and design complex software and hardware systems. The CE program is managed by a committee of 2-3 faculty from each of the departments (the School and ECE).

See Appendix I: BS CE requirements sheet
http://www.ce.utah.edu/files/2015/09/CE-4-yr-Sample-Degree-15-16.pdf

Semester	CE grads
2010-11	21
2011-12	14
2012-13	16
Semester	CE grads
(Fall 13)	5
(Spring 14)	15
(Sum 14)	2
2013-14	22
Semester	CE grads
(Fall 14)	6
(Spring 15)	8
(Sum 15)	3
2014-15	17
Semester	CE grads
(Fall 15)	8
(Spring 16)	15
(Sum 16)	0
2015-16	23

Table 4A-2: BS in CE graduation

4A.1.3 BS minor in CS

The School offers a minor for students who desire to gain sufficient background to use and program computers in another field. In order to be admitted as a CS minor, a student must have a declared major in another department and be making progress in that major.

Semester	Minors
(Fall 09)	2
(Spring 10)	3
(Sum 10)	2
2009-10	7
Semester	Minors
(Fall 10)	4
(Spring 11)	7
(Sum 11)	3
2010-11	14
Semester	Minors
(Fall 11)	5
(Spring 12)	3
(Sum 12)	1
2011-12	9
Semester	Minors
(Fall 12)	2
(Spring 13)	7
(Sum 13)	1
2012-13	10
Semester	Minors
(Fall 13)	0
(Spring 14)	12
(Sum 14)	0
2013-14	12
Semester	Minors
(Fall 14)	4
(Spring 15)	12
(Sum 15)	0
2014-15	16
Semester	Minors
(Fall 15)	4
(Spring 16)	13
(Sum 16)	2
2015-16	19

Table 4A-3: BS in CS Minor graduation data

See Appendix J: Minor requirements sheet:
http://www.cs.utah.edu/docs/Undergraduate/CSMinor_2016-17.pdf

4A.1.4 BS/MS program

The combined Bachelor of Science/Master of Science degree in CS allows students to earn a BS and MS in approximately five academic years. The BS/MS can combine a BS in either CS or CE with an MS in either CS or computing. Undergraduates begin taking graduate-level courses during the senior year in order to complete the master's degree in just one additional year.

Semester	BS/MS
(Fall 12)	0
(Spring 13)	7
(Sum 13)	0
2012-13	7
Semester	BS/MS
(Fall 13)	2
(Spring 14)	6
(Sum 14)	1
2013-14	9
Semester	BS/MS
(Fall 14)	2
(Spring 15)	3
(Sum 15)	0
2014-15	5
Semester	BS/MS
(Fall 15)	2
(Spring 16)	5
(Sum 16)	0
2015-16	7

Table 4A-4: BS/MS program graduation data

4A.1.5 Online degree/certificate offerings

The School does not offer any online degree or certificate options.

4A.2 Undergraduate CS Courses Offered

List all the courses offered in the program, including online courses.
Standard Undergraduate Offerings (See: http://catalog.utah.edu, search courses → CS)

- CS 1000 - Engineering Computing
- CS 1001 - Engineering Computing using MATLAB
- CS 1030 - Foundations of CS
- CS 1040 - Creating Interactive Web Content
- CS 1060 - Explorations in CS
- CS 1410 - Introduction to Object-Oriented Programming

- CS 1960 - Special Topics
- CS 2050 - Making Noise: Sound Art and Digital Media
- CS 2100 - Discrete Structures
- CS 2420 - Introduction to Algorithms & Data Structures
- CS 2950 - Independent Study
- CS 2960 - Apple Certification
- CS 2963 - Linux Professional Institute Level 1
- CS 2965 - Special Topics
- CS 2966 - Special Topics
- CS 3011 - Industry Forum
- CS 3020 - Research Forum
- CS 3100 - Models of Computation
- CS 3130 - Engineering Probability and Statistics
- CS 3200 - Introduction to Scientific Computing
- CS 3470 - Scripting Language Design and Implementation
- CS 3500 - Software Practice
- CS 3505 - Software Practice II
- CS 3700 - Fundamentals of Digital System Design
- CS 3710 - Computer Design Laboratory
- CS 3810 - Computer Organization
- CS 3960 - Special Topics
- CS 3991 - CE Junior Seminar
- CS 3992 - CE Pre-Thesis/Pre-Clinic/Pre-Project
- CS 4000 - Senior Capstone Project - Design Phase
- CS 4010 - CS Internship
- CS 4150 - Algorithms
- CS 4190 - Programming Challenges
- CS 4230 - Parallel Programming
- CS 4300 - Artificial Intelligence
- CS 4400 - Computer Systems
- CS 4480 - Computer Networks
- CS 4500 - Senior Capstone Project
- CS 4530 - Mobile Application Programming
- CS 4540 - Web Software Architecture
- CS 4600 - Introduction to Computer Graphics
- CS 5100 - Theory of Computation
- CS 5130 - Computational Statistics
- CS 5150 - Advanced Algorithms
- CS 5310 - Robotics
- CS 5320 - Computer Vision
- CS 5340 - Natural Language Processing
- CS 5350 - Machine Learning
- CS 5460 - Operating Systems
- CS 5470 - Compiler Principles and Techniques
- CS 5510 - Programming Language Concepts

- CS 5530 - Database Systems
- CS 5540 - Human/Computer Interaction
- CS 5610 - Interactive Computer Graphics
- CS 5630 - Visualization
- CS 5650 - Visual Perception from a Computer Graphics and Visualization Perspective
- CS 5710 - Digital VLSI Design
- CS 5720 - Fundamentals of Analog Integrated Circuit Design
- CS 5740 - Computer-Aided Design of Digital Circuits
- CS 5745 - Testing and Verification of Digital Circuits
- CS 5750 - Synthesis and Verification of Asynchronous VLSI Systems
- CS 5780 - Embedded System Design
- CS 5785 - Advanced Embedded Software
- CS 5789 - Embedded Systems and Kinetic Art
- CS 5830 - VLSI Architecture

Special Topic Courses

Any faculty member from the School, with permission from the director, can teach a special topics course on the subject of their own choosing. Once such a course is taught more than twice, it generally goes before the School Curriculum/Undergraduate (or Graduate) committee to be assigned a permanent number. The Undergraduate Committee maintains quality control by looking at proposed syllabi, enrollment numbers and course evaluations, as well as how the course fits in the overall picture of the CS degree and other offerings.

4A.3 Undergraduate Programs of Study

Both CS and CS-EAE Bachelor of Science degrees can be completed in four years. Students are encouraged to meet with the academic advisor early for course planning. Summer semesters are suggested, but not required, to ease the fall and spring semester schedule.

Four-year plans are provided to students in the New Student Orientation packet, advising office and online. The University also provides our suggested plans in the Graduation Planning System (GPS) and University general catalog pages.

See Appendix K: CS four-year plan
See Appendix L: CS-EAE four-year plan

4A.4 Undergraduate Professional Development

4A.4.1 Standardized professional development across entire curriculum

Most CS bachelor's degree awardees move forward to a career in the private sector. These students receive professional skills and ethics training through a variety of methods. First, most of these students will work as programmers/software developers, and thus have a large set of courses devoted to basic/proper programming techniques:

- CS 1410 - Object Oriented Programming (Primarily Java)
- CS 2420 - Data Structures and Algorithms (Primarily Java)
- CS 3500 - Software Practice I (Primarily C#)
- CS 3505 - Software Practice II (Primarily C++)

Students also receive a strong software engineering background including exposure to Agile methodologies, versioning, testing, team development, etc., in the Software Practice Sequence (CS 3500, CS 3505). These courses also highlight ethical and professional values aspired to by software designers (e.g., The Pledge of the Computing Professional).

As students progress through the program, they have various optional courses that directly translate into professional opportunities, including: CS 4010 - Internship, CS 4530 - Mobile Development, CS 4540 - Web Software Architecture, and CS 5530 - Database Systems.

In their senior year, students work on software teams to build a Capstone Project. During this project, the students practice (as far as we are able to simulate in academia) the entire design, implement, test, deliver sequence found in industry. The project teams are usually four members in size and work across two semesters to produce a fully functional software system. These courses also stress written and oral communication skills, with presentations and a final public demonstration of the project. Further, the teams create multiple written documents as well as a final poster for use during the public demonstration day.

4A.4.2 Additional professional development opportunities

A significant minority of our students are also trained as teaching or research assistants. In positions as TAs or RAs, the students are trained by their faculty mentors in the proper way to interact with other students, complete research oriented tasks, write and or publish technical material, etc.

4A.4.3 Information systems development opportunities

The School offers certification programs in cooperation with Continuing Education. Certification benefits both individual workers and their employers because of the close relationship between CCNA program goals and the fundamental job skills necessary for IT workers in today's job market. Technology Education at the University of Utah now offers the IT Skills Certificate to keep students up-to-date. The certificate program provides the hands-on training needed to better manage and maintain ever-changing IT responsibilities. The courses linked to this program through Continuing Education are as follows: Java, Linux, Apple and CISCO Certification.

These courses are very cost-effective. They are offered to students at a fraction of the cost that would be incurred to obtain certification elsewhere. It should be noted that these courses do not fulfill upper division CS elective requirements, but can be used toward the generic 122 hours of credit necessary to graduate from the University.

4A.5 Undergraduate Outreach Education

The School undergraduate program does not offer education programs at remote sites. The School, positioned as a top department in a flagship research university, believes its overall instructional mission is to train students via direct interactions with faculty and with small recitation sections run by teaching assistants. While our typical student is not as traditional as found in a small non-commuter liberal arts college, we maintain that students need to attend lectures and classes on campus on a daily/weekly basis. Further we strongly believe in the power of pair work, as well as group work.

Students from other Utah schools are able to directly transfer first year courses (we have common course numbering across the state). Courses beyond the first year, and courses from out of state institutions are evaluated on an individual basis to establish compatibility (courses are evaluated based on syllabi by faculty teaching in the same area). As a rule, **at least 10** three-credit CS courses must be taken from the School in order to graduate.

4B. Graduate Curriculum and Programs of Study

4B.1. Graduate Degree and Certificate Requirements

The School of Computing offers two degree types in its graduate program; both degrees are available in both master's and PhD levels. More specifically, they are:

- MS in Computer Science
- PhD in Computer Science
- MS in Computing
- PhD In Computing

At the master's level, both the Computer Science and the Computing degrees offer three routes towards completion:

- Option 1: MS by Course Only
- Option 2: MS by Project
- Option 3: MS by Thesis

The course-only option requires a student to clear 30 graduate level credits. A typical graduate level course is 3 credits; hence this translates to taking and passing 10 graduate level courses. The minimum passing grade for a course is B- for a graduate student and the average grade (of all courses) also needs to be at least B.

The computing degree implements a track structure, where tracks can be created/removed by small groups of faculty (4-5), with approval of the Director of the School. These Track Committees, headed by a Track Director, manage the curriculum for each track. The School offers specializations in the following tracks (all track options are available for both master's and PhD students):

- Computer Engineering — Erik Brunvand is the track director; joint track with the Department of Electrical and Computer Engineering.
- Data Management and Analysis — Jeff Phillips is the track director.
- Graphics and Visualization — Chuck Hansen is the track director.
- Image Analysis — Tom Fletcher is the track director.
- Networked Systems — Sneha Kumar Kasera is the track director.
- Robotics — John Hollerbach is the track director, joint track with
- Department of Mechanical Engineering.
- Scientific Computing — Hari Sundar is the track director.

The CS degree and different tracks in the computing degree differ by the set of required courses they require. For example, in the CS degree, students are required to take the following three required course:

- CS6150: Advanced Algorithms
- CS6460: Operating Systems
- CS6810: Computer Architecture

Whereas each track in the computing degree has its own set of required courses. For example, the Data Management and Analysis track has the following set of required courses:

- CS6140: Data Mining, OR CS6350 Machine Learning
- CS6150: Advanced Algorithms
- CS6350: Database Systems
- CS6630: Visualization

The MS Supervisory Committee (SVC) consists of three members. The following two policies are in place for SVCs:

1. The chair of an SVC must be a regular faculty member (tenured/tenure track) from the School.
2. 2. The majority of the SVC must be regular faculty members (tenured/tenure track) within the School.

Research or adjunct faculty may chair or may be members of supervisory committees if accorded that privilege by the School faculty and the Graduate School. However, exception to only one of the two policies listed above but not both simultaneously will be allowed.

For the project and thesis options of this degree, the MS comprehensive exam will be administered by the student's supervisory committee and can be coupled with (i.e., satisfied by) a project or thesis proposal defense, and/or meeting a specified level of performance on a set of classes. For students not opting for a project or thesis, the comprehensive exam will typically be passed by meeting the grade requirements in the courses required for completing their degree/track, but this can be modified at the discretion of the student's committee.

The master's project is done through an independent study (often formally as an independent study course) with a professor in the School. The parameters for the scope of the project is set forth at the onset of the independent study, and the defense of the project will be done before the student's entire committee plus the professor in charge of the independent study (normally with the chairperson of the committee being the professor with whom the independent study is done). The student is responsible for arranging a time and place for the defense together with the committee.

For the master's thesis option, the supervisory committee must give preliminary approval of the thesis prior to the defense. The defense can be scheduled after this approval. The student must provide one copy of the thesis to the chair of the supervisory committee at least three weeks before the defense, and one copy to each of the other committee members at least two weeks prior to the defense. A complete draft of the thesis must be emailed as a PDF to the Graduate Advisor two weeks prior to the announced time of defense. This copy will be made available for department access. After successfully defending the thesis, the student must obtain approval that the thesis is satisfactory by obtaining signatures from their committee members and the chair of the department by using the Final Reading Approval form, and the Supervisory Committee Approval form. These forms will be submitted along with the final draft of the thesis manuscript to the thesis office. The majority of the signatures of the committee members are required for the thesis editors to start the format approval and the editing process.

At the PhD level, the degrees have similar requirements except for the core course requirements as listed above. The average grade of graduate-level courses for a PhD student needs a grade of B or better, and the GPA for all required course must be at least 3.5. In addition, all PhD students will be required to complete 4 credit hours of Teaching Mentorship with a “Pass” grade. Teaching mentorship will involve working with one or more faculty members (Teaching mentors) on tasks including but not limited to the following:

- Holding student contract hours
- Developing teaching resources (e.g., web pages)
- Lecturing
- Developing and grading assignments

The teaching mentorship must be spread across two semesters (2 credit hours each semester). The required tasks will be laid out by the teaching mentors before the start of the mentorship each semester. A Pass/Fail grade will be assigned for each semester by the teaching mentors based on how well the mentee performs the required tasks. The teaching mentorship must be completed before the written qualifying examination.

A student who has been accepted by the Graduate School is formally admitted to candidacy for the PhD by the University at the recommendation of the student’s supervisory committee. Admission to candidacy occurs after the student:

- forms a supervisory committee,
- files an approved Program of Study form
- completes the course requirements
- passes the written portion of the qualifying examination
- passes the oral portion of the qualifying examination (i.e. proposal defense).

An application for candidacy must be submitted to the Graduate School no later than two months prior to the semester of graduation. For the degree to be conferred, the approved Program of Study form must be completed and the dissertation completed and publicly defended.

A PhD Supervisory Committee (SVC) conducts the student's written qualifying examination, oral qualifying examination, and dissertation defense. This committee consists of five faculty members, at least one member must be from outside the School. The following two policies are in place for SVCs:

1. The chair of an SVC must be a regular faculty member (tenured/tenure track) from the School.
2. 2. The majority of the SVC must be regular faculty members (tenured/tenure track) within the School.

Research or adjunct faculty may chair or may be members of supervisory committees if accorded that privilege by the School faculty and the Graduate School. However, exception to only one of the two policies listed above but not both simultaneously will be allowed. For Computing degrees, further restrictions on committee makeup may apply. All official decisions of the committee are decided by majority vote.

All PhD students must pass a qualifying examination, as specified by the Graduate School. The details of which will be described in Section 4B.5.

The supervisory committee must give preliminary approval of the dissertation prior to the defense. The defense can be scheduled after this approval. The student must provide one copy of the dissertation to the chair of the supervisory committee at least three weeks before the defense, and one copy to each of the other committee members at least two weeks prior to the defense. A complete draft of the dissertation must be sent by email as a PDF to the graduate advisor two weeks prior to the announced time of defense. This copy will be made available for department access.

After successfully defending the dissertation, the student must obtain approval that the thesis is satisfactory by obtaining signatures from their committee members and the chair of the department by using the Final Reading Approval form, and the Supervisory Committee Approval form. These forms will be submitted with the final draft of the thesis manuscript to the thesis office. The majority of the signatures of the committee members are required for the thesis editors to start the format approval and the editing process. The Dean of the Graduate School signs the Final Reading Approval form after all editing is completed and before the thesis release.

Figure 4B-1 summarizes the number of PhD and master's degrees that School has awarded in the recent years; and Table 4B-1 summarizes the pool of applications to the School's graduate program. Lastly, Table 4B-2 summarizes the diversity of School's graduate student body.

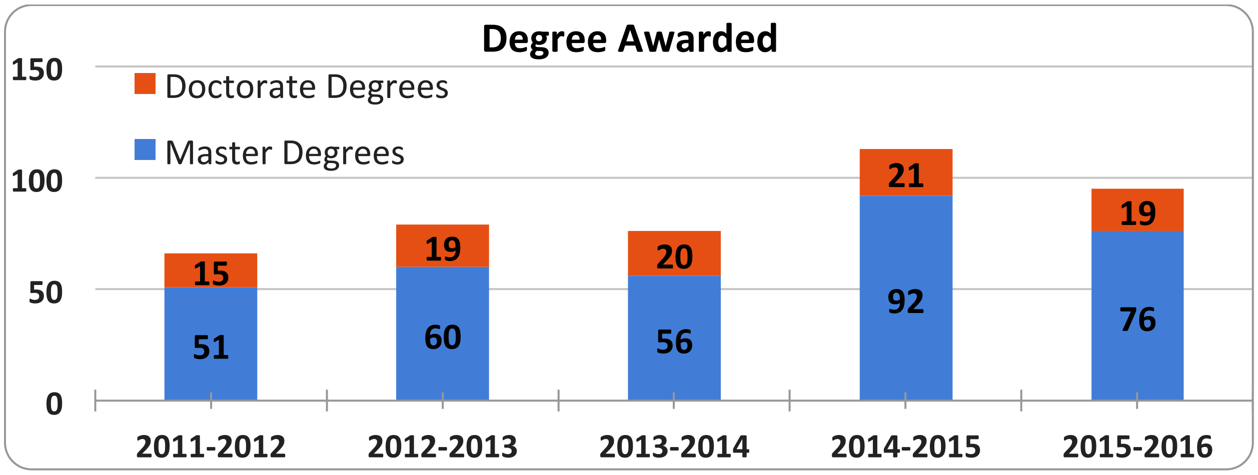


Figure 4B-1: Number of degrees awarded

Academic Year	Number of Students who Applied to Graduate Programs	Average Overall GPA	Average Overall GRE Score	Number of Graduate Students Admitted
2015-16	828	3.68	625/157	194
2014-15	735	3.55	672/157	240
2013-14	558	3.52	655/157	101
2012-13	566	3.60	627/156	72
2011-12	578	3.76	633	58
2010-11	554	3.44	623	92
2009-10	517	3.40	622	84

Table 4B-1: Graduate admissions data

Year/ Spring	Hispanic/ Latino		Black or African-American		American Indian or Alaska Native		Asian		Native Hawaiian or Other Pacific Islander	
	F	M	F	M	F	M	F	M	F	M
2016	0	1	0	0	0	0	4	10	0	0
2015	1	1	1	0	0	0	3	8	0	0
2014	1	3	1	0	0	0	2	6	0	0
2013	0	2	1	1	0	0	2	7	0	0
2012	0	0	0	1	0	1	6	9	0	0
2011	0	0	0	0	0	1	2	4	0	0
2010	0	0	0	1	0	1	2	1	0	0
Year/ Spring	Two or More Races		White		Non-Resident Alien		Race and Ethnicity Unknown		Total	
	F	M	F	M	F	M	F	M	F	M
2016	1	1	11	55	43	109	1	1	60	177
2015	1	0	12	51	40	141	0	2	58	203
2014	1	2	9	64	42	126	0	5	56	206
2013	1	3	3	63	29	108	0	8	36	192
2012	0	1	3	71	22	111	1	11	32	205
2011	0	0	6	71	17	119	1	12	26	207
2010	0	0	3	66	17	102	1	10	23	181

Table 4B-2: Gender and race/ethnicity of graduate students (Spring census data)

The School also offers a Big Data Certificate program. Big Data is impacting many areas of science, engineering, and industry; from analyzing troves of weather data to modeling traffic patterns to processing millions of online customers, it is the enormous data which is creating new opportunities and challenges. To tackle these challenges, one must have the training to store, manage, process and analyze data at these scales. But the challenges are beyond scale alone, the complexity of the data requires new powerful analytical techniques. Finally, it is crucial to have skills in communicating and interpreting the results of this analysis. In the big data program in the School, students will take classes from tenure-track professors actively developing the new techniques for these emerging challenges of big data. And students will learn by doing. They will work on real data, building on the techniques they learn in class under the guidance of the professors. The classes are hands on, and project-focused allowing them to interact with modern software tools and data processing techniques.

Students must complete 5 classes (15 credit hours) with a B or better. At least 4 classes must be among the CORE classes (it is suggested to take all 5). The 5th classes can be any other graduate level classes approved by the Data Management and Analysis Track director. The ELECTIVE classes are pre-approved to fulfill this requirement, but many other (often more sporadically offered) classes are available. The core classes for the big data certificate program are listed below; the list of elective courses is available from the graduate handbook.

- CS6140 Data Mining
- CS6150 Advanced Algorithms
- CS6350 Machine Learning
- CS6530 Database Systems OR CS5530 Database Systems
- CS6630 Visualization

4B.2. Graduate Courses Offered

The list below shows the courses offered by the School in its graduate program.

- CS 6020 Early-Career Research
- CS 6100 Foundations of Computer Science
- CS 6110 Rigorous System Design
- CS 6140 Data Mining
- CS 6150 Advanced Algorithms
- CS 6160 Computational Geometry
- CS 6170 Computational Topology
- CS 6180 Clustering
- CS 6190 Probabilistic Modeling
- CS 6210 Advanced Scientific Computing I
- CS 6220 Advanced Scientific Computing II
- CS 6230 Parallel Computing HPC
- CS 6235 Parallel Program Many-Core
- CS 6300 Artificial Intelligence
- CS 6310 Robotics
- CS 6320 3D Computer Vision
- CS 6330 Intro to Robot Control
- CS 6340 Natural Language
- CS 6350 Machine Learning
- CS 6360 Virtual Reality
- CS 6370 Motion Planning
- CS 6390 Information Extraction
- CS 6460 Operating Systems
- CS 6475 Advanced Compilers
- CS 6480 Advanced Computer Networks
- CS 6490 Network Security

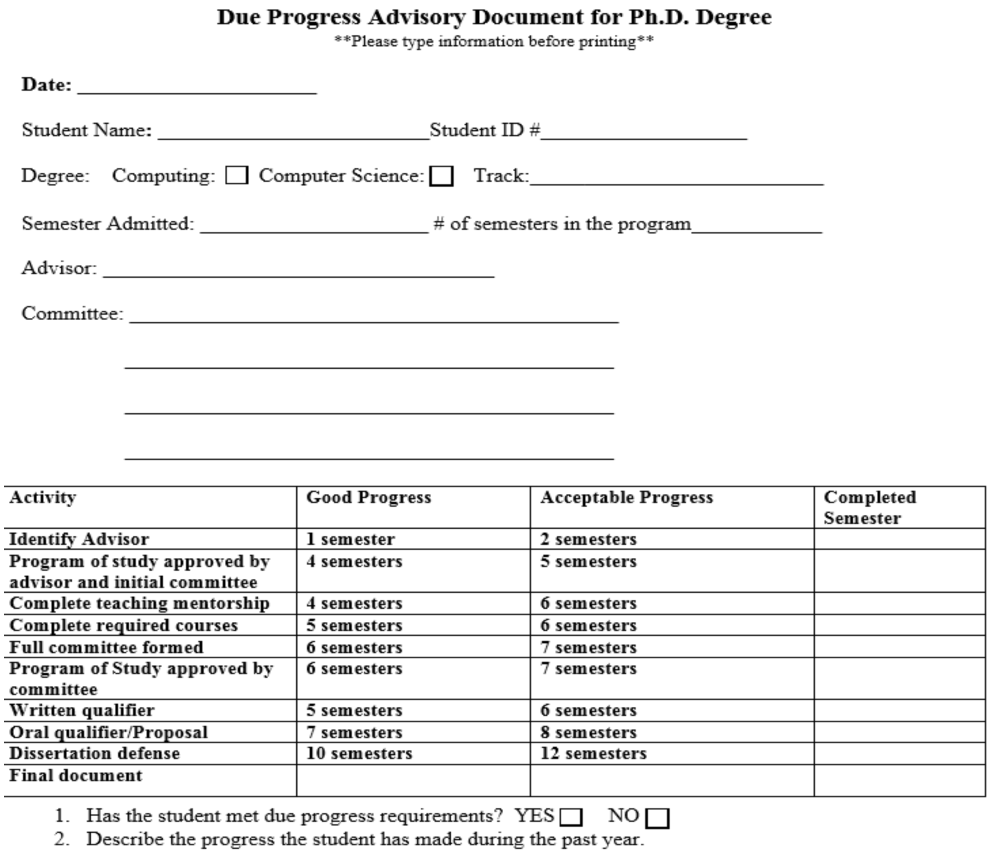
- CS 6510 Functional Programming
- CS 6530 Database Systems
- CS 6550 Foundations of Algorithms in Computer Graphic
- CS 6600 Mathematics of Computer Graphics
- CS 6610 Interactive Comp Graph
- CS 6620 Ray Tracing for Graphics
- CS 6630 Visualization
- CS 6640 Image Processing
- CS 6650 Perception for Graphics
- CS 6660 Physics-based Animation
- CS 6665 Character Animation
- CS 6670 Computer-Aided Geometry
- CS 6680 Computer-Aided Geometric Design II
- CS 6710 Digital VLSI Design
- CS 6712 Digital IC Testing
- CS 6720 Analog IC Design
- CS 6740 CAD of Digital Circuits
- CS 6745 Testing and Verification of Digital Circuits
- CS 6750 Synthesis and Verification of Asynchronous VLSI System
- CS 6770 Advanced Digital VLSI Systems Design
- CS 6780 Embed Sys Design
- CS 6785 Advanced Embedded Software
- CS 6810 Computer Architecture
- CS 6830 VLSI Architecture
- CS 6950 Independent Study
- CS 6956 Wireless and Mobile Networks
- CS 6957 Software Defined Network Architecture
- CS 6960 RT Asynchronous Design
- CS 6961 VSLI Memory Design
- CS 6962 Programming for Engineers
- CS 6963 Distributed Systems
- CS 6964 Caching Networks
- CS 6965 Functional Programming Studio
- CS 6967 Numerical Simulation
- CS 6968 Algorithms & Approximation

- CS 7120 Information-Based Complexity
- CS 7210 Advanced Topics in Scientific Computing
- CS 7310 Robot Mobility and Manipulation
- CS 7320 Sys Identification. Robotics
- CS 7450 Simulation Methods
- CS 7640 Advanced Image Processing
- CS 7650 Realistic Image Synthesis
- CS 7810 Advanced Architecture
- CS 7820 Parallel Architecture
- CS 7930 Colloquium
- CS 7931 Machine Learning Seminar
- CS 7932 Image Motion Estimation and Interpretation
- CS 7933 Graphics Seminar
- CS 7934 Computer Systems Seminar
- CS 7935 Distributed Linear Algebra
- CS 7936 Security Seminar
- CS 7937 Arch/VLSI
- CS 7938 Image Analysis Seminar
- CS 7939 Robotics
- CS 7940 Performance Optimization
- CS 7941 Advanced Seminar
- CS 7942 Visualization Seminar
- CS 7943 Networking Seminar
- CS 7944 Parallel Algorithms Seminar
- CS 7950 Independent Study
- CS 7960 Neuromorphic Architectures
- CS 7970 PhD Dissertation Research

4B.3. Graduate Programs of Study

The followings are copies for the program of study for a master's student. In particular, we have provided a copy for a master's program of study by course only, by project, and by thesis in the CS degree and the Computing degree (for the data management track), respectively. Please refer to Appendix M.

Lastly, we also provide a copy of program of study for a PhD student in the CS degree, and a PhD student in the computing degree from the Image Analysis track. Please refer to Appendix N.



Due Progress Advisory Document for Ph.D. Degree
Please type information before printing

Date: ____________________

Student Name: ________________________ Student ID #__________________

Degree: Computing: ☐ Computer Science: ☐ Track:__________________________

Semester Admitted: ____________________ # of semesters in the program____________

Advisor: ___________________________________

Committee: ____

____

____

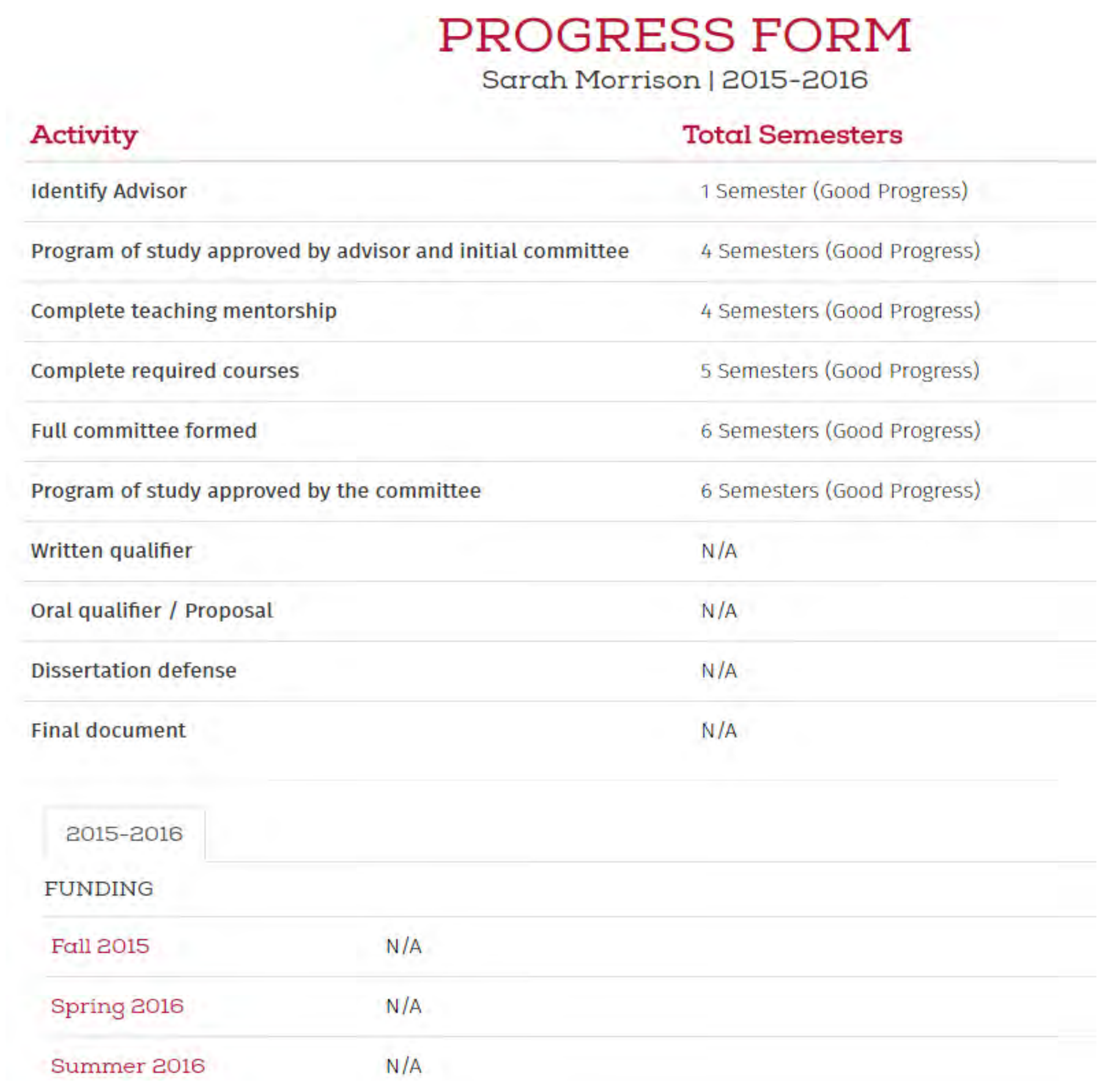
____

Activity	Good Progress	Acceptable Progress	Completed Semester
Identify Advisor	1 semester	2 semesters	
Program of study approved by advisor and initial committee	4 semesters	5 semesters	
Complete teaching mentorship	4 semesters	6 semesters	
Complete required courses	5 semesters	6 semesters	
Full committee formed	6 semesters	7 semesters	
Program of Study approved by committee	6 semesters	7 semesters	
Written qualifier	5 semesters	6 semesters	
Oral qualifier/Proposal	7 semesters	8 semesters	
Dissertation defense	10 semesters	12 semesters	
Final document			

1. Has the student met due progress requirements? YES ☐ NO ☐
2. Describe the progress the student has made during the past year.

Figure 4B-2: Due Progress form for PhD Students

In order to maintain acceptable progress through the graduate program, the School implements a *due progress* system for all PhD students. PhD students are required to fill in the due progress form every fall. A copy of the due progress form is shown in Figure 4B-2. In the past year, the School has moved to an online-based graduate student tracking system that allows all students to fill in the due progress online and the system will automatically decide if a student is making good progress or not. See Figure 4B-3. The Graduate Studies Committee (Director of Graduate Studies and Track Directors) review these forms every fall, and they identify student who are not making good progress (and do not have a clear explanation). The cases of poorly progressing students (typically about 20) are discussed in a faculty meeting, where the faculty make recommendations (for improving/monitoring progress or removal from the program) and the Director of Graduate Studies communicates these findings to the student and/or PhD advisor.



PROGRESS FORM

Sarah Morrison | 2015-2016

Activity	Total Semesters
Identify Advisor	1 Semester (Good Progress)
Program of study approved by advisor and initial committee	4 Semesters (Good Progress)
Complete teaching mentorship	4 Semesters (Good Progress)
Complete required courses	5 Semesters (Good Progress)
Full committee formed	6 Semesters (Good Progress)
Program of study approved by the committee	6 Semesters (Good Progress)
Written qualifier	N/A
Oral qualifier / Proposal	N/A
Dissertation defense	N/A
Final document	N/A

2015-2016

FUNDING

Fall 2015	N/A
Spring 2016	N/A
Summer 2016	N/A

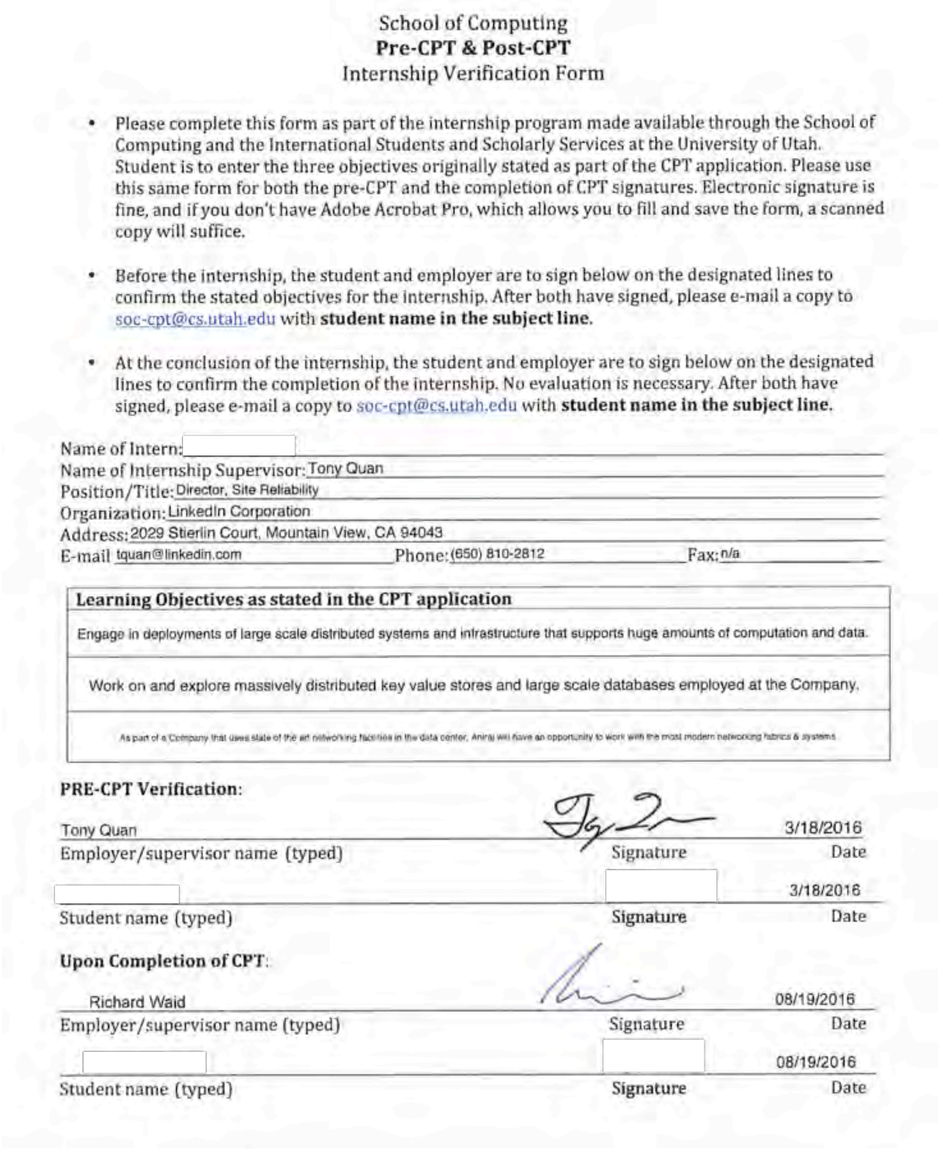
Figure 4B-3: Online Due Process form for PhD students

4B.4. Graduate Professional Development

The School has devoted an increasing amount of efforts towards training professional development and professional ethics and standards. The School offers training sessions for all TAs and TMs, as well as a boot camp for all first year fellowship PhD students.

The school offers various seminars and talks on a wide selection of topics during any given semester, covering different research areas as well as professional development topics. Through a close collaboration with Goldman Sachs, the School has developed the Goldman Sachs distinguished lecture series where distinguished speakers are invited from both academia and industry to present interesting research talks and their work and life experiences with our graduate students (and undergraduate students). These lecture series also offer a venue for the graduate students to engage with working professionals from Goldman Sachs, as well as other local industry partners.

The School also encourages graduate students to find and participate in industrial internships during summer semesters. A rigorous procedure is developed to approve an internship position and to evaluate the outcome of an internship program. For example, all international students need to apply for CPT in order to participate in an internship program. The school has used the following Pre-CPT and Post-CPT form to keep track of students' internship activities, and to provide feedback to the graduate students as how to improve their internship experience.



School of Computing
Pre-CPT & Post-CPT
Internship Verification Form

- Please complete this form as part of the internship program made available through the School of Computing and the International Students and Scholarly Services at the University of Utah. Student is to enter the three objectives originally stated as part of the CPT application. Please use this same form for both the pre-CPT and the completion of CPT signatures. Electronic signature is fine, and if you don't have Adobe Acrobat Pro, which allows you to fill and save the form, a scanned copy will suffice.
- Before the internship, the student and employer are to sign below on the designated lines to confirm the stated objectives for the internship. After both have signed, please e-mail a copy to soc-cpt@cs.utah.edu with **student name in the subject line.**
- At the conclusion of the internship, the student and employer are to sign below on the designated lines to confirm the completion of the internship. No evaluation is necessary. After both have signed, please e-mail a copy to soc-cpt@cs.utah.edu with **student name in the subject line.**

Name of Intern:
Name of Internship Supervisor: Tony Quan
Position/Title: Director, Site Reliability
Organization: LinkedIn Corporation
Address: 2029 Stierlin Court, Mountain View, CA 94043
E-mail tquan@linkedin.com Phone: (650) 810-2812 Fax: n/a

Learning Objectives as stated in the CPT application
Engage in deployments of large scale distributed systems and infrastructure that supports huge amounts of computation and data.
Work on and explore massively distributed key value stores and large scale databases employed at the Company.
As part of a Company that uses state of the art networking facilities in the data center, Aniraj will have an opportunity to work with the most modern networking fabrics & systems.

PRE-CPT Verification:

Tony Quan	[illegible]	3/18/2016
Employer/supervisor name (typed)	Signature	Date
		3/18/2016
Student name (typed)	Signature	Date

Upon Completion of CPT:

Richard Waid	[illegible]	08/19/2016
Employer/supervisor name (typed)	Signature	Date
		08/19/2016
Student name (typed)	Signature	Date

Figure 4B-4: Curricular Practical Training (CPT) example, Pre-CPT and Post-CPT form

4B.5. Graduate Outreach Education

The School doesn't offer education programs at remote sites.

4B.6. Qualifying Exams

The School does not require a qualifying exam for master's students who are in the master's thesis option.

For PhD students, the qualifying exam consists of two parts, a written examination covering the candidate's chosen area of specialization and an oral examination involving a defense of the candidate's written thesis proposal. The written portion of the qualifying examination will cover the candidate's general area of specialization in sufficient depth to demonstrate his/her preparation for conducting PhD level research. Each member of the student's supervisory committee will contribute one or more questions to this exam. The supervisory committee will provide a written evaluation of this portion of the exam, including an indication of whether or not the student will be allowed to proceed to the oral portion of the qualifying examination. Specific details of the written qualifying exam procedures are available from the graduate student handbook that's available online from the department's website.

The oral portion of the qualifying exam involves a defense of the candidate's dissertation proposal. At the supervisory committee's option, it may also include follow up questions relating to the written portion of the exam. All members of the candidate's committee should certify that the proposal is ready to be defended prior to conducting the oral portion of the qualifying exam.

The entire exam should be completed in no more than seven days from initial question assignment to completed answers. Grading should be completed within seven days after the student delivers his/her answers. Each committee member contributing a question will grade that question and provide a specific, written evaluation of the quality and correctness of the answer. Allowable grades on individual questions are:

- HP - high pass
- P - pass
- F – fail

A grade of P signifies the minimal acceptable performance expected from a PhD student. An F grade indicates an answer that is partially correct, but not up to the standards we expect from our PhD students. The members contributing questions will each cast a Pass / Fail vote on the examination as a whole. An overall passing grade should be given to candidates who, through their answers, demonstrate that they are well prepared to conduct PhD level research in their specialty area of CS. The overall exam Pass / Fail grade will be determined by majority vote of those contributing questions. In the event of equal numbers of Pass and Fail votes, the deciding vote will be cast by the Director of Graduate Studies.

A student who fails his/her first attempt may retake the exam once. No conditional pass grades will be given. However, the supervisory committee can at their discretion include fewer questions on repeated exams.

Appendix O provides copies of the most recent five qualifying exams.

In all five cases, students received an overall passing grade.

4B.7. Theses and Dissertations

Table 4B-3 below summarizes the PhD dissertations completed by PhD students graduated between 2009-2016 in the School. Appendix P includes sample thesis/dissertation abstracts.

Student Name	Year	Dissertation Title	Advisor
Margarita Bratkova	2009	Artistic Rendering Of Natural Environments	William Thompson Peter Shirley
Mark Van Langeveld	2009	Educational Impact Of Digital Visualization Tools On A Digital Character Production Course	Robert Kessler
Vincent Pegoraro	2009	Efficient Physically-Based Simulation Of Light Transport In Participating Media	Steven Parker
Brian Rague	2010	A CSI Pedagogical Approach To Parallel Thinking	Joe Zachary
Emanuele Santos	2010	Simplifying The Creating And Deployment Of Collaborative Data Analysis And Visualization Tools	Claudio T. Silva Juliana Freire
Guodong Li	2010	Formal Verification Of Programs And Their Transformations	Ganesh Gopalakrishnan
Joel Daniels	2010	Feature-Aligned, Semi-Regular, Quadrilateral-Only Mesh Generation	Elaine Cohen
Joshua Cates	2010	Shape Modeling And Analysis With Entropy-Based Particle Systems	Ross Whitaker
Junxing Zhang	2010	Wireless Link Signature: Measurements, Methodologies, And Applications	Sneha Kasera Neal Patwari
Kristi Potter	2010	The Visualization Of Uncertainty	Richard Riesenfeld
Luciano Barbosa	2010	Uncovering The Hidden-Web	Juliana Freire
Robert Ricci	2010	Enhancing Realism And Scalability In Network Testbeds	Sneha Kasera
Sarvani Vakkalanka	2010	Efficient Dynamic Verification Algorithms For MPI Applications	Ganesh Gopalakrishnan Robert Michael Kirby II

Siddharth Patwardhan	2010	Widening The Field Of View Of Information Extraction Through Sentential Event Recognition	Ellen Riloff
Tina Ziemek	2010	Evaluating The Effectiveness Of Orientation Indicators With An Awareness Of Individual Differences	William Thompson Sarah Creem Regehr
Abraham Stephens	2011	Control Of Spatial And Temporal Fidelity With Adaptive Sampling	Steven Parker
Andrew Kensler	2011	Software Algorithms For Hardware Ray Tracing	Steven Parker
Anh Vo	2011	Scalable Formal Dynamic Verification Of MPI Programs Through Distributed Causality Tracking	Ganesh Gopalakrishnan
Daniel Gebhardt	2011	Energy-Efficient Design Of An Asynchronous Network-On-Chip	Kenneth Stevens
Elizabeth Jurrus	2011	Segmentation Of Neurons From Electron Microscopy Images	Tolga Tasdizen
Erik Anderson	2011	The Analysis And Visualization Of EEG Data Using A Provenance-Enabled Environment And Its Applications For Visualization	Claudio T. Silva
Hoa Thanh Nguyen	2011	Automatic Catalog Construction For Product Search Engines	Juliana Freire
Huy T. Vo	2011	Designing A Parallel Dataflow Architecture For Streaming Large-Scale Visualization On Heterogeneous Platforms	Claudio T. Silva
Justin Luitjens	2011	The Scalability Of Parallel Adaptive Mesh Refinement Within Uintah	Martin Berzins
Kevin Atkinson	2011	ABI Compatibility Through A Customizable Language	Matthew Flatt Gary Lindstrom
Linh Khanh Ha	2011	High Performance Multiscale Image Processing Framework On Multi-Gpus (Graphics Processing Units) With Applications To Unbiased Diffeomorphic Atlas Construction	Claudio T. Silva
Mathias Schott	2011	Using Incremental Filtering For Enhancing The Depth Perception Of Interactive Direct Volume Rendering	Chuck Hansen
Robert Thacker	2011	A New Verification Method For Cyber-Physical Systems	Chris Meyers
Sachin Goyal	2011	A Collective Approach To Harness Idle Resources Of End Nodes	John B. Carter
Suraj Musuvathy	2011	Medial Axis Of Regions Bounded By B-Spline Curves And Surfaces	Elaine Cohen
A.N.M. Imroz Choudhury	2012	Visualizing Program Memory Behavior Using Memory Reference Traces	Paul Rosen Steven G. Parker
Aniruddha Udipi	2012	Designing Efficient Memory For Future Computing Systems	Rajeev Balasubramonian
Avishek Saha	2012	Some Models And Measures For Learning On A Budget	Suresh Venkatasubramanian
Blake Nelson	2012	Accurate And Interactive Visualization Of High-Order Finite Element Fields	Robert Michael Kirby II
Carson Brownlee	2012	Parallel Ray Tracing In Scientific	Chuck Hansen

		Visualization	
Dafang Wang	2012	Finite Element Solutions To Inverse Electrocardiography	Christopher R. Johnson Robert Michael Kirby II
David Koop	2012	Managing Provenance For Knowledge Discovery And Re-Use	Juliana Freire Claudio Silva
Eric Eide	2012	Software Variability Mechanisms For Improving Run-Time Performance	Matthew Flatt
Karthik Ramani	2012	Cogene: An Automated Design Framework For Domain-Specific Architectures	Alan Davis
Lethuy Tran	2012	Numerical Study And Improvement Of The Methods In Uintah	Martin Berzins
Lu Zhao	2012	A Program Logic And Its Application In Fully Verified Software Fault Isolation	John Regehr
Matthew Jared Probst	2012	Distributed Friend-To-Friend Framework And Services Using Schoolial Networks	Sneha Kasera
Saha Avishek	2012	Some Models And Measures For Learning On A Budget	Suresh Venkatasubramanian
Samuel Gerber	2012	Nonparametric Models For High-Dimensional Data Analysis	Ross Whitaker
Tobias Martin	2012	Analysis-Aware Higher Order Smooth Three-Dimensional Representations: Creation, Simulation And Visualization	Elaine Cohen
Anton Burtsev	2013	Deterministic Systems Analysis	John Regehr
Brian Summa	2013	Interactive Digital Photography At Scale	Valerio Pascucci
Curtis Madsen	2013	Stochastic Analysis Of Synthetic Genetic Circuits	Chris Meyers
Hanieh Mirzaee Teshnizy	2013	Smoothness-Increasing Accuracy-Conserving Filters (SIAC) For Discontinuous Galerkin Solutions	Robert Michael Kirby II
Jeffrey Jestes	2013	Efficient Summarization Techniques For Massive Data	Feifei Li
Jianjun Duan	2013	Formal Verification Of Device Drivers In Embedded Systems	John Regehr
Jonathan Rafkind	2013	Syntactic Extension For Languages With Implicitly Delimited And Infix Syntax	Matthew Flatt
Kevin Tew	2013	Places: Parallelism For Racket	Matthew Flatt
Kshitij Sudan	2013	Data Placement For Efficient Main Memory Access	Rajeev Balasubramonian
Matthew Berger	2013	Shape Analysis Of Defect-Laden Data	Claudio T. Silva
Michael Allen Parker	2013	Efficient User-Level Event Notification	Alan Davis
Nikhil Singh	2013	Multivariate Regression Of Shapes Via Deformation Momenta: Application To Quantifying Brain Atrophy In Aging And Dementia	P. Thomas Fletcher
Niladrish Chatterjee	2013	Designing Efficient Memory Schedulers For Future Systems	Rajeev Balasubramonian
Parasaran Raman	2013	Exploring The Landscape Of Clusterings	Suresh Venkatasubramanian
Piyush Rai	2013	Learning Latent Structures Via Bayesian Nonparametrics: New Models And Efficient Inference	Hal Daume III
Sriram Nandha	2013	Exploiting Cross Layer Opportunities For	Sneha Kasera

Premnath		Secrecy And Efficiency In Wireless Networks	
Subodh Sharma	2013	Predictive Analysis Of Message Passing Applications	Ganesh Gopalakrishnan
Tiago Queiroz	2013	Towards The Theory And Practice Of Verifying Visualizations	Claudio T. Silva
Wangchao Le	2013	Supporting Scalable Data Analytics On Large Linked Data	Feifei Li
Yong Wan	2013	Fluorender, An Interactive Tool For Confocal Microscopy Data Visualization And Analysis	Chuck Hansen
Zhisong Fu	2013	Parallel-Streaming Algorithms For Solving Partial Differential Equations On Unstructured Meshes	Ross Whitaker Robert Michael Kirby II
Andrew M. Wilder	2014	Computer-Assisted Approaches To Intrafascicular Multielectrode Stimulation	Gregory A. Clark Harold C. Daume III
Arthur W. Mahoney Jr.	2014	Advanced Methods For Controlling Untethered Magnetic Devices Using Rotating Magnetic Fields	Jake Abbot
Carlos Scheidegger	2014	Provenance Of Exploratory Taska In Scientific Visualization: Management And Applications	Claudio T. Silva
Christopher Earl	2014	Introspective Pushdown Analysis And Nebo	Matthew Might
Daniel Gerszewski	2014	Physics-Based Animation Of Large-Scale Splashing Liquids, Elastoplastics Solids, And Enhanced Reduced Fluid Simulation	Adam Bargteil
David Nellans	2014	Improving Operating System And Hardware Interactions Through Co-Design	Erik Brunvand
Haimashree Bhattacharya	2014	Tools For Physics-Based Computer Animation For Generating Surfaces, On Surfaces And From Surfaces	Adam Bargteil
Liang Zhou	2014	Multivariate Transfer Function Design	Chuck Hansen
Manu Awasthi	2014	Managing Data Locality In Future Memory Hierarchies Using A Hardware Software Codesign Approach	Rajeev Balasubramonian
Qingyu Meng	2014	Large-Scale Distributed Runtime System For DAG-Based Computational Framework	Martin Berzins
Ruihong Huang	2014	Improving Information Extraction By Discourse-Guided And Multifaceted Event Recognition	Ellen Riloff
Shuying Liang	2014	Static Analysis Of Android Applications	Matthew Might
Varun Shankar	2014	Radial Basis Function-Based Numerical Methods For The Simulation Of Platelet Aggregation	Robert Michael Kirby II
Wei Liu	2014	Resting State Functional Magnetic Resonance Imaging Analysis By Graphical Model	P. Thomas Fletcher
Weibin Sun	2014	Harnessing GPU Computing In System-Level Software	Robert Ricci
Xiang Hao	2014	Improved Segmentation And Analysis Of White Matter Tracts Based On Adaptive Geodesic Tracking	P. Thomas Fletcher

Yang Chen	2014	Improving The Utility Of Compiler Fuzzers	John Regehr
Benjamin Jones	2015	Artist-Guided Physics-Based Animation	Adam Bargteil
Bo Wang	2015	Modeling Pathological Changes By Leveraging Normative Models, Alternate Domain Database, And User Interactions	Guido Gerig
Brad Loos	2015	Modular Radiance Transfer	Peter-Pike Sloan Chuck Hansen
Harsh Bhatia	2015	Consistent Feature Extraction From Vector Fields: Combinatorial Representations And Analysis Under Local Reference Frames	Valerio Pascucci
James Fishbaugh	2015	Spatiotemporal Modeling Of Anatomical Shape Complexes	Guido Gerig
Jonathan Bronson	2015	New Approaches To Quality Tetrahedral Mesh Generation	Ross Whitaker
Josef Bo Spjut	2015	Efficient Ray Tracing Architectures	Erik Brunvand
Joshua DeBever	2015	Adaptive Model-Predictive Control And 3D Acoustic Radiation Force Imaging For The Improvement Of Magnetic Resonance-Guided Focus	John Hollerbach
Mingwang Tang	2015	New Problems In Exploring Distributed Data	Feifei Li
Peng Li	2015	Practical Symbolic Execution Analysis And Methodology For GPU Programs	Ganesh Gopalakrishnan
Seth Hintze Pugsley	2015	Opportunities For Near Data Computing In Mapreduce Workloads	Rajeev Balasubramonian
Steven Lyde	2015	Improving Control-Flow Analysis Of Higher-Order Languages	Matthew Might
Xing Lin	2015	Using Similarity In Content And Access Patterns To Improve Space Efficiency And Performance In Storage Systems	Robert Ricci
Xuejun Yang	2015	Random Testing Of Open Source C Compilers	John Regehr
Amirali Abdullah	2016	Bounds For Nearest Neighbor Algorithms And Embeddingsbounds	Suresh Venkatasubramanian
Anand Venkat	2016	An Integrated Compiler And Runtime Framework For Sparse Matrix Codes	Mary Hall
Anshul Joshi	2016	WPCA: The Wreath Product Cognitive Architecture	Thomas Henderson
Arijit Banerjee	2016	Designing Novel, Efficient Future Communication Systems Leveraging Network And Application Collaboration	Sneha Kasera Jacobus van Der Merwe
Ashequl Qadir	2016	Acquiring Knowledge For Affective State Recognition In Schoolial Media	Ellen Riloff
Ashley Guinan	2016	Skin Stretch Feedback To Guide Hand Motions	William Provancher
Daniel Kopta	2016	Ray Tracing From A Data Movement Perspective	Erik Brunvand
James King	2016	Reducing Irregularities In Control Flow And Memory Access On GPU Architectures	Robert Michael Kirby II
John Moeller	2016	Kernals And Geometry Of Machine Learning	Suresh Venkatasubramanian

Linda DuHadway	2016	Course Transformation: Content, Structure And Effectiveness Analysis	Thomas Henderson
Manjunath Shevgoor	2016	Enabling Big Memory With Emerging Technologies	Rajeev Balasubramonian
Mark Kim	2016	GPU-Enabled Surface Visualization	Chuck Hansen
Pascal Grosset	2016	Investigating Depth Of Field In Volume Rendering, And Distributed Volume Rendering On High Performance Computing Systems	Chuck Hansen
Protonu Basu	2016	Compiler Optimizations And Autotuning For Stencils And Geometric Multigrid	Mary Hall
Saurav Muralidharan	2016	Abstractions And Autotuning Techniques For Adaptive Programming	Mary Hall
Sidharth Kumar	2016	A Scalable And Turnable Adaptive Resolution Parallel I/O Framework	Valerio Pascucci
Sriram Aanthakrishnan	2016	A Composable Framework For Program Analysis	Ganesh Gopalakrishnan
Ting Liu	2016	Image Segmentation With Hierarchical Models	Tolga Tasdizen
Veni Gopalakrishna	2016	Surface- Based Image Segmentation Using Application-Specific Priors	Ross Whitaker
Wei-Fan Chiang	2016	Efficient Floating-Point Error Testing And Rigorous Mixed Precision Tuning	Ganesh Gopalakrishnan

Table 4B-3: PhD Dissertations from 2009--2016.

Table 4B-4 below summarizes the master's theses completed by master's students graduated between 2009-2016 in the School.

Student Name	Year	Thesis Title	Advisor
Clark Michael	2010	Enhancing Covert Communications With Colluding Receivers	Sneha Kasera
Jeong Mina	2010	Proof Producing Satisfiability Modulo Theory	Konrad Slind
Meakin Benjamin	2010	Multicore System Design With Xum: The Extensible Utah Multicore Project	Ganesh Gopalakrishnan
Rudy Gabriel	2010	Cuda-Chill: A Programming Language Interface For Gpgpu Optimizations And Code Generation	Mary Hall
Umadevi Venkataraju Kannan	2010	Automatic Markup Of Neural Cell Membranes Using Boosted Decision Stamps	Tolga Tasdizen
Chiang Wei-Fan	2011	Heuristics For Efficient Dynamic Verification Of Message Passing Interface And Thread Programs	Ganesh Gopalakrishnan
Maheshwari Manas	2011	Enhancing Reliability In Device-Free Localization	Sneha Kasera
Pagariya Rohit	2011	Direct Equivalence Testing Of Embedded Software	John Regehr

Pokkunuri Rama Krishna Sandeep	2011	Exploiting Example Structure In Multiple Instance Learning	Hal Daume III
Seth Manav	2011	Emergency Service In Wi-Fi Networks Without Access Point AsSchooliation	Sneha Kasera
Tuttle Claurissa	2011	Pedvis: A Structured, Space-Efficient Technique For Pedigree Visualization	Claudio T. Silva
Jadhav Shreeraj	2012	Consistent Representation Of Two-Dimensional Flow	Valerio Pascucci
Lakshmane Gowda Prarthana	2012	Exploring Bluetooth For Received Signal Strength Indicator-Based Secret Key Extraction	Sneha Kasera
Pastor Acosta Isaac	2012	Upper Limb Rehabilitation Of Stroke Patients Using Kinect And Computer Games	Stacy J. M. Bamberg
Ramalingam Shreyas	2012	Improving High-Performance Sparse Libraries Using Compiler Assisted Specialization: A Petsc (Portable, Extensible Toolkit For Scientific Computation) Case Study	Mary Hall
Ward Stephen	2012	Deformation Embedding For Point-Based Elastoplastic Simulation	Adam W. Bargteil
Gupta Shobhit	2013	Detecting And Tracking Human Motion In Variance-Based Radio Tomography Imaging	Suresh Venkatasubramanian
Machanavajhala Swetha	2013	Accent Classification: Learning A Distance Metric Over Phonetic Strings	Suresh Venkatasubramanian
Raj Mukund	2013	Effect Of Animated Self-Avatar In Virtual Environments	William B. Thompson
Saquib Nazmus	2013	Visualizing Intrinsic Isosurface Variation Due To Uncertainty Through Heat Kernel Signatures	Robert Michael Kirby II
Strum Matt	2013	Flowops: Open Access Network Management And Operation	Robert Ricci Jacobus Van der Merwe
Stuart David Alexander	2013	Coarse Tetrahedral Meshing For Interactive Simulation	Adam W. Bargteil
Tirpankar Nishith	2013	Using Sparse Parametrization Of Deformation Fields As Means To Classification	Guido Gerig
Dasa Subramanyam Naveen	2014	Efficient Switching Between Wifi And Cellular Networks For Robust Internet Connectivity	Sneha Kasera
Desai Amey	2014	Streaming Algorithms For Matrix Approximation	Jeffrey Phillips
Gritton Christopher E.	2014	Ringing Instabilities In Particle Methods	Martin Berzins Robert Michael Kirby II
Koslover Rebecca	2014	Influence Of Direction Cueing Modality On Situation Awareness In Mobile Navigation	William Provancher

Lewis Thomas	2014	A Gpu-Based Maximal Independent Set Aggregation Strategy: Algorithms, Comparisons, And Applications Within Algebraic Multigrid	Robert Michael Kirby II
Lewis Thomas	2014	Hybrid Scheduling For Graph-Based Algorithm Decomposition In High-Performance Computing Environments	Robert Michael Kirby II
Nayak Prashanth	2014	Detecting And Mitigating Malware In Virtual Appliances	Eric Eide
Pullakandam Raghuveer	2014	Emustore: Large Scale Disk Image Storage And Deployment In The Emulab Network Testbed	Robert Ricci
Rivera Axel Y.	2014	Using Autotuning For Accelerating Tensor Contraction On Graphics Processing Units (Gpus)	Mary Hall
Robison Braden	2014	Hybrid Scheduling For Graph-Based Algorithm Decomposition In High-Performance Computing Environments	James Sutherland
Sorensen Tyler Rey	2014	Testing And Exposing Weak Graphics Processing Unit Memory Models	Ganesh Gopalakrishnan
Syed Aisha	2014	Realistic Traffic Shaping In Dummynet Link Emulator	Robert Ricci
Michelle Hromatka	2015	Multisite Learning In Medical Image Analysis	P. Thomas Fletcher
Joseph Jithu	2015	Cenet— Capability Enabled Networking: Towards Least-Privileged Networking	Jacobus Van der Merwe
Kano Makito	2015	Seacat: An Sdn End-To-End Containment Architecture	Jacobus van der Merwe
Rungta Atul	2015	Manyvis: Multiple Applications In An Integrated Visualization Environment	Valerio Pascucci
Simonic Klemen	2015	Concept Aware Co-Occurrence And Its Applications	Feifei Li
Singh Raghvendra	2015	Scalable Spatial Scan Statistics	Jeffrey Phillips
Yadav Nitin	2015	Community-Affinity: Measuring Strength Of Memberships Of Nodes In Network Communities	Suresh Venkatasubramanian
Shanmugam Praveen Kumar	2016	Deidtect - Distributed Elastic Intrusion Detection Architecture	Jacobus Van Der Merwe
Shen Yang	2016	Soft Shadow Mip-Maps	Cem Yuksel

Table 4B-4: Master's Theses from 2009--2016.

5A. Undergraduate Program Effectiveness - Outcomes Assessment

The School of Computing has established a comprehensive set of *learning outcomes* specific to the Bachelors Degree in Computer Science. We have numerous avenues for evaluating of these outcomes based on the success of our students in courses and in industry (or graduate school).

Bachelors of Computer Science Learning Outcomes:

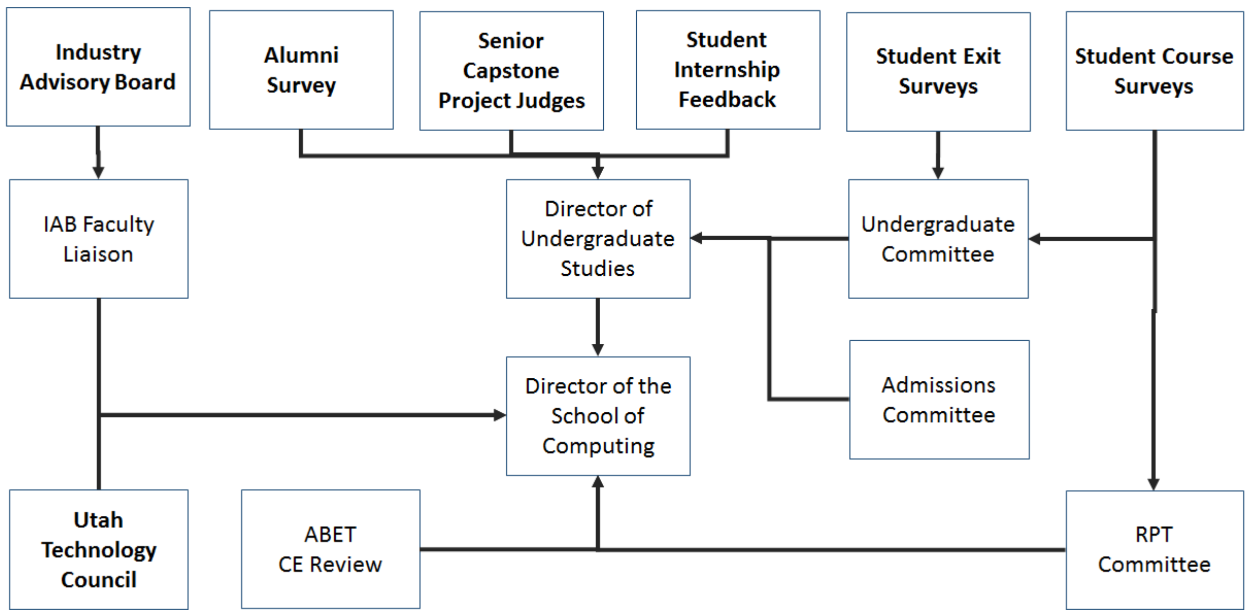
- Demonstrate a knowledge of general computer science principles:
  - Demonstrate the ability to develop and work with abstractions
  - Demonstrate a knowledge of classical algorithms and data structures
  - Be able to analyze the efficiency of algorithms
  - Be able to design and implement efficient algorithms to solve computational problems
  - Demonstrate deep knowledge of a specific area of computer science
- Show proficiency as a software engineer:
  - Demonstrate programming skills and the ability to learn new languages and tools
  - Demonstrate the ability to work with a large code base
  - Demonstrate an understanding of the interaction between application software, systems software, and hardware
- Demonstrate general engineering and communication skills:
  - Be able to work in a team setting
  - Demonstrate clarity in technical communication
  - Demonstrate the ability to apply current ethical standards as related to computer science / software engineering

The following sections detail the procedures put in place to monitor success and some of the results/changes we have made.

5A.1 Undergraduate Outcomes Assessment Procedures

5A.1.1 BS CS assessment pathways

The following chart demonstrates the various pathways for assessment used in evaluation of our degree. A discussion of each assessment strategy is listed below.



5A.1.2 Undergraduate student exit surveys

Undergraduate students who have applied for graduation are sent an electronic exit survey at the end of their final semester. Responses are voluntary but strongly encouraged. Questions include three main areas: (1) Data about the student's completion of the program. For example: Why did the student choose the University of Utah? How long did they take to graduate? Did they attend full or part-time? (2) Students future endeavors. For example: Are they going to grad school or into industry? Where did they apply for jobs? Where will they be going, and what till the pay range be? and (3) Assessments of our program. For example: Which professors and/or courses did they like the best/least? What feedback do they have regarding our program, staff or facilities?

In the past, an ad hoc system was in place for summarizing and evaluating these surveys. This has been identified as needing improvement, and a new more formal process is being introduced. Going forward, at the end of each graduation semester, the undergraduate exit surveys will be provided to the School Undergraduate Committee. This committee will review the data and then bring any trends and/or recommendations to the attention of the Director of Undergraduate Studies and to the Director of the School.

5A.1.3 Faculty/Instructor course evaluations

Students in all courses at the University are asked to complete a post-course instructor evaluation. These evaluations have traditionally been read by the Director/Associate Director of the School and the RPT (Retention, Promotion, Tenure) Committee to help evaluate and evaluate and mentor instructors. Further, this information is used by the Dean of the College to recognize outstanding faculty. Public recognition of the importance of teaching excellence is part of the University culture.

Going forward the School is developing a plan to have the undergraduate course evaluations reviewed by the Undergraduate Committee, which will then summarize any trends and/or recommendations and bring them to the School Director (via the Director of Undergraduate Studies).

Feedback from course evaluations has generally impacted the curriculum through ad-hoc processes. For instance in the past, the course evaluations were mostly used during the RPT process, and impressions from this exercise would feed back up to the Director's office. One example of this was the reviews for the instructor of our discrete math course (CS 2100). While the reviews were below average, further evaluation found that all professors were receiving below-average feedback. This observation led to a restructuring of the course with better learning outcomes.

5A.1.4 Alumni survey

In 2016, the School completed its first, long-range, alumni survey, asking questions about their overall success as graduates from the CS or CE programs. Results were collected from over 150 alumni, approximately 85 who have graduated with a Bachelors of Computer Science from 2009-2016 (~15% response rate). The alumni were asked questions relating to their industrial career, including how well they were prepared technically and in their soft-skills, how soon they were employed, what salaries they earned, etc.

Several very interesting conclusions can be made from their responses.

5A.1.4.a Preparedness

Below is a rating scale for each graduating class from 2009 to 2010 on a 1-5 with 1 being "Very Unprepared" and 5 being "Very Prepared". "Overall" refers to their perception of how well they have done over their entire career based on their Utah education. "When Hired" refers to how well they thought they were prepared for their first post-graduation employment compared with other first year employees. We further asked about their perception on how well they were prepared in their hard vs. soft skills. We believe that our alumni are overall very pleased with their lifelong skill preparation.

	Total Responses	Overall Post-Degree	When Hired	Tech Preparation	Soft Skills Preparation
2009	10	4.3	4.2	4.0	3.8
2010	12	3.9	4.0	3.7	3.3
2011	5	4.4	4.0	3.8	3.8
2012	13	4.1	4.1	3.8	3.8
2013	5	5.0	4.6	4.2	4.4
2014	15	3.9	4.1	4.2	4.2
2015	17	4.2	4.3	4.1	3.7
2016	8	4.4	3.9	4.1	4.1

Table 5A-1: Results from alumni surveys

5A.1.4.b. Industry Work Areas

In order to assess if we are teaching courses in the proper areas, we asked our alumni to identify key areas/topics of work that they have been involved with in their career.

software engineering	24
web	21
database	15
testing	13
research	11
mobile	8
game	7
visual	7
graphics	3
security	1

Table 5A-2: Key topics/areas that alum ni work with in their careers

It should be noted that we currently teach all of these topics, either in specific courses or as units in more general courses. See section 5A.2.2 Additional Topic Offerings for more discussion.

5A.1.4.c. Salary

As part of the survey we asked for alumni salary levels (starting and current, years 2009-2016). It is clear that our students, on average are making very competitive salaries (e.g., the median current salary is close to $100,000). In all but one case our student's salaries have increased over time, and the average increase has been over $5000 a year. It should be noted, that when we wrote the survey, we considered $160,000+ a reasonable "cap" for top salaries, but we have subsequently learned a non-trivial number of our students are making substantially more than that.

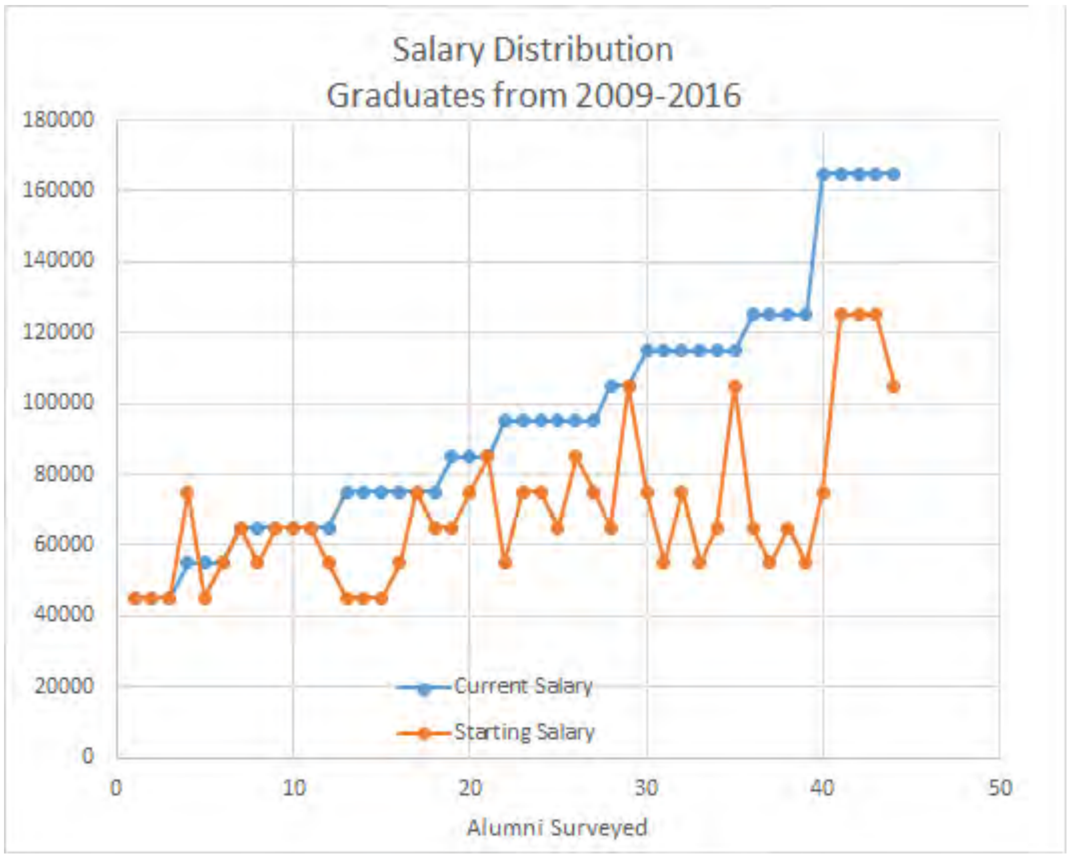


Figure 5A-1: Alumni salary distribution from survey

Our alumni survey has been very illuminating. While our alumni seem quite positive with their overall education and preparedness, they did list areas they thought could use improvement. These include more focus on software engineering skills on large preexisting code bases; suggestions for making Internships either required or strongly encouraged; more courses teaching *up-to-date* technologies (e.g., Node and Angular, or Ruby/Rails); more emphasis on machine learning, data science, and visualization; etc. Some suggested a distinguishing between different tracks, such as *software engineer* (more coding and tools) and *computer scientist* (more math and theory). It should be noted, that in some cases, efforts have already been put in place to address some of these concerns. Going forward, the School's Undergraduate Committee will be considering all of these suggestions and propagate these findings to the School faculty. We plan to follow this up with more alumni surveys in the future, as well as some specific highlighted alumni reports where we will ask our alumni to better describe their career.

5A.1.5 Industry Advisory Board and Utah Technology Council

Please see section 1.2.2.b for an overview of the Industry Advisory Board (IAB) and the Utah Technology Council (UTC). The IAB has made several suggestions throughout the past 7 years, but has consistently argued for a rigorous training of students across a broad range of "core" CS topics. For example, the IAB suggested that we incorporate tracks, and especially a "systems" or "honors" track, that would help guide our students toward a program of study that would help them a) stand apart, and b) be valuable employees. The School instituted a track system (recommended selections of electives and an associated certificate) in 2013 and has identified twelve separate tracks through the program (e.g., Software Development, Artificial Intelligence, Visual Computing, Theory, etc.).

5A.1.6 Undergraduate student internship feedback

The School offers a technical elective course for students who participate in full-time internships (usually over the summer). On average, about 10-20 students take the course each year. It should be noted that many other students work part/full-time jobs and/or do internships without taking the formal course.

The for-credit, internship course requires the students to write weekly reports detailing the things they are learning while at the internship, as well as describing what they are learning about themselves. The program further requires a midterm and final evaluation by student's employer, and a visit from a supervising faculty member.

The final evaluation asks the employer to comment on what the student has done, but also asks questions such as: "If you had a position available, would you hire this student?" and "How well has the student's education prepared them?" *Universally*, these answers come back with positive responses such as: "yes we would offer a position" or "yes we have offered a position" and "yes, the University curriculum has done a good job of preparing the student". We feel the overwhelmingly positive feedback that our students receive is a strong indicator of the success of our program.

It should be noted that this ubiquitous positive feedback has been a hallmark of the internship program since its inception. Further, the supervising faculty has received very positive feedback from employers during meetings as well as a strong indication that industry is looking for tighter bonds with the School, and a greater pipeline of interns. This feedback is also generally consistent with feedback from direct interactions with industry in the Salt Lake Valley/Wasatch Front as well as other large markets (e.g. California), where employers claim that the preparation of our students compares favorably with those from the best schools in the nation.

5A.1.7 Senior capstone project judges

For each graduating class of seniors (those completing their senior capstone project) the School holds a *Public Demonstration Day*, which requires student teams to demonstrate their software projects in a public setting. The teams consist of approximately 4 students working over two semesters to create a substantial software solution. Members of academia and industry are asked to attend as judges and to rate the teams on a scale from 1 (very poor) to 10 (outstanding) based on their own corporate (or academic) background. Over the past five years, the top teams have consistently averaged 8 to 9.5 points. Informal feedback from the judges as been uniformly positive, and many of the students have been recruited at that event. Last year, the winning team built an application for air-quality monitoring in Salt Lake Valley and was featured on the local TV News.

5A.1.8 Computer Engineering ABET Certification

The joint School/ECE Computer Engineering Degree is ABET certified. Many of the required and/or elective CE courses are taught directly via Computer Science, and thus are subject to the ABET review process. Sample courses reviewed are CS 1410, CS 2420, CS 2100, CS 3500, CS 3505, CS 3810, etc. The strong formalisms in the ABET review require faculty to assert course objectives, collect and analyze student "exemplars", and to show success (or failure) of the students, followed by improvement plans. The School feels that many of the goals of ABET (consistent and rigorous review and improvement) are similar to the goals we have for our own program.

5A.2 Undergraduate Outcomes Assessment Feedback

Below we discuss specific examples of how the curriculum/School has changed in response to assessment measures:

5A.2.1 Senior capstone expansion

The School, College, and University all firmly believe in a *capstone experience*. The School has offered a software engineering lab, as a class, for over 15 years, but in the last 7 years feedback from students and instructors indicated that this course needed to be explicitly denoted and run as a *Senior Capstone*, and that a single semester was not enough time to fully realize the benefits of such an experience (including work on writing, presenting, and programming). In 2011, the School developed a two-semester Capstone Design sequence (which is now required). Students taking this course spent an extra semester to define teams, decide on a project, create a design specification, and prototype an initial system. The second semester (Capstone Project) was then available to complete three full "sprints" (or iterations) on the idea. Feedback from teams who have completed substantial projects over the two semesters indicate that they are very much in favor of the year long version of the course.

It should be noted that a few students, mainly those who are working in a software development capacity in industry, have suggested that we make the Senior Capstone optional. While it is unlikely that we would remove the course, we are in discussions about if an alternative could be put in place, perhaps having students take one more elective and the internship course in place of the capstone. Currently, students who chose to do a BS Thesis (and take the associated supervisory credits for that) are not required to take the Capstone Design course (approximately 10-20 students each year choose the thesis option).

5A.2.2 Multiple yearly course offerings

Based on student feedback, TA feedback, course surveys, and published education best practices, it became clear that the continued increased sizes of our required CS courses was doing a disservice to our students.

In 2010, the School noted that CS 2100 (Discrete Structures), was consistently receiving poor feedback in course evaluations. The faculty instructors primarily responsible for rotating through the course met with the Director of the School to propose some major changes. Of these, one was the decision to offer the course every semester, and thus halving the course size. While this has not reduced the percentage of students who do not successfully complete the course, it has offered smaller sections, and an immediate option for repeating the course.
In 2015, based on the perceived success of CS 2100 (and the consistently increasing enrollments), the School responded to the increased demand for CS courses, as well as the requests from students for more flexibility (resulting in a higher rate of retention) by doubling its offerings of the required sophomore software practice sequence, CS3500-CS3505 (now offered both Fall-Spring and Spring-Fall). There have been several advantages to this scheme, including:

1. Students who were unsuccessful in CS 3500, were previously required to wait a year before continuing the major. We believe that the increased offerings of CS 3500 has resulted in an increased retention rate.
2. Multiple faculty now teach the course, giving struggling students a chance to find an instructor that (perhaps) better meets their personal learning style.
3. The class size of the (traditional) fall session of CS 3500 has been able to be reduced in size by 20% while still increasing the overall quantity of new majors.

Based on the success of offering CS 2100 and CS 3500 twice a year, *the School now offers all required courses twice a year*.

5A.2.3 Additional topic offerings

Feedback from our Industry Advisors has indicated that various expertise are in very high demand in industry, including: security, human computer interaction, data science, visualization, etc. The School has expanded course offerings over the past 7 years in all of these areas, including new courses titled Human Centered Security, Network Security, HCI, Math for Data, Intro to Data Science, Data Mining, Big Data Computer Systems, Visualization, etc. Further, the School has targeted new faculty hires in these fields, hiring in data analysis, human computer interaction (Jason Wiese joined in 2016), and security.

5A.3 Undergraduate Degree Completion Data

The School graduated BS candidates in Computer Science in the years 2007–2011 at the following rates: 62%, 64%, 59%, 60%, 55%. (Note: these percentages show graduation rates 4 years after admittance to Full Major Status (typically after the first year). The 6 year graduation rates are about 4-5% higher, but the vast majority of students who graduate do so in 4 years (after full major status). On average, the University graduates students at a 60-65% rate. Thus, for most years the School has exceeded the University average.

It is important to note that CS has been in a *"boom, bust, boom"* cycle from 2000–2006 – 2016. The graduation rates for students admitted at various phases of this cycle (e.g., admit 2002–2007, graduate 2007–2011) correlate somewhat to the number and quality of students admitted. In 2006 we had our lowest ebb in the number of applicants to the major, and consequently, the criteria for admission were at their lowest (e.g. GPA). This may explain why the graduation rates from 2007 to 2011 have ticked downward (e.g. 68%, 69%, 63%, 64%, 57%). We expect for the percentages to increase as the quality of students have increased from 2006 to 2016. Even as we consider these trends, however, we have noticed that as the demand for CS students grows, there is a counter pressure pulling students in their 3rd or 4th year to take a full time (high paying) job and neglect finishing their degree.

Of the 30%-40% of students who do not graduate, approximately half of them are simply unable to complete the rigorous upper division course work (i.e., they fail required courses, ranging from software practice, to discrete math, to computer architecture, to calculus III, etc.). Note, this may be aggravated by demographic issues specific to Utah, where there is a high rate of working full time, having families, etc. Of the remaining non-graduates, some go directly to industry, some have encountered overwhelming medical conditions, a few switched degrees (a number to the Film side of our EAE program), and a few promising students just *disappeared*. We are in the process of identifying a representative group of these missing students and contacting them; preliminary results seem to indicate that in various ways, "life got in the way" (e.g., military duties, full time work to support a family, followed spouse to different city).

At this point it is not entirely clear what the School should/could do to increase graduation rates. Possibilities include lowering expectations (e.g., probation GPA requirement, course work requirements) to allow students with lower academic performance into a job market, which probably can absorb them. This is likely to cut against the prevailing culture and overall mission of the School, and therefore we are unlikely to pursue this approach. Another way to address retention might be to increase the ratio of teaching assistants/faculty to students (currently at 40 to 1, and 100–200+ to 1 respectively). There are efforts in the School to pursue this option. Our first approach will be to increase TA ratios in several of our challenging, required classes and try to discern what impact this has on outcomes. A third option is to provide more advising (currently 1.5 advisors handle 1000+ students). The School has recently hired a second graduate student advisor (to address the size and complexity of the graduate program) and will be looking into the possibility of another undergraduate student advisor.

5A.4 Employment

Economic and employment projections from the Bureau of Labor Statistics, released December 2015, show that Computer Science/Information Technology continues to be a high growth and high paying field. They predict a 23% overall growth between 2014 – 2024 in computer systems design and related services, a 26% overall growth between 2014 – 2024 in management, scientific, and technical consulting services. The overall size of growth in fields relating to computing generally outpaces other STEM disciplines (and, for example, all other engineering fields combined).

Computing Research Administration's Taulbee Survey shows a steady employment growth for new Ph.D's in industry since 2010, but a decline in academia during the same period. The below figure is from their 2014 survey. This is consistent with what we see in respect to what our graduates are doing. This data has impacted our thinking concerning curriculum. i.e. Seeing that there is a large demand in computation science and how that should impact our educational mission.

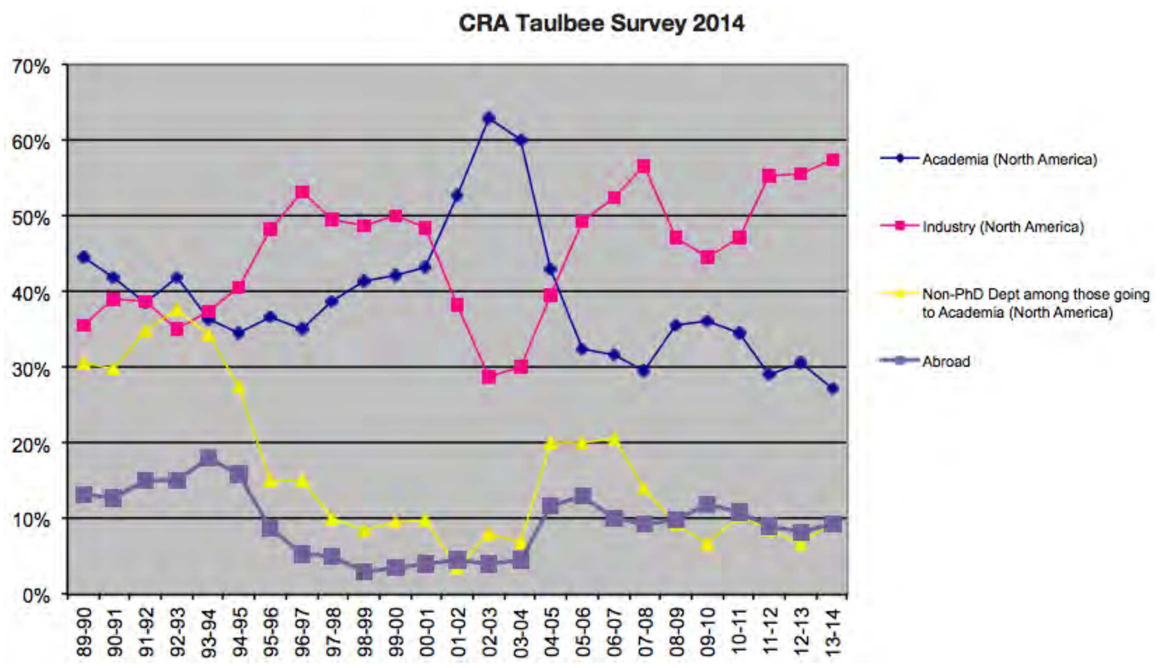


Figure 5A-2: CRA Taulbee Survey 2014

For Utah employment, Utah Technology Council's 2015 Hot STEM Occupations rankings put several computer science jobs in their top 15. With Several other computer science careers making their top 50.

- # 3: Software Developers, Applications with a growth projection of 4.1%
- #4: Network and Computer Systems Administrators at 3.9%
- #5: Computer Systems Analysts at 2.5%
- #11: Computer Programmers at 2.1%

- #15: Software Developers, System at 3.6%

The Career Services, at the University of Utah, has some limited data on their website. They offer salary wages for graduates they have contacted. Below is the data from their site.

	Degree	U of U Grad Mean	U of U Grad Median	# Rep	National Mean	National Median
Computer Science	Bachelor's	$69,000	$72,339	14	$66,801	$ 65,637

Table 5A-3: Career outlook - Information on the University's Career Service site

Lastly, Appendix Q provides a (partial) list of the School of Computing's graduate students who graduated in last 7 years (from 2009 to 2016) and their employment information (first and current employment if known).

5B. Graduate Program Effectiveness – Outcomes Assessment

5B.1 Graduate Outcomes Assessment Procedures

The learning objectives and outcomes for the different types of graduate degrees from the School of Computing are summarized as follows.

A. *Master of Science in Computer Science*

Learning Objective: The purpose of the MS degree in computer science is to expose students to cutting edge research in all areas of computer science, including topics in the foundations of computer science, hardware and software design, and computer and network systems.

Learning Outcomes:

- To demonstrate some mastery of a breadth of topics in computer science, and mastery of at least one topic if choosing the thesis/project option.
- For students choosing the thesis/project option: to be able to conduct independent investigations and advance the state-of-the-art in at least one area of computer science, in academic, industrial or government settings.
- To be experienced in leading and participating in research projects in computer science.
- To be able to adapt to and assimilate new developments in the state-of-the-art.
- For students choosing the thesis option: to be able to communicate advanced research concepts to both specialized as well as general audiences, in both written and oral form.

B. *Master of Science in Computing*

Learning Objective: The program in computing reflects an increasing interest in the tools of computer science as applied to a variety of application areas, and reflects the diverse strengths of the School across many disciplines that use computing. The purpose of the MS degree in computing is to expose students to cutting edge research in a specific (often multidisciplinary) area of computing, with this breadth reflected in their coursework and/or thesis/project work.

Learning Outcomes:

- To demonstrate some mastery of a specific focus area in computing, and additionally mastery of at least one subtopic in this area if choosing the project/thesis option.
- For students having chosen the thesis/project option: to be able to conduct independent investigations and advance the state-of-the-art in this focus area.

- To be experienced in leading and participating in research projects in this focus area.
- For students having chosen the thesis/project option: to be able to adapt to and assimilate new developments in the state-of-the-art, across the different disciplines that come together to form this area.
- For students having chosen the thesis option: to be able to communicate advanced research concepts to both specialized as well as general audiences, in both written and oral form, to audiences across the different disciplines that form this area.

C. *PhD in Computer Science*

Learning Objective: The purpose of the PhD degree in Computer Science is to train students to conduct original research in a specialized area of computer science, advancing the state of the art in a way that is novel, useful and substantial, and makes a lasting impact on the field of study.

Learning Outcomes:

- To demonstrate complete mastery of a chosen subfield of computer science with broad expertise across the areas of foundations, hardware and software design, and computer systems.
- To have conducted original published peer-reviewed research in their chosen area.
- To be able to articulate proposed work: both its motivation and a work plan.
- To have authored and defended a dissertation in their chosen area.
- To be able to communicate state-of-the-art research concepts to both specialized and broader audiences.
- To teach or assist in teaching computer science material to graduate/undergraduate students.
- To be prepared to conduct and lead independent research efforts in academia, industry or governmental organizations

D. *PhD in Computing*

Learning Objective: The program in computing reflects an increasing interest in the tools of computer science as applied to a variety of application areas, and reflects the diverse strengths of the School across many disciplines that use computing. The purpose of the PhD degree in Computing is to train students to conduct original research in a selected focus area straddling multiple disciplines within and outside computer science. This research will strengthen the links between the areas, forge new connections between them and advance the state of the art in a way that is novel, useful and substantial.

Learning Outcomes:

- To demonstrate complete mastery of a specific focus area in computing, with broad expertise across the different topics spanning multiple disciplines within and outside computer science.
- To have conducted original published peer-reviewed research in their chosen area.
- To be able to articulate proposed work: both its motivation and a work plan.
- To have authored and defended a dissertation in their chosen area.
- To be able to communicate state-of-the-art research concepts to both specialized and broader audiences across the different disciplines comprising their focus area.
- To teach or assist in teaching computer science material to graduate/undergraduate students.
- To be prepared to conduct and lead independent research efforts in academia, industry or governmental organizations.

The School implements a number of procedures to evaluate the effectiveness of its education and research programs for the graduate student population. Note that Graduate Sections 3B and 4B in this self study have already presented the details on the student graduation, degree awarded, and admission data. These procedures include but are not limited to the following processes:

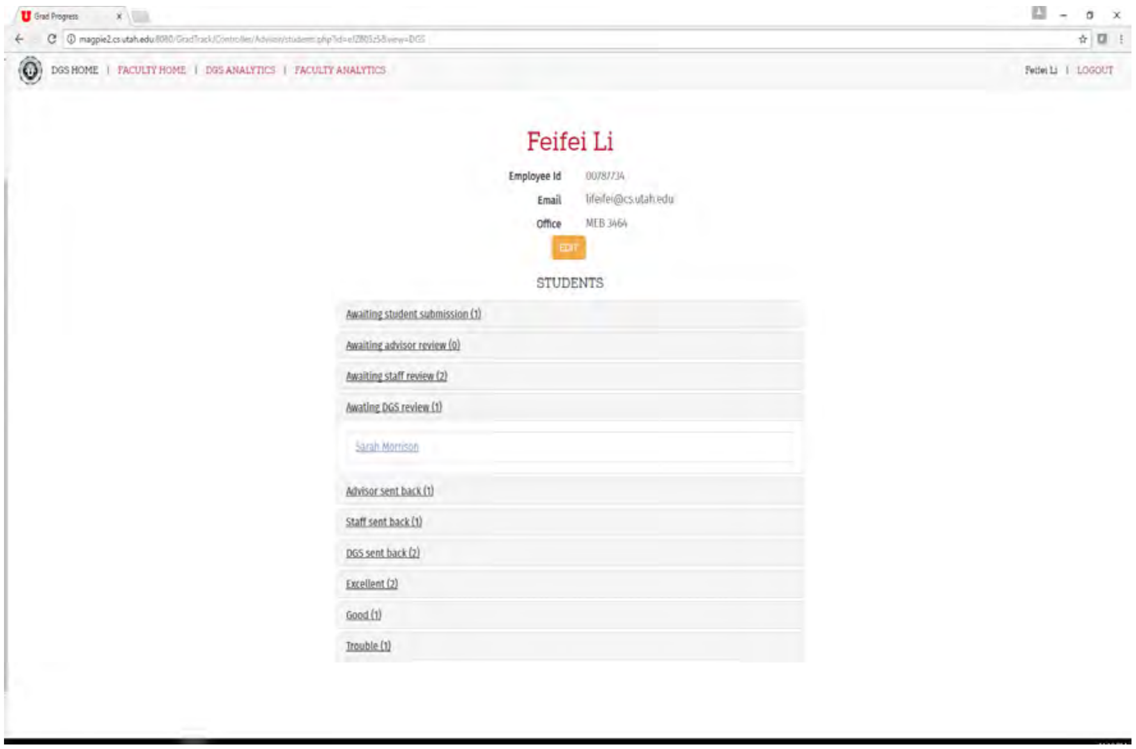
(a) *Mid-program assessments through due-progress forms and grad tracking.* All PhD students are required to complete a PhD due-progress form annually, which summaries the progress made by a student in the past academic year. The due-progress forms are evaluated by the DGS and the graduate student advisors to identify students who are struggling in their research and PhD study. A copy of the due-progress form is shown in Figure 4B-2 in Section 4B.3.

A faculty meeting is held every year at the end of the fall semester to review all cases that have been identified as having unsatisfactory performance, and a concrete plan and recommendation will be developed for each such case. Both the student and his/her advisor are informed about the plan and the recommendation made by the faculty, so that they can respond accordingly.

For master's students, the School keeps track of the courses they have taken and their academic performance through various tracks. Track directors are required to sign off each program of study form for all graduate students in their track, which provides an opportunity for track directors (as well as faculty members who serve on a student’s committee) to review the progress made by a student and provide feedback.

Recently, the School has developed a grad tracking system that enables faculty, students, and graduate student advisors to enter such data online and interact through a web-based system. Figure 5B-1 and Figure 5B-2 show the faculty interface to review a student’s progress form. In addition to the due progress form, the system also keeps track of funding support, internship program, and courses taken for a student.

The same system also offers rich analytics support, such as analyzing the average GPA per track, number of students supervised by each faculty, etc. An example is shown in Figure 5B-3. Lastly, the system is able to automatically identify students who have made excellent, acceptable, and unsatisfactory progresses respectively, through a combination of rule-based and mining-based approach.



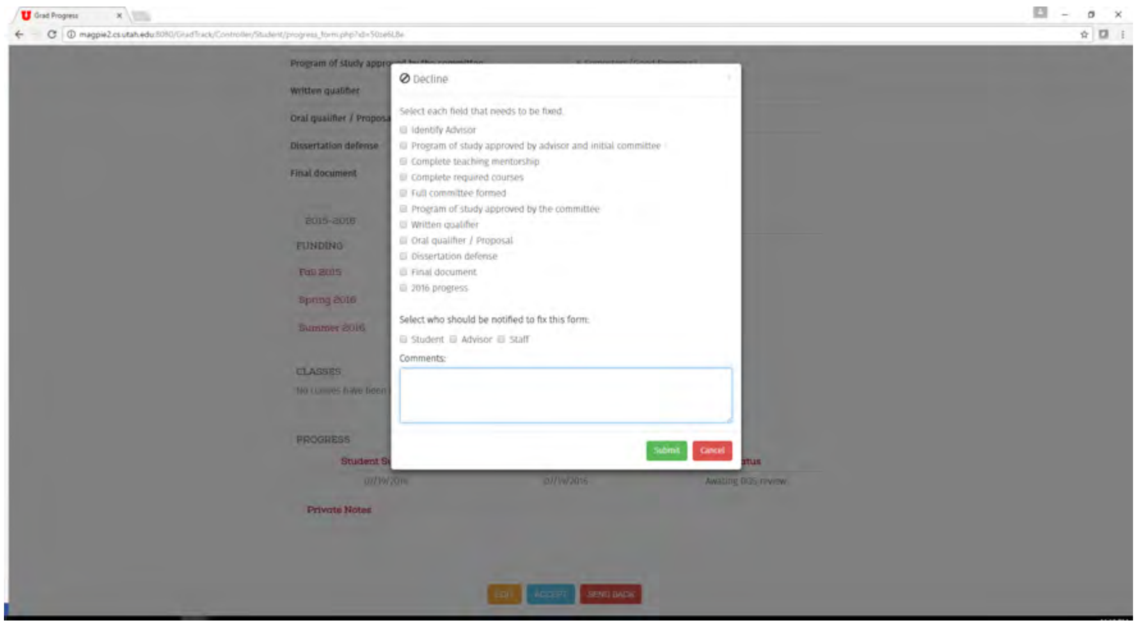
Feifei Li

Employee Id 00787734
Email lifeifei@cs.utah.edu
Office MEB 3464

STUDENTS

Awaiting student submission (1)
Awaiting advisor review (0)
Awaiting staff review (2)
Awating DGS review (1)
Sarah Morrison
Advisor sent back (1)
Staff sent back (1)
DGS sent back (2)
Excellent (2)
Good (1)
Trouble (1)

Figure 5B-1: Grad Tracking DGS Home View



Decline

Select each field that needs to be fixed.

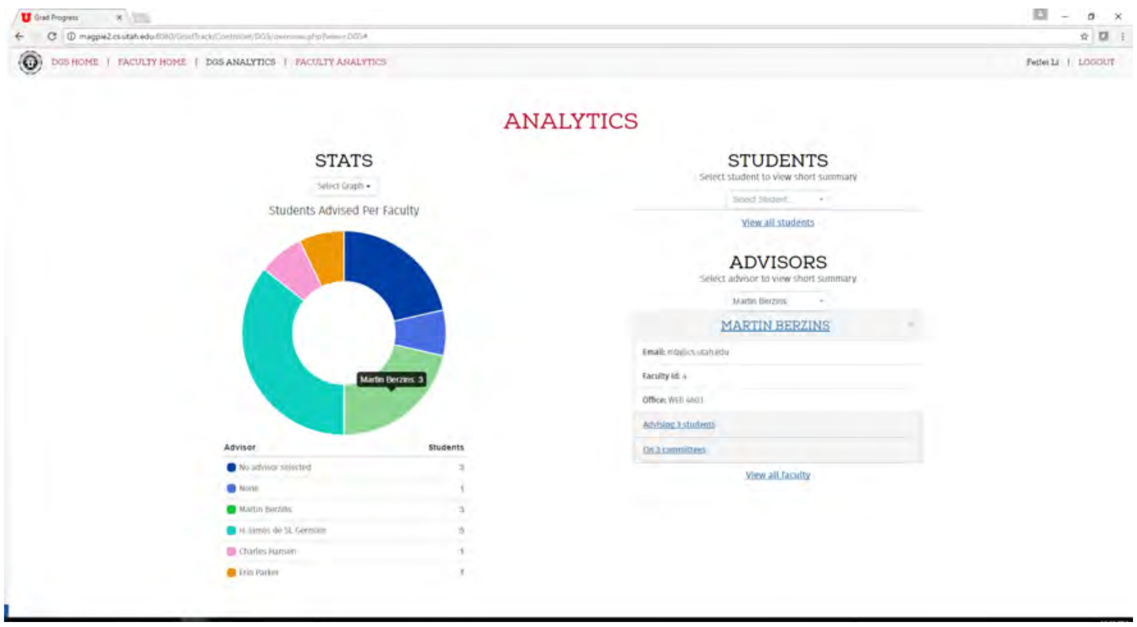
- Identify Advisor
- Program of study approved by advisor and initial committee
- Complete teaching mentorship
- Complete required courses
- Full committee formed
- Program of study approved by the committee
- Written qualifier
- Oral qualifier / Proposal
- Dissertation defense
- Final document
- 2016 progress

Select who should be notified to fix this form:

Student Advisor Staff

Comments:

Figure 5B-2: Grad Tracking Faculty/DGS Review Interface


Grad Progress
DGS HOME | FACULTY HOME | DGS ANALYTICS | FACULTY ANALYTICS
Feifei Li | LOGOUT
ANALYTICS
STATS
Select Graph
Students Advised Per Faculty
Martin Berzins: 3
STUDENTS
Select student to view short summary
Select Student...
View all students
ADVISORS
Select advisor to view short summary
Martin Berzins
View all faculty

Advisor	Students
● No advisor selected	3
● None	1
● Martin Berzins	3
● H. James de St. Germain	5
● Charles Hansen	1
● Erin Parker	1

MARTIN BERZINS

Email: mb@cs.utah.edu

Faculty Id: 4

Office: WEB 4603

Advising 3 students

On 3 committees

Figure 5B-3: Grad Tracking Faculty/DGS Analytics Interface

(b) *Internship Evaluation.* The School requires all students who participate in a summer internship program to complete an internship application and evaluation form. The form asks for a brief and concise learning objective for the internship program before students go for an internship, and requires a representative from the company to complete the form with actual learning objectives achieved after the completion of an internship program. A sample of this form is shown in Figure 4B-4 (for an international student).

(c) *Students and Peer Teaching Evaluation.* The School conducts a student evaluation (survey) for all its courses at the end of each semester. Overall, the School's teaching evaluation is consistently above the University and College average. On a typical course evaluation form, 6 questions are asked about the course effectiveness:

- Objectives clearly stated
- Objectives met
- Content well-organized
- Course materials helpful
- Assignments & exams covered the course
- Learned great deal

and another 6 questions are asked about the instructor effectiveness:

- Instructor was organized
- Instructor presented effectively

- Instructor created respectful environment
- Demonstrated thorough knowledge
- Instructor encouraged questions/ opinions
- Instructor available for student consultation

The response to each question is one of the followings: strongly disagree, disagree, mild disagree, mild agree, agree, and strongly agree. These responses are mapped to a weighted sum of numerical score in the range of 1 to 6 (1 being the worst and 6 being the best possible scores respectively).

The following table shows the average score of all School courses with respect to these 12 questions in last 7 years, and Table 5B-2 shows a more detailed breakdown for the score distribution.

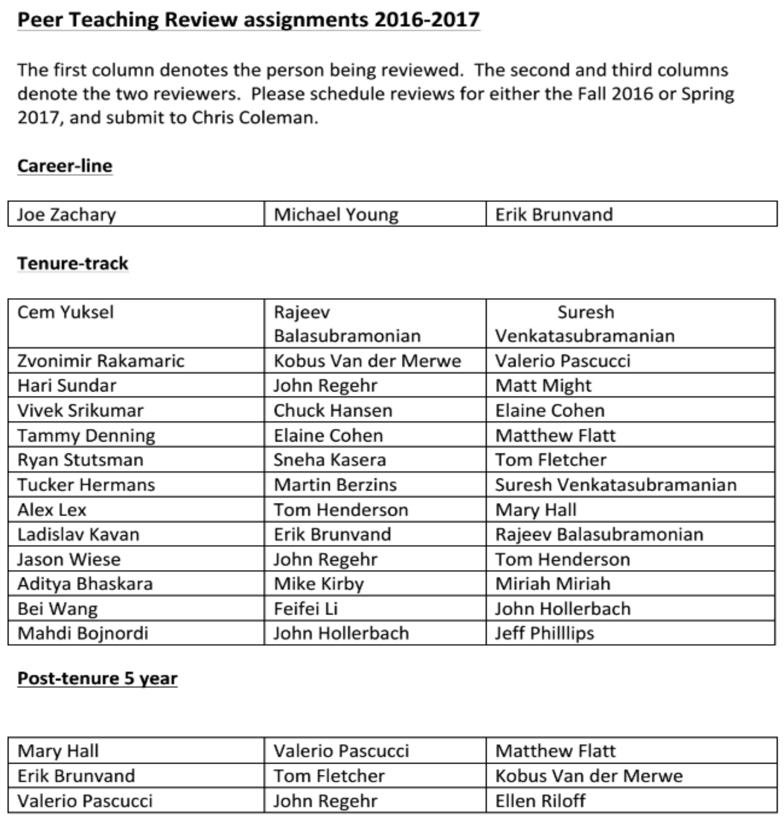
In addition to student course evaluations, the school also conducts peer teaching evaluations annually. An example of the peer teaching evaluation for academic year 2016-2017 is shown the following figure.

Year	Avg Score
2009	5.72
2010	5.65
2011	5.72
2012	5.66
2013	5.74
2014	5.71
2015	5.71
2016	5.76
Grand Total	5.71

Table 5B-1: Average student evaluation scores for the School's courses (on a scale of 1-6). College and University averages for graduate level classes are not available from the administration at this time.

Score Range	2009	2010	2011	2012	2013	2014	2015	2016	Grand Total	
<3.2 or (blank)			3	7	2		1	2	15	1%
3.2-3.5				1		1			2	0%
3.5-3.8			1			1	1		3	0%
3.8-4.1	2	1			1	2		1	7	0%
4.1-4.4	1	2	1	6		2		2	14	1%
4.4-4.7	1	2	1	1	1	2	3		11	1%
4.7-5	2	2	3	1	2	3	1	7	21	2%
5-5.3	10	17	14	19	12	12	18	7	109	9%
5.3-5.6	18	20	23	31	28	26	29	12	187	16%
5.6-5.9	16	20	27	19	29	20	36	13	180	15%
>5.9	67	54	70	86	89	88	91	73	618	53%
Grand Total	117	118	143	172	165	155	181	116	1167	

Table 5B-2: Student Evaluation Score Distribution for the School's Courses (on a scale of 1-6)



Peer Teaching Review assignments 2016-2017

The first column denotes the person being reviewed. The second and third columns denote the two reviewers. Please schedule reviews for either the Fall 2016 or Spring 2017, and submit to Chris Coleman.

Career-line

Joe Zachary	Michael Young	Erik Brunvand

Tenure-track

Cem Yuksel	Rajeev Balasubramonian	Suresh Venkatasubramanian
Zvonimir Rakamaric	Kobus Van der Merwe	Valerio Pascucci
Hari Sundar	John Regehr	Matt Might
Vivek Srikumar	Chuck Hansen	Elaine Cohen
Tammy Denning	Elaine Cohen	Matthew Flatt
Ryan Stutsman	Sneha Kasera	Tom Fletcher
Tucker Hermans	Martin Berzins	Suresh Venkatasubramanian
Alex Lex	Tom Henderson	Mary Hall
Ladislav Kavan	Erik Brunvand	Rajeev Balasubramonian
Jason Wiese	John Regehr	Tom Henderson
Aditya Bhaskara	Mike Kirby	Miriah Miriah
Bei Wang	Feifei Li	John Hollerbach
Mahdi Bojnordi	John Hollerbach	Jeff Philllips

Post-tenure 5 year

Mary Hall	Valerio Pascucci	Matthew Flatt
Erik Brunvand	Tom Fletcher	Kobus Van der Merwe
Valerio Pascucci	John Regehr	Ellen Riloff

Figure 5B-4: Peer-teaching evaluation assignment

(d) *Industrial Advisory Board.* The School organizes an industrial advisory board meeting every year with the members on its industrial advisory board, which consists of leaders and directors from local IT and CS industry. They will review the School's education and research program and offer detailed feedback and suggestion as how to improve and better serve the local industry needs.

(e) *Exit Survey.* The school admins an exit survey procedure for graduating graduate students, but it is not yet required of students. Thus, the data is very sparse and is not very yet able to provide quantitative insights into the program. However, the School has the plan to integrate the exit survey into the grad tracking system so that this data collection step becomes much more effective.

(f) *Alumni Interactions.* Similarly, the School maintains a website for connecting to alumni, but the response rate is not high enough to serve as an effective tool to connect with the school's large alumni population. The School also has a Facebook page that does attract a large number of alumni and current and future students. The School has conducted alumni survey to understand its alumni population better and reconnect with them. The survey was done through emailing the School's alumni with a link to a google form.

5B.2 Graduate Outcomes Assessment Feedback

The School is constantly looking for ways to solicit outcomes assessment feedback and taking these inputs seriously into the design and improvement of the School's education and research activities. Given the extremely fast-paced computer science discipline, and the complex, dynamic, and extremely wide-range of applications found in computer science (or computing), rather than sticking with the same set of mechanisms in collecting outcome assessment feedback, the School has adopted a very flexible and broadly-defined approach. More specifically, the School has mainly relied on the following channels for collecting outcomes assessment feedback and program improvement suggestions:

- Industrial Advisory Board (IAB):
  The annual IAB meeting provides a great opportunity for the school to learn the needs and demands from local industry leaders, as well as receiving feedback from them on how to improve our programs. Many school of computing's graduate students join the local industry force after they have graduated from our program. As a result, engaging the representatives from local industry is a great resource for collecting valuable feedbacks.

- Students' feedback:
  The school of computing runs a GradSAC committee (there is also an UGSAC committee for the undergraduate population). Members of the GradSAC committee are selected from the school's current graduate student population. They represent a bridge between the student body and the school, and constantly provide useful feedbacks they have received and collected from our students. For example, the school asks for the votes from the GradSAC on important issues such as faculty hiring, and RPT cases. Feedback from students also comes to the faculty directly through MS/PhD advisors and supervisory committees.

- Graduate School from the College and the University:
  Both the College of Engineering and the University of Utah's Graduate School hold regular DGS (Director of Graduate Studies) meetings for all units within the college and the university respectively. The DGS from the School of Computing attends these meetings monthly and reports issues and plans from the school, and exchange ideas with other DGS from other departments. These meetings also offer a direct channel between the college and the university's graduate school offices and the school to discuss ways to improve the various research and education programs in the school.

There are many instances where outcomes assessment feedback collected through one or a combination of these channels have inspired changes and improvements to the school's existing programs, or led to the creation of new programs. This section will describe three such examples from the school's graduate program in the interest of space.

1) The reform of the "Data Management and Analysis Track":
   In 2011, given the feedback from both IAB and our students that knowledge and skills in machine learning, data mining, and information visualization are increasingly important

for various data analysis tasks, faculty members from the then *Information Management Track* embarked on a major reform of the track, which had resulted in a new track called *Data Management and Analysis*, which requires the following courses:

- Data Mining or Machine Learning
- Advanced Database Systems
- Advanced Algorithms
- Information Visualization

This reform is proven to be a huge success, and the number of students enrolled in the new data track continue to rise.

2) The creation of the *Big Data Certificate Program*:
With the increasing interest and demands in big data applications, companies and various agencies and institutes in the state of Utah have a constant and growing demand for a well-trained workforce with the experience and skills to help in the management and analysis of *big data*. In light of this, and with the feedback the School has received through its IAB meetings, alumni networks, meetings with various industry partners, and surveys, the school realized that there is a huge demand for a formalized program (and credential) that is accessible to a wide range of student as well as working professionals.

   The observation is that many working professionals are extremely interested in the school's data management and analysis track, but unfortunately they are not able to complete the track requirements due to their working schedules. Hence, based on the core curriculum of the data track, the school developed the *Big Data Certificate Program* that essentially requires only the four core courses from the data track. The details of this certificate program is available from http://www.cs.utah.edu/bigdata/.

   Furthermore, after many round of active discussions and exchange of ideas with local companies such as Adobe, Domo, and others, to better facilitate working professionals to enroll into the big data certificate program, the certificate program has established a YouTube Channel called UofU Data, available at
https://www.youtube.com/channel/UCDUS80bdunpmvWVPyFRPqFQ,
that offers both live streaming videos of all courses from the program and archive channels with videos for all past lectures for courses offered by the program.

3) The creation of the *Certificate in Data Center Engineering*:
In a similar fashion to the process described above, when NSA and other companies set up their data centers in Utah, the school proactively approached them and solicited feedback and demands from their leaders and work force, which led to the creation of the *Certificate in Data Center Engineering*.

5B.3 Graduate Degree Completion Data

Please refer to Table 4B-1 and Figure 4B-1 for admission and degree completion data for the last 7 years respectively.

It is difficult to obtain an exact estimate of the attrition data per year; even though an MS typically completes his/her program of study in two years (four semesters), and a PhD student normally takes five years to complete his/her program of study, students may take different number of semesters to complete his/her program, making it hard to estimate the attribution rate based on the admission data and the degree awarded data from the following 2-years and 5-years for the master's and the PhD programs respectively.

That said, Table 4B-1 and Figure 4B-1 indicate that a large majority of our graduate students have successfully graduated from their respective programs.

5B.4 Employment

This information can be found in Undergraduate Section 5A.4.

6. Facilities and Resources

6.1. Operating Budget

The School of Computing serves two missions: academic and research. To fulfill our two missions, we maintain a dual accounting structure: a zero-sum fund consisting of base funds that come to us from the State towards maintaining our academic mission, which is augmented by the results of our teaching productivity (SCH and engineering differential funds) and degree production, and returned F&A (overhead) funds, which we are allowed to accumulate and whose increase represents a percentage of our yearly research income. Expenditures related to our academic mission, such as faculty salaries, academic staff (advisors, etc.) and TA funding, are charged against the first of these funds. Expenditures related to our research mission, such as our research computing facility, funding research start-up funds of new faculty, and staff allocations associated with our research mission, are charged against the second of these funds.

6.1.1. Budget to meet our academic mission

The academic departmental budget for the new fiscal year typically consists of the base budget from the previous year plus any price levels increases approved by the Utah State Legislature. In addition, departments have received supplemental funding that is based on changes in SCH productivity (student credit hours taught by each unit), labeled *productivity funding*. This funding can increase or decrease depending on the population of students within the courses. When the productivity funding has been consistent from year to year, departments may be allowed, with appropriate permission, to "harden" some of these dollars into the base budget lines, which can be used to pay for long-term commitments (such as faculty salaries). The overall contribution for the current year is $435K along with $24K for degree production. The College also generates differential tuition for engineering degrees. This is a supplemental charge (instituted in 2009-2010 following a severe budget cut of 19%) consists of a $52 per credit hour for 3000-5999 level classes and $74 per credit hour for 6000 level and above credit classes. The overall contribution to the budget is approximately $769 K for 2016-2017. See Figure 6-1, which represents our State Appropriated Funds. In 2012, the administration added degree production as a form of incentive funding.

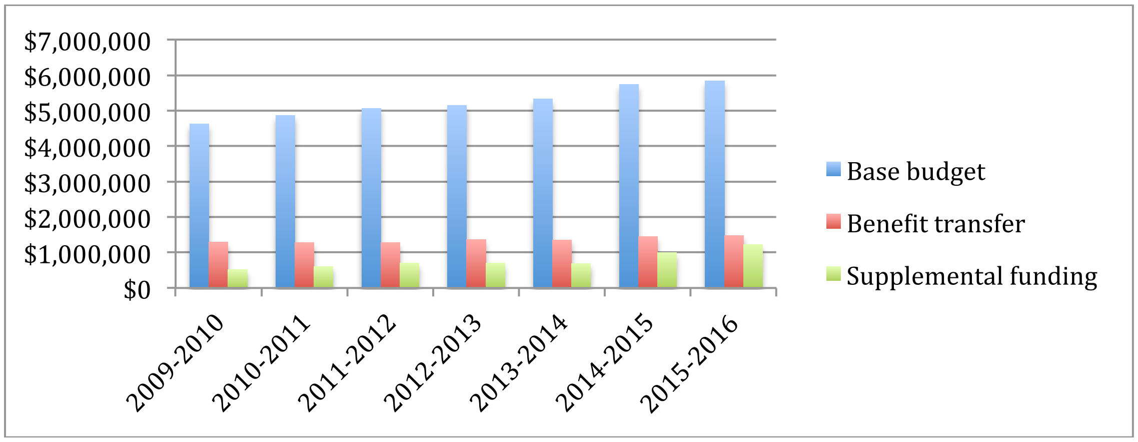


Figure 6-1: State appropriated budgets

In addition to the standard inputs of productivity and differential to our yearly academic operating budget, the College and in particular the School has also been supported through a number of statewide Engineering Initiatives (a program initiated a number of years ago with the strong support of local industries, with the goal of increasing the number of engineering graduates in Utah). This program has provided both on-going and one-time funding in support of the engineering programs in the State of Utah. As part of the requirements for receiving these special funds, the recipient institution is required to provide institutional matching funds to the on-going funds provided by the legislature, making this a true partnership. In recent years, the legislature has established the Technology Industrial Advisory Board (TIAB), made up of key players in the local economy, who oversee the allocation of the Engineering Initiative Funds. Competitive proposals are submitted to the TIAB, and funds received by the institution from the TIAB, along with the institutional matching funds, are then allocated by the Dean to the departments based on the number of students in each program, relative growth, and other potential needs identified by the department director and Dean. The School has received from these Engineering Initiatives. In 2012 we received $250,000 to assist with our recruitment of new faculty and in 2015 we received $770,00 to assist in funding our increase in growth. These funds were used to fund new faculty recruits at both the tenure-line and career-line level, to enable us to hire an additional graduate advisor, and supplemented our TA funding.

In 2005 the Universities across the State of Utah have benefited from the Utah Science and Technology Research (USTAR) Initiative program. This program was created to stimulate economic development activities by enhancing our ability to "recruit world-class researchers to Utah." The School has benefited from this program with the addition of four faculty positions over the past seven years. If and when faculty give up these positions (e.g. leaving the University), the funds go back to the USTAR Authority, and thus these faculty "lines" are with the School as long as the faculty remain in their positions. For example, one USTAR faculty member, Guido Gerig, has left the University.

Some of the faculty in the School (approximately nine) are not paid from the School budget, but are counted as regular faculty, for the purposes of appointments, promotion, service, teaching loads, research expenditures, head counts, etc. These are faculty associated with the Scientific Computing and Imaging Institute. These faculty are hired in a process that is separate from the School (but with School members involvement, and final approval/appointed by the School faculty), and are chosen to suite the strategic goals of the SCI Institute. Their offices and research programs are also housed in space that is allocated by the Director of the SCI Institute (Chris Johnson).

6.1.2. Budget to meet our research mission

The second component of the departmental budget, related to our research mission, is our F&A based research funding. These funds are allowed to accumulate over short periods of time to permit strategic investments that are in the best interest of the School, such as the recent bootstrapping of our Fellowship Program. Returned Overhead funds provided by research funding are used for new faculty start-up packages, facilities renovation, our departmental fellowship program, and to support other research development activities. With the addition of new faculty there has been a steady growth in research funding over the past seven years, which has provided a steady increase in our Returned Overhead production.

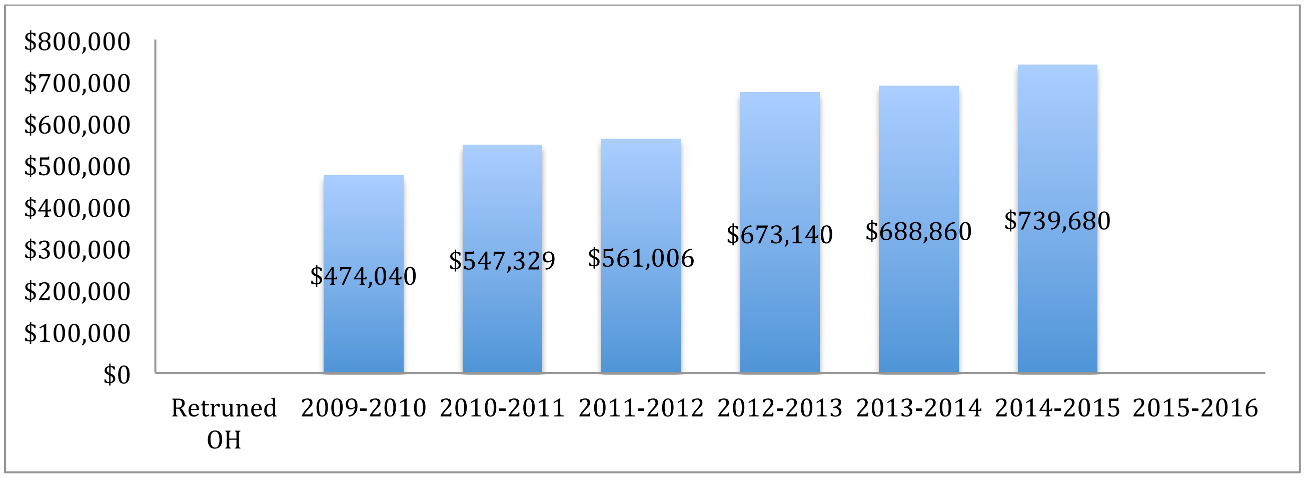


Figure 6-2: Returned Overhead

Gifts and donations are a relatively small part of the overall budget, and are used primarily for scholarships, outreach and facility additions, and other program improvements.

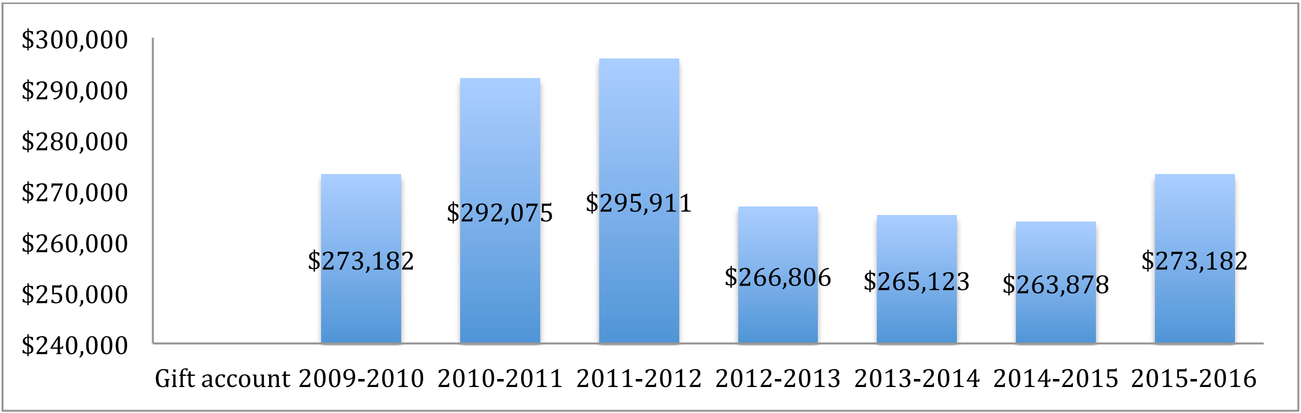


Figure 6-3: Gift funds received each year

6.1.3. Adequacy of budget

The School has seen an increase in our student population over the past seven years.

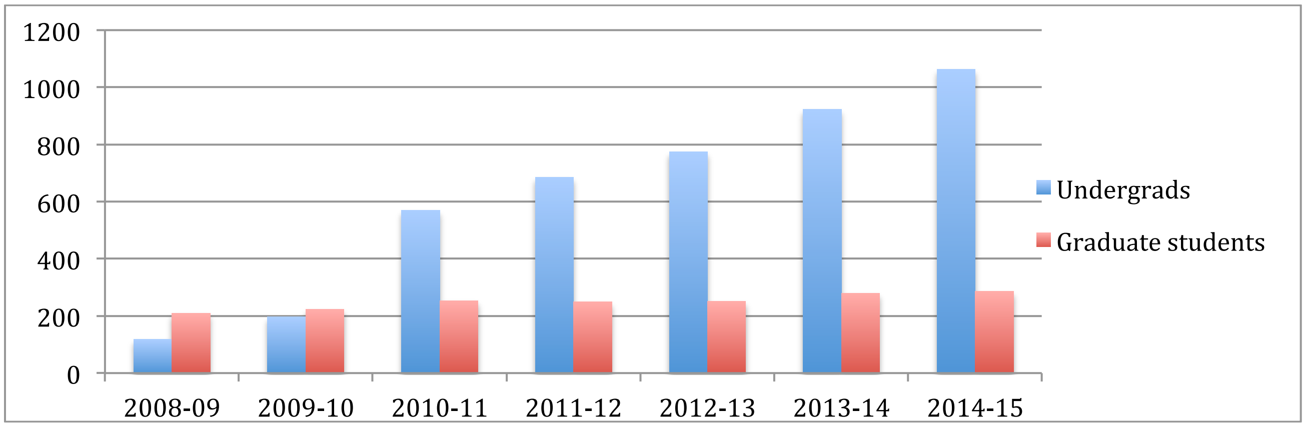


Figure 6-4: Growth in student population

To accommodate this increase in growth we have seen an increase in faculty positions through recent Engineering Initiatives, and USTAR (a total of approximately seven positions). The addition of differential tuition and the ability to use productivity dollars has helped to bridge some of the gap in the student /faculty ratio. The increase in the faculty has provided an increase in the number of grants obtained and an overall increase in research expenditures, funding graduate students.

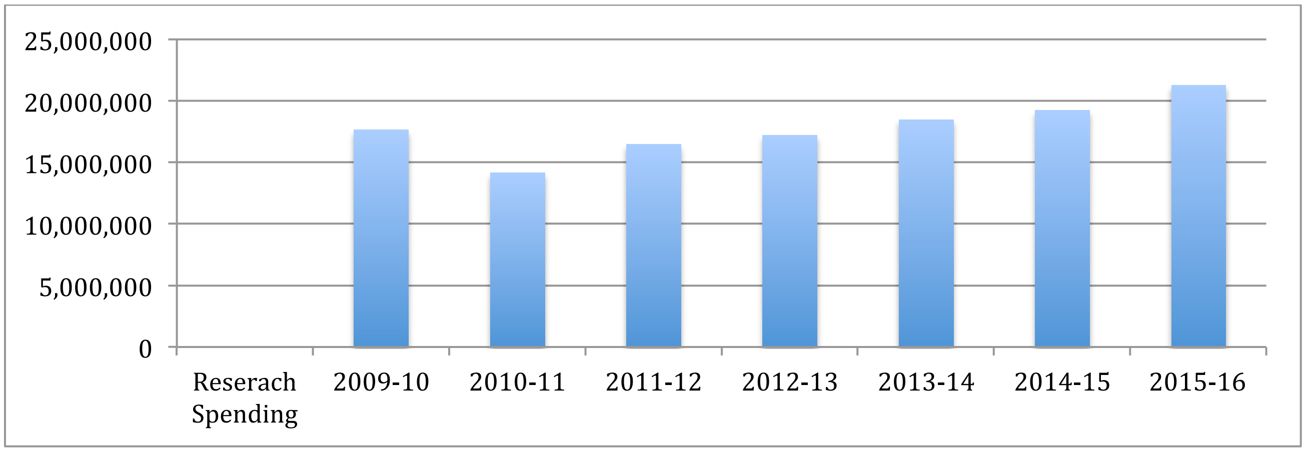


Figure 6-5: Research expenditures per year

To accommodate the need for specialized equipment the University has provided BEEF (Base Engineering Equipment Fund), and until 2014 support was provided by SCAC (Student Computing Advisory Committee) to help fund student computer laboratories. Over the past seven years we have received a total of $684,636 additional funds from these programs. Such funding has provided assistance with upgrading of labs and instructional equipment.

6.2. Physical Facilities

The physical facilities available to the School consist primarily of research and communications infrastructure (discussed in subsequent sections) and physical space. Here we discuss physical space. Space available to the School consists of two parts, one part is the space used by faculty who are associated with the SCI Institute, which is primarily in the top three floors of the Warnock Engineering Building (WEB). This space is allocated by the University President (David Pershing) and administered by the Director of the SCI Institute (Chris Johnson) to accommodate the mission of that Institute, and affects the School's needs only to the extent that faculty housed in WEB are (generally) not needing space resources from the College. The remaining 30+ faculty (and their associated research programs) as well as administration of the School and specialized, dedicated CS classrooms are housed in the Merrill Engineering Building (MEB), using space that is allocated by the Dean of Engineering (Richard Brown) and administered by the Director's Office. That space is broken down by its type (room size and location) and current use with the School (MEB) in Table 6-1.

Current space allocations represent a limiting factor in the productivity and growth of the School. This is particularly true for faculty/staff offices and graduate research laboratories. For instance, offices are approximately 180-200 square feet each, and the current allocation is approximately 40 offices, which is not enough to house all of our current tenure-track and full-time, career-line faculty (not to mention visitors, etc). This problem is made worse by plans to grow in the near future (three open positions advertised for fall 2017). Currently, the School deals with this issue by borrowing space from other units (e.g. SCI has generously lent, on a temporary basis, to the School space over in MEB) and borrowing offices from faculty who are on sabbatical. Normally, we allocate smaller, interior offices to post doctoral fellows and technical staff, but there are not enough of these available to consistently make space for hires associated with new projects (of which there are many). Because of this, the allocation of offices is a week-to-week affair, in which new people are sometimes waiting for offices to open up.

A similar situation exists for research space, which is primarily used to house graduate students. There are approximately 125 paid graduate students (RAs) and another 25 fellowships students, all of whom would typically need dedicated desks at which to do their work. The current allocation of 14,000 square feet provides 90 square feet per student. At first glance this seems adequate, but, in practice it is restrictive. There are several reasons for this. First is that much of this space is not appropriately formatted. That is, while most of it is in the form of rooms bigger than 1000 square feet, in which 90 per person will generally work, much of it is smaller rooms (less than 500 square feet), where walls, doors, and other constraints limit full utilization. Another problem is that some of this space is either substandard or inadequate for a variety of reasons. For instance, some rooms suffer from excessive noise due to building infrastructure (e.g. vents from other floors) or from poor climate control. Other rooms suffer from esthetics (old, appearance/smell) that make them difficulty to utilize effectively. These two reasons contribute to a third reason, which is that while the School (with generous help from the College) renovates older, underutilized spaces in MEB, it needs a buffer of space to make temporary moves. These issues together have created a situation which threatens the well being of the School.

For instance, during recruitment, some new faculty expect dedicated space for their research groups (e.g. because their advisors had it or they were offered such by competing universities). This is very difficult (and in some cases, impossible) to accommodate. Likewise, too little space or poor quality space is a quality of life issue for graduate students (several new faculty have complained), which undermines our recruitment of top students. Thus, while departments at other universities (and even within our own College) have invested in new facilities and are able to offer reasonable amounts of high-quality space to students and faculty, the School of Computing is currently unable to do so. This puts the School at a competitive disadvantage relative to its peers.

In the short term, the School is borrowing space from other units. There are agreements with the SCI Institute (for space in WEB) and the Department of Mechanical Engineering (for space in MEB) for them to lend the School space on a temporary basis.

Space/Type	Approx. Sq. Ft.
Classrooms (CS specialty)	7,500
Research labs	14,000
Faculty offices	8,000
Conference rooms	4,000
Individual staff/research offices	3000
General administration	3500
Mechanical	<500
Storage	2000

Table 6-1: Space allocated (used) by the School in the Merrill Engineering building by type/usage.

6.3. Libraries

We have a small departmental library for textbooks. For technical information, we have on-line access to journals and conferences which is available through the IEEE and ACM interfaces offered by the Marriott Library.

6.4. Centers, Institutes or Bureaus Associated with the Program

There are several institutions on campus that directly interact with the operations of the School of Computing. First is the Scientific Computing and Imaging (SCI) Institute. The SCI Institute (formed in approximately 2000, out of a research group from within the School) is one of eight designated *Research Institutes* at the University of Utah and home to over 200 faculty, students, and staff. The 15 tenure-line faculty are drawn primarily from the School of Computing (9), Department of Bioengineering, Department of Mathematics, and Department of Electrical and Computer Engineering, and man of those faculty have adjunct appointments in other, largely medical, departments. Over the past decade and a half, the SCI Institute has established itself as an internationally recognized leader in visualization, scientific computing, and image analysis applied to a broad range of application domains. The overarching research objective is to conduct application-driven research in the creation of new scientific computing techniques, tools, and systems. An important application focus of the Institute continues to be biomedicine, however, SCI Institute researchers also address challenging computational problems in a variety of application domains such as manufacturing, defense, and energy. SCI Institute research interests generally fall into the areas of: scientific visualization, scientific computing and numerics, image processing and analysis, and scientific software environments. SCI Institute researchers also apply many of the above computational techniques within their own particular scientific and engineering sub-specialties, such as fluid dynamics, biomechanics, electrophysiology, bioelectric fields, parallel computing, inverse problems, and neuroimaging.

The SCI Institute is housed in a set aside space in the Warnock Engineering Building, which is managed/allocated by the Director of the SCI Institute, who reports to the president of the University. Nine School of Computing faculty share that space (approximately 27,000 square feet) with 5 other faculty. The SCI Institute houses and manages a set of computational resource (see Section 6.5 below) that are available to SCI faculty, as well as other School faculty (upon request and as determined to be appropriate). The SCI Institute has a base budget, with allocations for tenure-line faculty, that comes from the President's office and other, more specialized resources (such as USTAR and the Clusters for Transformative Excellence—out of the Vice President's office). The SCI Institute plans those hire in accordance with its strategic goals and mission, conducts those hiring processes (with input from School faculty, where appropriate), and then seeks appointments from the School. SCI faculty with School appointments have offices and conduct research in the allocated SCI space, but participate toward (and count toward) School faculty in virtually every other way.

The School also interacts with the Entertainment Arts and Engineering (EAE) Program, which is a designated *Instructional Program* at the University, which grew out of activities within the School. The EAE Program, lead by Prof. Bob Kessler (an School faculty member, until recently, now full time with EAE), teaches undergraduate classes in technical areas relating to *computer gaming*, and these classes form the basis for undergraduate emphasis areas in EAE, which are available to students seeking BS degrees in CS and BA degrees in Film. Thus, these undergraduates are exposed to an interdisciplinary experience in video games, that complements their primary degree, and for which they get a certificate. The EAE program also offers a professional master's degree in EAE (which grew out of the MS in Computing within the School). The EAE program has also recently proposed a stand-alone BS in Games (BSG). The School interacts with EAE on the undergraduate EAE certificate as well as joint tenure-track hires that have appointments in the School. Most recently, EAE and School were successful in the joint hire of Michael Young from NC State. An MOU between EAE and School describes the parameters and procedures for such joint hires.

The School also interacts with the Center For High Performance Computing (as described below). The CHPC is an independently run unit on campus that provides access to high performance computing infrastructure. Several of our faculty have access to this infrastructure and use these resources to conduct experiments or teach classes. Recently, the School agreed to subsidize a joint faculty purchase of a set of dedicated nodes within this CHPC infrastructure.

6.5. Technology

Computing support for the School's educational mission is provided by the College of Engineering. The College has a large support staff and maintains seven student labs: two with Linux machines, four with Windows, and one with OS X. These labs can be accessed remotely and are physically available to students 24/7/365. Printers are available.

Computing support for the School's research mission and front office is provided by four facility staff. Some faculty members, graduate students, and research staff opt to be their own system administrators, but most rely on facility-supported machines. Computers used by faculty, staff, and graduate students have access to a shared file server, email server, and printers. Additionally, the SCI Institute has three facility staff and three media developers.

Several School faculty members maintain small clusters that are administered by the facility staff. The SCI Institute has several clusters totaling more than 800 cores. Additionally, in September 2016 a small cluster (168 cores) was brought online at the Center for High Performance Computing, which provides dedicated HPC support to the University. This cluster is available to all School faculty and students.

The Flux Research Group in the School has developed and continues to operate multiple network testbeds that support research and education activities in computer science and computing. These facilities include Emulab (2,592 cores), Apt (2,048 cores), CloudLab (geodistributed at three sites, with 4,680 cores in Utah cluster), the InstaGENI-DDC rack (528 cores), and PhantomNet (88 cores, including those in actual mobile phones and basestations). These facilities are Internet-accessible, federated, used 24/7 by people at Utah and around the world, and support literally thousands of experiments per year.

The Flux Research Group expects to enhance and operate these testbed facilities throughout the next five years, based on current and anticipated funding, with enhancements focused on CloudLab (for cloud-infrastructure activities), PhantomNet (for mobile 4G/5G networks), and new city-scale infrastructure to be determined and supported by the NSF's recently announced PAWR program.

In general, School faculty, staff, and students have access to high-quality facilities and administrative support; we plan to continue making these available.

6.5.1. School of Computing's Facilities

The School provides a state-of-the-art computing facility for both educational and research use. Facilities to meet both our educational and research missions shares a common network infrastructure that is based on 10+Gbps (gigabit per second) core that provides desktop connections with 1 Gbps ethernet. The School's network attaches via redundant 10 Gbps connections to the campus backbone routed via OSPF.

The campus backbone runs at 40+Gbps with a 100 Gbps research DMZ. The campus attaches via multiple 10 Gbps links to the Utah Education Network (UEN) which provides both commodity Internet and research connectivity. UEN maintains multiple gigabits of commodity from various carriers at strategic points throughout the state. For research connectivity, UEN connects at 100 Gbps to the Internet2, and approximately 30 Gbps aggregate to the commodity Internet.

In addition to the shared network infrastructure, the core School facility supplies many centralized services, including shared disk space (200 Terabytes), time, web/cgi/php, s/ftp, firewall, backups, printing resources, authentication (AD/LDAP/NIS), vpn, ssh/interactive servers, doorlock access, and email. The core of the server infrastructure runs on VMware's Enterprise vSphere virtualization product. Most services run on VM-Linux-based hosts, with some additional services being served from Windows and Solaris machines.

The instructional computing facility includes over 300 Unix, Linux, and Windows-based machines. Most of these machines are organized into three laboratories with the remainder being situated in graduate student offices. The Undergraduate Lab in EMCB 130 includes approximately 90 3200+ gigahertz Athlon-based PCs with 1G of RAM, running Windows XP, and the lab in EMCB 124 has 24 3500+ AMD64 systems with 2G of RAM. The electronic classroom in MEB 3225 contains 30 Pentium-based PCs arranged into a classroom configuration. The CES/Grad Lab in MEB 3161 contains 15 PCs based on Athlon 3200+ processors video. Complementing these labs are several specialized resources dedicated to academic instruction, including a suite of machines made available for student administration, several specialized hardware and software labs, and a 32-node Linux cluster.

The research computing facility is a heterogeneous mix of over 250 machines, including PC's and Sun-based hardware. The research computing facility includes major laboratories devoted to computer-aided design and graphics, computer systems, asynchronous digital systems and VLSI, robotics and vision, scientific computing and imaging, and information retrieval and natural language processing. These research laboratories contain a wide array of specialized equipment, including:

- a 600-node network testbed and emulation facility;
- a multi-source nonlinear video editing environment;
- Several Linux clusters, including a 32-node dual-Xeon 2.8G cluster
- a real-time signal processing lab;
- an image analysis lab;
- equipment for various types of custom hardware design;
- a Sarcos Dextrous Arm, Utah/MIT Dextrous Hand, and PUMA 560 robots;
- a Sarcos Treadport locomotion interface, several SensAble Phantom haptic interfaces, Fakespace Responsive Workbench, nVision Datavisor HiRes, and a variety of position trackers; and
- various 3D-printing facilities.

The College operates a research-scale integrated circuit (IC) fabrication facility that is used extensively by the School. Equipment for testing and debugging both internally and externally fabricated circuits is housed in an integrated circuit testing facility that contains state-of-the-art HP, Tektronix and Micromanipulator automated IC testing equipment.

6.5.2. Scientific Computing and Imaging Institute (SCI) Facilities

The SCI Institute computing facility, which has dedicated machine room space in the Warnock Engineering Building, includes: Shared memory multi-processor computers, clusters and dedicated graphics systems.

1. 264 core, 2.8TB shared memory SGI UV 1000 system with Intel X7542 2.67GHz Processors

2. 64 node GP-GPU cluster. Each node has 8 cores, 24GB of RAM, with Intel X5550 2.66GHz processors each node is connected to (32) NVIDIA s1070 Tesla GPU systems nodes are linked with a 4x DDR Infiniband backbone with dual 10G network connections to SCI core switches.
3. 32 node GP-GPU cluster. Each node has 16 cores, 64GB of RAM with Intel E5-2660 2.20GHz processors. Each node has 2x Nvidia k20 GPUs with 2 full speed FDR Infiniband connections. ystem has a total of 128 56Gb/s Infiniband connections.
4. 32 core, 192GB shared memory IBM Linux system with Intel Xeon X7350 3.0GHz processors This system can also be reconfigured into two separate 16 core systems with 96GB of RAM
5. 64 core, 512GB shared memory HP DL980 G7 with Intel Xeon X7560 2.27GHz processors
6. 4x 80 core, 842GB shared memory HP DL980 G7 with Intel E7- 4870 2.40GHz processors
7. (3) 8 processor (24 cores, 2.5GHz, AMD Opteron with Nvidia Quadro FX 5600 graphics card) with a dual Gigabit Ethernet backbone and 96GB RAM
8. 8 core, 2.0GHz, AMD Opteron, with Nvidia Quadro (2FX graphics card) with a dual Gigabit Ethernet backbone and 16GB RAM
9. 6 core Intel Xeon x5650 2.67GHz with 196GB of RAM and 2x c2070 GPUs
10. 8 core Intel Xeon x5570 2.93GHz (16 with HT enable) with 126GB of RAM and c2050 / c2070 GPUs
11. 12 core Intel Xeon E5-2640 2.50GHz with 32GB of RAM and 3x K20c GPUs

In addition, the SCI Institute computing facility contains:

1. An Isilon storage cluster with 13 36NL 36TB storage nodes for a total of 422TB usable space with 6x dual 10 Gigabit Ethernet links and significant expansion capacity (up to 1PB single namespace)
2. Dedicated IBM backup server to manage SAN backup system and SL500 robots.
3. IBM 10TB LTO-4 tape library providing backup for infrastructure systems such as email, web, DNS, and administrative systems
4. 500TB LTO-4 StorageTek SL500 tape library primary backup system
5. 2 fully redundant Foundry BigIron MLX-16 switching cores that provides a Gigabit network backbone for all HPC computers, servers, and individual workstations connected via Foundry floor switches
6. Connections to the campus backbone via redundant 10 Gigabit Ethernet links - the first such attachments on campus
7. A variety of Intel and AMD based desktop workstations running Linux with the latest ATI or Nvidia graphics cards
8. Numerous Windows 7 desktop workstation
9. Numerous MacPro workstations running OSX 10.8 with 30\textquotedbl{} displays
10. Six Sun quad-core AMD Opteron high availability Linux servers providing core SCI IT services - website (www.sci.utah.edu), ftp, mail, and software distribution
11. Dedicated version control server with 6TB of local disk space for all SCI code and software projects
12. UPS power, including 100 minutes of battery backup for critical SCI servers

Power Display Wall: The interactive Power display wall provides users with the ability to explore 2D/3D visualizations on 36 (4 x 9) 27-inch tiled screens at a 133-megapixel resolution with 72GB of graphics memory. The display can be controlled by a computer and/or tablet device either on-site or by remote collaborators. Its infrastructure was designed to handle massive, terascale data sets from local or remote sources. Each node of the display wall operates 4 screens and can be configured by the controller to stream process the data as it is displayed to aid analysis. This is an ideal resource for local and remote collaborations where users need to examine fine details of large datasets while maintaining the global context.

Office Space: The SCI Institute houses its faculty and staff in individual offices. Students have individual desk space equipped with a workstation and located in large, open common areas that facilitate student collaboration and communication. All workstations are connected to the SCI local area network via full-duplex Gigabit Ethernet.

University Network: The University is a member of the Internet2 advanced networking consortium. It is connected to the Internet2 Network via 100 Gigabit Ethernet (100 Gbit per second) initially. The University, in partnership with the Utah Education Network, has the ability to extend dedicated wavelengths or dedicated circuits from Internet2 collocated in a Level 3 Communications facility west of downtown Salt Lake City.

Science DMZ: The University of Utah Science DMZ supports 100Gb/s connectivity through to Internet2. The Science DMZ is available at both the University Downtown Data Center and on the campus. The University of Utah is able to bring end users to the DMZ via 10Gb, 40Gb and MPLS connections. The Science DMZ supports specific segments of departments/institutes requiring high end performance to end hosts or instruments. By the end of summer 2014, the hardware supporting the Science DMZ will also allow for emerging Software Defined Network (SDN) technologies at those performance speeds. A collaboration at the University, including the Center for High Performance Computing, the Network Operations Center and the School of Computing Flux group, are working to instrument the hardware with application software developed at the University. This software will allow for isolated sandboxes and workflow substrates for experiments and different domain sciences workflows.

6.5.3. Computer Aided Design and Engineering (CADE) Facilities

The College maintains several general purpose computing facilities that are available for use by students in all of the engineering departments. The Engman lab consists of 97 Windows computers and includes software such as MATLAB, SolidWorks, Ansys, and a Windows Studio. The Engman teaching lab contains the same software but is located in a smaller classroom adjacent to the main lab and is used for instruction. The CADE lab contains 70 Linux workstations which provide access to the CAD tools such as Cadence, Synopsys, and Mentor Graphics. The CADE lab also has 90 Windows computers that are available for students to use. One floor down from the CADE lab we provide we maintain 24 Apple Mac computers that are available for students. Within the last year we have developed a virtual Windows lab that has up to 40 available computers and is accessible over the Internet.

The College also provides file storage for students, which will persist throughout their time at the University. We offer many services such as GIT and SVN software repositories, virtual servers for students that need access to databases or other software that is not normally provided in the labs.

6.6. Staff Support

Taking into account the growth within the Computer Science Program, the support provided by the staff has continued to be of high quality with committed and capable teams in each area. There has been a mild increase in the number of the supporting staff.

6.6.1. Ethnicity and diversity

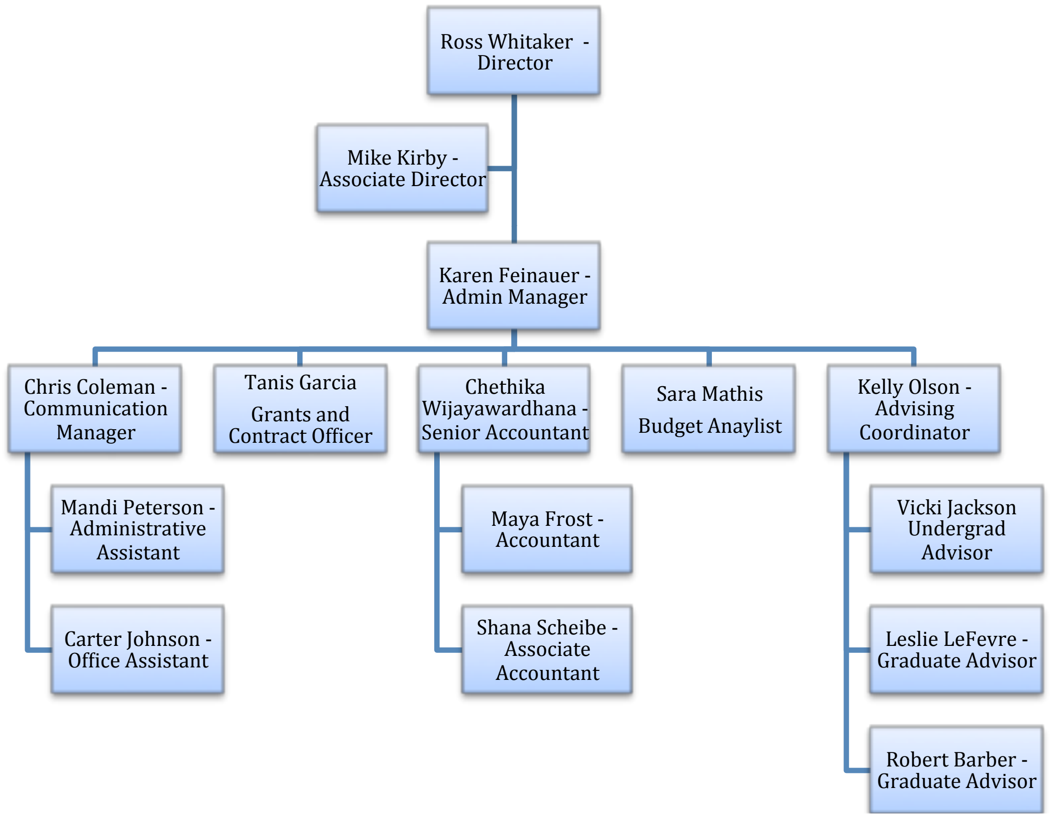
	Front Office Staff	Facility Staff	Post Docs
Male	2	4	7
Female	11	0	2
Caucasian	11	4	5
Asian	2	0	4

Table 6-2: Composition of the front office staff as of Fall 2016

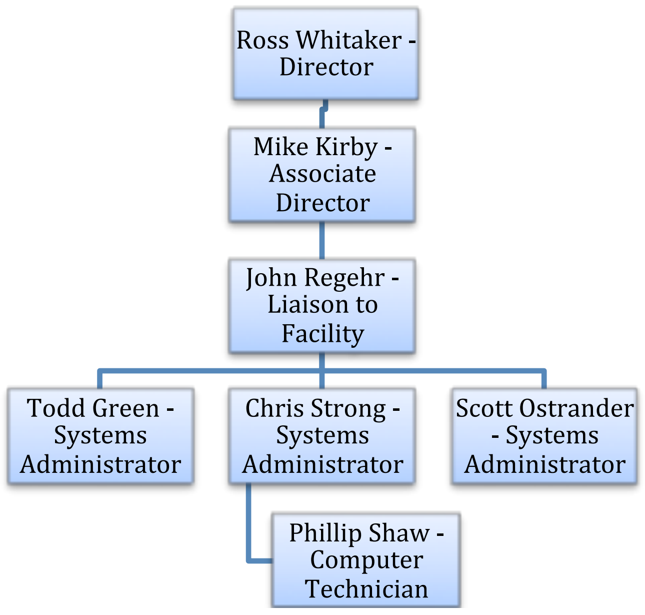
6.6.2. Staff participation and morale

The department has supported members of the team by providing in service events as well as providing access to conferences for advisors, accountants and grants and contract officers. This provides the department with well-versed and highly trained group of individuals. We are planning a staff retreat to allow for more input from staff concerning workload and the efficiency of the department.

Departmental Oganizational Chart


Ross Whitaker - Director
Mike Kirby - Associate Director
Karen Feinauer - Admin Manager
Chris Coleman - Communication Manager
Tanis Garcia Grants and Contract Officer
Chethika Wijayawardhana - Senior Accountant
Sara Mathis Budget Anaylist
Kelly Olson - Advising Coordinator
Mandi Peterson - Administrative Assistant
Carter Johnson - Office Assistant
Maya Frost - Accountant
Shana Scheibe - Associate Accountant
Vicki Jackson Undergrad Advisor
Leslie LeFevre - Graduate Advisor
Robert Barber - Graduate Advisor

Facility Organizational Chart


Ross Whitaker - Director
Mike Kirby - Associate Director
John Regehr - Liaison to Facility
Todd Green - Systems Administrator
Chris Strong - Systems Administrator
Scott Ostrander - Systems Administrator
Phillip Shaw - Computer Technician

Appendix A: School’s IAB Meeting Agenda

SoC Industrial Advisory Board Meeting

January 22, 2016

Agenda

9:00am	**Breakfast**	
9:20am	**Welcome and introductions**	
9:30am	**SoC overview**	Ross Whitaker
9:45am	**What does a computer scientist look like?**	Miriah Meyer
10:00am	**Development of personas**	
10:45am	**Break**	
11:00am	**Discussion about personas**	
11:45pm	**Space needs for SoC growth**	John LaLonde
12:15pm	**Lunch (incl. Dean Brown)**	Dean Rich Brown
1:00pm	**Retraining program**	Sneha Kasera
1:30pm	**Legislature update**	Galen Murdock
2:00pm	**Odds and ends**	
2:15pm	**CS+X**	Erik Brunvand
2:45pm	**Wrap-up**	
3:00pm	**Depart**	

Appendix B: School’s Career-line Faculty Review Operating Procedures

SoC Career-line Faculty Review Operating Procedures

(Updated Fall 2014)

On an annual basis, all Career-line faculty (as defined under UofU PPM 6-300) will be evaluated before reappointment. The evaluation will follow the College of Engineering policy, placing appropriate weight on teaching, research & scholarship, and service, depending on the type of Career-line appointment. The evaluation documents will be handled internally within the School of Computing.

It is expected that all multi-year Career-line faculty use the University FAR system to report their activities on an annual basis.

For promotion from Assistant Professor to Associate Professor or from Associate Professor to Full Professor, the College of Engineering RPT system will be used.

Career-line Teaching Faculty

On an annual basis, Career-line teaching faculty members should provide for evaluation and appointment:

1. A letter to the Director requesting reappointment.
2. A complete Curriculum Vita indicating research contributions (publications), courses taught, and service to the School of Computing, College of Engineering, University of Utah and external service.
3. A list of courses taught, enrollments, and the C7/I7 scores (course and instructor efficacy scores) for those courses.

The evaluation will be administered by the Director of the School of Computing, who may also request peer-teaching evaluations of each faculty, and a recommendation for reappointment will be made prior to the appointments faculty meeting.

For promotion, Career-line teaching faculty should provide the complete College of Engineering RPT file, which includes:

1. A complete Curriculum Vita indicating research contributions (publications), courses taught, and service to the School of Computing, College of Engineering, University of Utah and external service.
2. A teaching statement which describes teaching philosophy, courses taught, course evaluation, curriculum development (new courses and substantial changes to existing courses), and future plans for teaching.
3. A service statement which describes service philosophy (internal and external), service roles served and future plans for service.
4. A peer teaching review will be performed by Tenure-line faculty as described in the RPT policy. These reviews will serve as reference letters for the purposes of evaluation.

The evaluation will follow the School of Computing and College of Engineering RPT process and be conducted during an RPT meeting in the Fall. Exceptions to the timing may be allowed by the Director.

Career-line Research Faculty

On an annual basis, Career-line research faculty members should provide for evaluation and appointment:

1. A letter to the Director requesting reappointment.
2. A complete Curriculum Vita indicating research contributions (publications), current grants, grants submitted and not funded, courses taught (if appropriate), and service to the School of Computing, College of Engineering, University of Utah and external service.

The evaluation will be administered by the Director of the School of Computing, who may also request peer-teaching evaluations of each faculty if appropriate, and a recommendation for reappointment will be made prior to the appointments faculty meeting.

For promotion, Career-line research faculty should provide the complete College of Engineering RPT file, which includes:

1. A complete Curriculum Vita indicating research contributions (publications), funding, courses taught, and service to the School of Computing, College of Engineering, University of Utah and external service.
2. A research statement which describes research philosophy, research accomplishments and research goals.
3. A service statement which describes service philosophy (internal and external), service roles served and future plans for service.
4. A proposal of external examiners who can provide letters of recommendation. The Director will use the names provided along with other input to decide the final set of recommendations to request.

The evaluation will follow the School of Computing and College of Engineering RPT process and be conducted during an RPT meeting in the Fall. Exceptions to the timing may be allowed by the Director.

Appendix C: School’s Policy Statement on RPT

Policy Statement on Retention, Promotion and Tenure

School of Computing
University of Utah

April 2, 2008

1 Background

This document describes the policies and procedures used by the School of Computing relating to retention, promotion, and tenure of tenure track faculty during their probationary period at the University of Utah. It also covers policies and procedures relating to promotion of tenured faculty. University policy as defined in the Policies and Procedures Manual (P&PM) and the College of Engineering policy also apply.

2 Criteria for Retention, Promotion and Tenure

2.1 Areas

Faculty being considered for retention, promotion and/or tenure will be evaluated in the areas of research and scholarship, teaching, and service. A candidate's performance in each area will be assessed based upon the quality and impact, as well as the number, of accomplishments.

2.1.1 Research and Scholarship

Tenure or advancement in rank requires that the candidate contribute significantly and distinctly to the development and dissemination of new knowledge through research and publication of research results. The following will be considered in evaluating a candidate's research and scholarship according to accepted publishing patterns in the candidate's own research area:

- publication of original research papers in refereed technical journals and conference proceedings;
- the prestige of the journals and conferences and the quality, as well as number of publications will be considered;

- publication of research monographs, book chapters, and book reviews;
- presentations at conferences, workshops, colloquia or seminars. Keynote, plenary and invited talks will be noted;
- the ability to attract external funding sufficient to support an effective research program on a continuing basis; and
- patents issued and software licensed or otherwise distributed.

For tenure and promotion, external letters of evaluation from recognized authorities in the candidate's area will play a major role in helping assess the quality and impact of the candidate's research and scholarship, and his/her overall professional reputation.

2.1.2 Teaching

All regular faculty members are expected to be accomplished teachers at both the undergraduate and graduate levels. Quality teaching requires depth of pertinent knowledge, ability to inspire student interest in the subject, logical organization and presentation of the material, and fair and appropriate assessment of student performance. High quality student thesis and dissertation advising is essential.

Classroom teaching effectiveness is documented through:

- peer and student evaluations;
- development of new courses, improvement of existing courses, and introduction of innovative teaching techniques;
- the variety and nature of courses taught;
- advising of undergraduate student projects;
- publication of textbooks or other teaching materials; and
- teaching awards.

Research-related teaching contributions are evaluated based upon:

- the quality and impact of research undertaken by the candidate's students;
- the number of graduate students advised;
- the quality and number of publications authored jointly by the candidate and student advisees; and
- evidence of student mentoring outside of formal thesis and dissertation advising roles.

Other evidence of teaching contributions to be considered include external funding for curriculum development, and general impact of the faculty member's work on educational issues.

2.1.3 Professional Service

Candidates for tenure or advancement in rank are expected to have contributed significantly to departmental, college and/or university affairs through involvement in faculty governance, committee service, and other assignments. Participation is also expected in professional service beyond the university, such as involvement in professional society activities, editorial boards, conference committees, advisory committees, and reviewing of proposals and publications. Community and government service activities will also be considered. In addition to the list of service assignments, the candidate's effectiveness, leadership and reliability in these roles is expected.

2.2 Evaluation Criteria

The School of Computing is committed to excellence in each of the areas of evaluation. Reviews should consider the sum of all contributions a candidate has made in teaching, research and service. To be recommended for promotion or tenure, a candidate should be an outstanding scholar, with substantial contributions in each of these areas.

Retention is recommended for a tenure-eligible faculty member when there is a reasonable probability that tenure will be granted at the end of the probationary period. A faculty member will be retained when s/he is performing well, is making substantial progress, or despite concerns, has a reasonable possibility of meeting the requirements for promotion and tenure. In order to be tenured or promoted, the candidate will need to address the concerns, deficiencies and suggestions for improvement noted in the informal and formal reviews. The candidate should discuss progress on these points with the School Director and the RPT Chair each year during the probationary period.

Promotion to Associate Professor requires an individual to have: demonstrated substantial achievement and impact in research and scholarship; demonstrated teaching effectiveness; and performed an appropriate amount of quality service both within the University and in the individual's professional community.

Promotion to Professor requires candidates to have made major creative contributions to their areas of research and to have had significant impact on their discipline as verified through their national and international reputations. Promotion to Professor should include evidence of a demonstrated ability to sustain contributions to the field and to the School of Computing. High quality teaching and service within the School of Computing and professional community are required.

Tenure is awarded only to individuals who have demonstrated substantial achievement and future promise. To be tenured, a candidate must have established a vigorous research program; be seen by external reviewers as a leading scholar among his/her peers; be an outstanding teacher; be a responsible faculty member whose conduct has a positive influence on students and colleagues, and demonstrate a high likelihood of sustaining contributions. While it is desirable for faculty members being tenured to have graduated a Ph.D. student, evaluation should focus on a demonstrated competence to mentor and supervise Ph.D. students sufficient to indicate the candidate's ability to consistently produce Ph.D. graduates. The award of tenure carries an obligation of continued superior performance on the part of the candidate, for which the University in turn offers a stable environment in which to pursue excellence in teaching, research and service.

3 RPT Procedures

RPT reviews can be either formal or informal. At the level of the School of Computing, an informal review contains every aspect of a formal review except that external letters and SAC reports are not requested. The report of an informal review is placed in the permanent RPT file of the candidate and sent to the Dean of the College of Engineering.

Unless modified by officially designated credit for prior service elsewhere, the probationary period for those appointed at the rank of assistant professor is six years and is five years for those appointed at the ranks of associate professor or professor. All probationary faculty must receive a formal tenure review by the final year of their probationary period. In addition, all probationary faculty must receive a formal mid-probationary retention review in their third year of probationary service.

Additional formal retention reviews will be performed following a majority vote of the RPT Committee. Subject to the restrictions imposed by PPM, a candidate may request in writing an early formal promotion and tenure review based on extraordinary progress. An early formal promotion and tenure review must be supported by both the Director of the School and RPT Chair and a majority vote of the RPT Committee to proceed. The request or vote must occur prior to the deadline for formal review requests set by the College of Engineering (this date is customarily sometime in the spring semester preceding the academic year in which the formal review is to be conducted). If the request for promotion and tenure is more than one year earlier than the normal probationary period, it must also be approved by the dean and the senior vice president for academic affairs.

Formal promotion reviews for tenured Associate Professors will be done following a written request of the candidate, assuming that request is submitted prior to the deadline for formal review requests set by the College of Engineering.

All probationary faculty will be reviewed informally in each year they are not reviewed formally.

3.1 RPT File

It is the responsibility of each faculty member subject to either formal or informal review to ensure that his or her file contains the necessary, current, and complete documentation required for the review. As a minimum for both formal and informal reviews, this includes the material listed in the College of Engineering Instructions for RPT Submissions.

- The following materials should be compiled and furnished by the candidate for the RPT file (see the College instructions for details on content).

  Curriculum Vita (updated and complete)

  Research/Scholarship Data Formal cases will include a signed copy of the OSP funding report. A research statement describing the candidate's research vision and goals should be included.

  Service Data This should include information on participation and contributions to conference committees, school of computing, college and university committees, review panels, editorial duties, and other service roles accomplished.

  Publications **Faculty undergoing formal review should include copies of selected publications representing their most important work.**

- The following materials will be compiled and put into the RPT file by the School of Computing under the direction of the SoC Director:

  Teaching Data: This should include information from the past three years for retention cases, and information for the entire probationary period for tenure cases. **For promotion to Professor, teaching data since the previous promotion (or appointment if hired as Associate Professor) should be included. If that promotion or appointment was more than five years earlier, teaching data should be included for at least the most recent five years.**

  Peer Teaching Reviews: The RPT subcommittee assigned to the candidate will prepare peer teaching reviews for the file.

  SAC Reviews: For formal RPT files the SoC Director will inform the SACs (UGSAC (Undergraduate Student Advisory Committee) and GradSAC (Graduate Student Advisory Committee)) of their responsibility to produce a report for the review, the guidelines for that report, and the timetable for that report's completion. The SoC Director will ensure that the SAC reports, if produced according to the timetable, are included in the file.

  External Letters: External letters of evaluation will be included in formal reviews. The candidate will be asked to provide the names of four suggested reviewers. Letters will be requested by the Director of the School from all four of the candidate's suggestions. The RPT committee will provide names of four or five additional persons from whom letters will also be requested. The Director may request letters from additional external reviewers as well. The file will include a sheet that identifies who suggested each of the names of the external reviewers: the candidate, the RPT committee, or the Director. It should also contain brief biographical information on each letter writer, sufficient to provide an indication of the prominence of the letter writer. The names of the additional reviewers and the biographical information are not given to the candidate. The file for formal reviews must include a minimum of six substantive letters from evaluators outside the University. At least three of these letters must come from individuals not suggested by the candidate.

In those cases where the candidate has waived his or her rights to access to these letters, reasonable efforts should be taken by the School of Computing to protect the identity of those solicited for letters as well as the contents of any response. See the University Policy and Procedures Manual, PPM 9-5.1 section D.9 for specifics on the form to waive or not waive the candidate's right to see the external letters of evaluation. A signed copy of this form is a required element of the RPT file.

Unsolicited Letters from Interested Parties: Parties interested in the RPT case (former students, colleagues, coworkers, etc.) may send unsolicited letters to be included in the candidate's RPT file. These letters are included in the open portion of the candidate's file.

All RPT Reports from Prior Years: Copies of each of the candidate's informal and formal RPT committee reports from prior years, along with candidate responses to those reports, if any. **For promotion to Professor, the file shall include the candidate's vita at the time of the previous promotion (or at appointment if hired as Associate Professor), and all reports and recommendations from tenured faculty reviews.**

Summary report of RPT Advisory Committee: The summary report lists members present at the review and is signed by all those committee members. The report summarizes the substance of the discussion at the RPT meeting and describes the findings and recommendations of the RPT Advisory Committee. For mid-probationary and informal reviews, the report should summarize comments from members of the RPT Advisory Committee concerning areas requiring improvement. If necessary, majority and minority summaries may appear in the report; however, the language of the report should not contain any references to other specific and identifiable RPT cases. If disagreements regarding the report cannot be resolved, dissenting members of the School RPT Advisory Committee will be allowed to add their own statements to the file. The original document is placed in the file and a copy is sent to the candidate.

School Director's Response: The response from the Director of the School must state objective reasons for the decision and should also be a stand-alone document that can be understood without reference to the candidate's file. Again, the candidate should be informed clearly of areas where performance needs improvement. The original document is placed in the file and a copy is sent to the candidate.

3.2 RPT Logistics

RPT actions proceed according to the following logistics:

- The School RPT Committee consists of all regular faculty in the School eligible to vote on RPT issues as specified in PPM.

- A school RPT chair is selected from the members of the School's RPT Committee by vote of the **regular** faculty. In accordance with Policies and Procedures, all **regular** faculty members at the rank of professor, associate professor, assistant professor, and instructor shall be entitled to vote. The RPT chair must be tenured and cannot be the Director of the School. The selection of the School's RPT Chair will be done early enough in the Spring semester prior to formal RPT actions in the following academic year so that subcommittees for formal reviews may be formed in time to perform Spring Semester peer teaching evaluations.

- The School RPT Chair forms a subcommittee for each candidate being reviewed. Subcommittees will consist of at least two faculty members for informal reviews and at least three faculty members for formal reviews. Subcommittee members must be members of the School's RPT Committee and eligible to vote on the case, as specified in this document and PPM 9-5.1 section A.3.a.

  For formal reviews, the responsibilities of the subcommittee are:

  – Nominate external evaluators for consideration by the full RPT Committee.

  – Conduct and report on peer teaching evaluations in both Spring and Fall semesters. All relevant peer teaching reports must be entered into the candidate's file at least one week in advance of the first meeting of the RPT Committee to consider the case, and should be made available to the candidate for potential **written** comment by the candidate.

  – Prepare a summary of the candidate's publications and other research and scholarship accomplishments, along with a qualitative assessment of the journals and conference proceedings in which the publications appeared. This summary should be produced in consultation with the candidate, entered into the candidate's file at least one week in advance of the first meeting of the RPT Committee to consider the case, and should be made available to the candidate for potential comment by the candidate.

  For informal reviews, the responsibilities of the subcommittee are the same as for formal reviews, except that no external letters of evaluation are involved.

- Voting in RPT meetings shall always be by open ballot. Absentee voting is permitted according to the University Policy and Procedures Manual PPM 9-5.1 section E.4.

- Remote attendance at RPT meetings will be allowed by phone, video conference, or other remote conferencing technology if there is a majority agreement from the RPT committee members physically present at the RPT meeting.

- A secretary of each meeting shall be designated by the chairperson of the department RPT advisory committee and shall take notes of the discussion to provide the basis for developing a summary. Within one week after completion of deliberations on a candidate, the secretary shall prepare a draft of the RPT Committee Report, which summarizes the substance of the discussion and provides the findings and recommendations of the department advisory committee. This report should be made available to the members of the RPT committee, after which there will be a five day review period in which the committee members can review the report and suggest changes.

- For formal reviews the RPT process proceeds according to the College of Engineering RPT calendar. Each year the College prepares a calendar listing deadline dates for formal RPT actions. The School RPT chair will schedule School RPT actions, including the School's fall RPT meeting during the time frame specified in the College calendar.
- For informal reviews, the School RPT chair should specify in advance a schedule that is patterned after the College of Engineering calendar for formal RPT actions, except that the latest date for final completion of the process should be the last day of classes in the Fall semester.

Appendix D: Faculty Curriculum Vitae

CURRICULUM VITAE

Rajeev Balasubramonian
rajeev@cs.utah.edu, http://www.cs.utah.edu/~rajeev

Education:

Ph.D., Computer Science, University of Rochester, August 2003
Dissertation: Dynamic Management of Microarchitecture Resources in Future Microprocessors
Advisor: Prof. Sandhya Dwarkadas (University of Rochester)
Co-advisor: Prof. David Albonesi (Cornell University)

M.S., Computer Science, University of Rochester, May 2000

B.Tech., Computer Science and Engineering, Indian Institute of Technology, Bombay, 1998

Recent Employment:

July 2015 – present. Professor, School of Computing, University of Utah.

July 2009 – June 2015. Associate Professor, School of Computing, University of Utah.

May 2014 – May 2015. Visiting Scholar, HP Labs, Palo Alto.

August 2003 – June 2009. Assistant Professor, School of Computing, University of Utah.

Research Interests:

Computer Architecture: Innovations to improve performance, energy efficiency, reliability, security, and cost of memory systems and neuromorphic architectures.

Honors and Awards:

Member of the ISCA, MICRO, and HPCA Hall of Fame:
`http://pages.cs.wisc.edu/~arch/www/iscabibhall.html`.
`http://newsletter.sigmicro.org/micro-hof.txt/view`.
`http://www.ieeetcca.org/newsletter/hpca-hall-of-fame/`.

ISPASS'16 paper on DRAM refresh nominated for a Best Paper award.

HPCA'14 paper on the Sandbox prefetcher selected as a Top Picks Honorable Mention by IEEE Micro.

IBM Faculty Partnership Award, 2012, 2013.

HP Innovation Research Program Award, 2010, 2011, 2012.

HPCA'10 paper on DRAM caching selected to appear in the Top Picks special issue by IEEE Micro[1].

PACT'10 paper on multiple memory controllers selected for a Best Paper award.

[1]IEEE Micro's annual Top Picks special issue recognizes a dozen papers as "the year's most significant research publications in computer architecture based on novelty and industry relevance".

HiPC'09 paper on low-power interconnects selected for a Best Paper award.

MICRO'07 paper on large cache interconnects selected to appear in the Top Picks special issue by IEEE Micro.

NSF Faculty Early Career Development Award (CAREER), 2006.

Outstanding Teaching Award 2005, School of Computing, University of Utah.

Dean's teaching commendation letters for CS 7968 Parallel Computer Architecture (Spring 2005) and CS/ECE 6810 Computer Architecture (Fall 2005, Fall 2008, Fall 2012, Spring 2015) for student teaching evaluation ratings among the top 15% in the College of Engineering.

Funding:

Total funding: $3,481,227. Total funding as PI: $3,193,936.

Active large grants:

"CSR: Small: Adaptive Brink-of-Failure Memory Architectures for Future Technologies and Workloads", R. Balasubramonian (PI), NSF Award No. CNS-1423583, $499,096, August 2014 - July 2017.

"CSR: Medium: Energy-Efficient Architectures for Emerging Big-Data Workloads", R. Balasubramonian (PI), A. Davis, M. Hall, F. Li, NSF Award No. CNS-1302663, $873,286, July 2013 - June 2017.

Publications:

Only top-tier publications from the last three years are listed.

1. A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, V. Srikumar, "ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars," 43rd International Symposium on Computer Architecture **(ISCA-43)**, Seoul, June 2016. *Acceptance rate: 58/291 (20%).* ◐

2. M. Shevgoor, S. Koladiya, R. Balasubramonian, S. Pugsley, C. Wilkerson, Z. Chishti, "Efficiently Prefetching Complex Address Patterns," 48th International Symposium on Microarchitecture **(MICRO-48)**, Hawaii, December 2015. *Acceptance rate: 61/283 (22%).* ◐

3. A. Shafiee, A. Gundu, M. Shevgoor, R. Balasubramonian, M. Tiwari, "Avoiding Information Leakage in the Memory Controller with Fixed Service Policies," 48th International Symposium on Microarchitecture **(MICRO-48)**, Hawaii, December 2015. *Acceptance rate: 61/283 (22%).* ◐

4. C. Xu, D. Niu, N. Muralimanohar, R. Balasubramonian, T. Zhang, S. Yu, Y. Xie, "Overcoming the Challenges of Crossbar Resistive Memory Architectures," 21st International Symposium on High-Performance Computer Architecture **(HPCA-21)**, San Francisco, February 2015. *Acceptance rate: 51/226 (23%).* ○

5. N. Chatterjee, M. O'Connor, G.H. Loh, N. Jayasena, R. Balasubramonian, "Managing DRAM Latency Divergence in Irregular GPGPU Applications," **SC'14** – The International Conference for High Performance Computing, Networking, Storage, and Analysis, November 2014. *Acceptance rate: 82/394 (21%).* ◐

6. S. Pugsley, J. Jestes, R. Balasubramonian, V. Srinivasan, A. Buyuktosunoglu, A. Davis, F. Li, “Comparing Different Implementations of Near Data Computing with In-Memory MapReduce Workloads,” **IEEE Micro Special Issue on Big Data**, July/August 2014. ◐

7. R. Balasubramonian, J. Chang, T. Manning, J. Moreno, R. Murphy, R. Nair, S. Swanson, “Near-Data Processing: Insight from a Workshop at MICRO-46,” **IEEE Micro Special Issue on Big Data**, July/August 2014. ○

8. A. Shafiee, M. Taassori, R. Balasubramonian, A. Davis, “MemZip: Exploiting Unconventional Benefits from Memory Compression,” 20th International Symposium on High-Performance Computer Architecture **(HPCA-20)**, February 2014. *Acceptance rate: 55/215 (26%).* ◐

9. S. Pugsley, Z. Chishti, C. Wilkerson, T. Chuang, R. Scott, A. Jaleel, S.-L. Lu, K. Chow, R. Balasubramonian, “Sandbox Prefetching: Safe, Run-Time Evaluation of Aggressive Prefetchers,” 20th International Symposium on High-Performance Computer Architecture **(HPCA-20)**, February 2014. *Acceptance rate: 55/215 (26%).* ○

10. M. Shevgoor, J.-S. Kim, N. Chatterjee, R. Balasubramonian, A. Davis, A. Udipi, “Quantifying the Relationship between the Power Delivery Network and Architectural Policies in a 3D-Stacked Memory Device,” 46th International Symposium on Microarchitecture **(MICRO-46)**, December 2013. *Acceptance rate: 39/239 (16%).* ◐

External Service (top-tier venues only):

- ASPLOS: co-General Chair (2014), Steering Committee (2015, 2016), Program Committee member (2015, 2016), External Review Committee (2012).
- ISCA: Program Committee member (2012, 2015, 2016), Student Travel Grant Chair (2013), Finance Chair (2011, 2016), External Review Committee (2010, 2014).
- MICRO: Program Committee member (2008, 2011, 2012, 2015), External Review Committee (2014, 2016).
- HPCA: Program Committee member (2008, 2011, 2014, 2016), Finance Chair (2013), Registration Chair (2008, 2010), External Review Committee (2015).

University Service:

- Faculty Search Committee: Architecture (Spring 2015, 2016), HCI (Spring 2013, 2015).
- CoE/SoC scholarship committee, Spring 2015, 2016.
- Graduate admissions chair, School of Computing, 2010-2012.

Advising:

Ph.D. Graduates: 8

M.S. Graduates: 9

Teaching:

CS/ECE 3810 Computer Organization. A required under-graduate course, based on Patterson and Hennessy's "Computer Organization and Design". URL: `http://www.cs.utah.edu/~rajeev/cs3810/`

Years Taught	Enrollment	Effective Course Rating	Effective Instructor Rating
Fall 2006	104	4.98/6.0 (CS), 4.47/6.0 (ECE)	5.18/6.0 (CS), 5.15/6.0 (ECE)
Fall 2015	181	5.03/6.0	5.33/6.0

CS/ECE 6810 Computer Architecture. A required graduate course, based on Hennessy and Patterson's "Computer Architecture: A Quantitative Approach". The course has used a modified flipped classroom model in Fall 2012 and Fall 2013. Received **Dean's Teaching Commendation Letter** for student teaching evaluation ratings among the top 15% in the College of Engineering (**Fall'05, Fall'08, Fall'12, Spring'15**). URL: `http://www.eng.utah.edu/~cs6810/`

Years Taught	Enrollment	Effective Course Rating	Effective Instructor Rating
Fall 2004	36	5.59/6.0	5.66/6.0
Fall 2005	48	5.61/6.0	5.72/6.0
Fall 2007	53	5.40/6.0	5.66/6.0
Fall 2008	50	5.76/6.0	5.82/6.0
Fall 2010	80	5.48/6.0	5.62/6.0
Spring 2012	50	5.37/6.0	5.48/6.0
Fall 2012	30	5.64/6.0	5.75/6.0
Fall 2013	73	5.53/6.0	5.69/6.0
Spring 2015	103	5.60/6.0	5.70/6.0

CS/ECE 7810 Advanced Computer Architecture. An advanced graduate course, based on recent papers and simulation-based projects. Student class projects have appeared at five ISCA workshops. URL: `http://www.eng.utah.edu/~cs7810/`

Years Taught	Enrollment	Effective Course Rating	Effective Instructor Rating
Spring 2004	4	6.0/6.0	6.0/6.0
Spring 2006	15	6.0/6.0	6.0/6.0
Spring 2009	10	6.0/6.0	6.0/6.0
Spring 2011	15	5.5/6.0	5.67/6.0
Spring 2013	8	5.33/6.0	5.75/6.0
Spring 2014	15	5.4/6.0	5.75/6.0

CS 7940/7937 Architecture Reading Seminar. Organized every semester. URL: `http://www.cs.utah.edu/arch-rd-web/`

168 screencasts of my graduate and undergraduate computer architecture classes on YouTube have collectively received over 735,000 views and 3,100 subscribers worldwide.

Curriculum Vitae for Professor Martin Berzins

Professional Career

2006-	Visiting Professor School of Computing,University of Leeds
2005-10	Director School of Computing,University of Utah
2003-05	Associate Director School of Computing,University of Utah
2003-	Professor,School of Computing,University of Utah
2003-	Professor,SCI Institute,University of Utah
2001-03	Dean for Research, Faculty of Engineering, University of Leeds
1999-03	Professor of Scientific Computation, School of Computer Studies, University of Leeds
1997	Sabbatical as Visiting Associate Professor, Computer Science at University of Utah, Salt Lake City, Utah, U.S.A.
1996-99	Reader in Computational PDEs, School of Computer Studies, University of Leeds
1992-96	Senior Lecturer, School of Computer Studies, University of Leeds
1988	Sabbatical as Visiting Associate Professor at R.P.I. Troy, New York, USA
1984-91	Lecturer in Numerical Analysis, School of Computer Studies, University of Leeds
1982-84	Research Fellow, School of Computer Studies, University of Leeds Topic: Software for Time Dependent P.D.E. Problems
1981-82	Research Fellow, Department of Computer Studies, University of Leeds Topic: Parallel Processing
1978-81	Ph.D. Chebyshev Polynomial Methods for Parabolic Equations. Department of Computer Studies, University of Leeds
1975-78	B.Sc. Hons. Maths (II.i), University of Leeds (Year 1-Manchester(II.i))

Research Center Leadership Roles

1996-2003	Co-Founder and Director Computational PDEs Unit University of Leeds.
2012-2022	P.I. and Manager ARL Co-operative Alliance in Multi-scale Material Modeling, $2M per year.
2013-2018	PSAAP2 Center, University of Utah $3.2M per year, Computer Science Lead $1.3M per year.

Address: Scientific Computing and Imaging Institute, 72 S.Central campus Dr. Rm 3750, Salt Lake City Utah 84112. Phone 801 585 1545, email mb@sci.utah.edu

Professional Bodies.

Fellow of the United Kingdom Institute of Mathematics and Its Applications (IMA) and Chartered Mathematician (C.Math.)
Member of the SIAM, ACM and IEEE.
Member of NAG Ltd (software company) and member of NAG Inc Board until 2014.

A Motivating Quote.

” For a university should be before all else, and I borrow a favorite remark from Edward Boyle, one-time Secretary of State for Education and then until his early death Vice-Chancellor of the University of Leeds: *A university should be a place in which teaching is conducted in an atmosphere of research.* He would not have objected if the sentence had been turned on its head. It has a rock-like simplicity...” Richard Hoggart.

Publications I. Papers in Refereed Journals 2009-2016

[1] Tran L.T. Kim J. and Berzins M. Solving Time-Dependent PDEs using the Material Point Method, A Case Study from Gas Dynamics. International Journal for Numerical Methods in Fluids 2009 Vol. 62, No. 7, pp. 709–732. 2009.

[2] M. Steffen, R. M. Kirby, M. Berzins, *Decoupling and Balancing of Space and Time Errors in the Material Point Method (MPM)*, International Journal for Numerical Methods in Engineering, Vol. 82 (10), pp. 1207-1243, 2010. Published online December 9th 2009

[3] D.Hart, M.Berzins C.E. Goodyer and P.K. Jimack. *Using Adjoint Error Estimation Techniques for Elastohydrodynamic Lubrication Line Contact Problems.* International Journal for Numerical Methods in Fluids, Vol. 67. pp1559-1570, 2011.

[4] M.Berzins *Nonlinear data-bounded polynomial approximations and their applications in ENO methods* Numerical Algorithms Volume 55, Issue 2 (2010) , Page 171ff.

[5] J. Luitjens and M.Berzins *Parallel regridding algorithms for block-structured adaptive mesh refinement* Concurrency and Computation, 23, 1522-1537, 2011.

[6] L.T. Tran, M. Berzins. *IMPICE Method for Compressible Flow Problems in Uintah,* International Journal For Numerical Methods In Fluids, Vol 69, Issue 5, 926-965, 2012.

[7] H.Lu, M.Berzins, C.E. Goodyer and P.K. Jimack *Adaptive High-Order Discontinuous Galerkin Solution of Elastohydrodynamic Lubrication Point Contact Problems* Advances in Engineering Software Vol. 45, No. 1, pp. 313–324. 2012. .

[8] M.Berzins *Data and range-bounded polynomials in ENO methods* Journal of Computational Science. Vol. 4, No. 1-2, pp. 62–70. 2013.

[9] Q. Meng, M. Berzins. Scalable Large-scale Fluid-structure Interaction Solvers in the Uintah Framework via Hybrid Task-based Parallelism Algorithms, Concurrency and Computation (accepted 2013).

[10] J. Beckvermit, J. Peterson, T. Harman, S. Bardenhagen, C. Wight, Q. Meng, and M. Berzins. "Multiscale Modeling of Accidental Explosions and Detonations", Computing in Science and Engineering, pp76-86, July/August 2013.

[11] A. Dubey and A. Almgren and John Bell and M. Berzins and S. Brandt and G. Bryan and P. Colella and D. Graves and M. Lijewski and F. Löffler and B. OShea and E. Schnetter and B. Van Straalen and K. Weide", "A survey of high level frameworks in block-structured adaptive mesh refinement packages", Journal of Parallel and Distributed Computing, 2014,

[12] A. Humphrey and Q. Meng and M. Berzins and D. Caminha B.de Oliveira and Z. Rakamaric and G. Gopalakrishnan", Systematic Debugging Methods for Large-Scale HPC Computational Frameworks, Computing in Science Engineering,16,3,48–56,2014,May.

[13] Q. Meng and M. Berzins, Scalable large-scale fluid-structure interaction solvers in the Uintah framework via hybrid task-based parallelism algorithms, Concurrency and Computation: Practice and Experience,26,7,1388–1407,2014,

[14] M. Berzins, J. Beckvermit, T. Harman, A. Bezdjian, A. Humphrey, Q. Meng, J. Schmidt,, C. Wight. Extending the Uintah Framework through the Petascale Modeling of Detonation in Arrays of High Explosive Devices, In SIAM Journal on Scientific Computing (Accepted), 2016.

Publications II. Refereed Confererence Papers and and Parts of Books 2009-2016

[15] J. Luitjens, M. Berzins. *Improving the Performance of Uintah: A Large-Scale Adaptive Meshing Computational Framework,* In Proceedings of the 24th IEEE International Parallel and Distributed Processing Symposium (IPDPS10), Atlanta, GA, pp 1-10. 24 May 2010, ISBN: 978-1-4244-6442-5.

[16] J. Schmidt, M. Berzins. *Development of the Uintah Gateway for Fluid-Structure-Interaction Problems,* In Proceedings of the Teragrid 2010 Conference TG'10 , ACM ISBN: 978-1-60558-818-6, 2010.

[17] M. Berzins, J. Luitjens, Q. Meng, T. Harman, C.A. Wight, J.R. Peterson. *Uintah - A Scalable Framework for Hazard Analysis,* In Proceedings of the Teragrid 2010 Conference, No. 3, Note: Awarded Best Paper in the Science Track., ACM, ISBN: 978-1-60558-818-6. 2010.

[18] Q. Meng, J. Luitjens, M. Berzins. *Dynamic Task Scheduling for the Uintah Framework*, In Proceedings of the 3rd IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS10), 2010, co-located with IEEE/ACM Supercomputing 2010.

[19] M. Berzins, Q.Meng, J.Schmidt, J. Sutherland. *DAG-Based Software Frameworks for PDEs* In Proceedings of HPSS 2011, M. Alexander et al. (Eds.): Euro-Par 2011 Workshops, Part I, LNCS 7155, pp. 324333, 2012. Springer-Verlag Berlin Heidelberg 2012.

[20] Q. Meng, M. Berzins, J. Schmidt. *Using hybrid parallelism to improve memory use in Uintah,* In Proceedings of the Teragrid 2011 Conference, Salt Lake City, Utah, ACM, July, 2011.

[21] J. R. Peterson, J. C. Beckvermit, T. Harman, M. Berzins, C.A. Wight Multiscale Modeling of High Explosives for Transportation Accidents Proceedings of XSEDE12 July 16-20, 2012, Chicago, IL, USA.

[22] Humphrey, A., Meng, Q., Berzins, M., Harman, T. "Radiation Modeling Using the Uintah Heterogeneous CPU/GPU Runtime System" Proceedings of XSEDE12 July 16-20, 2012, Chicago, IL, USA. (ACM)

[23] Qingyu Meng, Alan Humphrey and Martin Berzins *The Uintah Framework: A Unified Heterogeneous Task Scheduling and Runtime System* Proceedings of The Second International Workshop on Domain Specific Languages and High-level Frameworks for High Performance Computing (WOLFHPC) 2012, Salt Lake City, November 2012, ACM.

[24] M. Berzins, J. Schmidt, Q. Meng, A. Humphrey. Past, Present, and Future Scalability of the Uintah Software, Proceedings of the Extreme Scaling Workshop. University of Illinois at Urbana-Champaign, Champaign, IL, USA. (Chicago, IL, USA July 15 - 16, 2012) ACM 2013.

[25] D.C.B. de Oliveira, Z. Rakamaric, G. Gopalakrishnan, A. Humphrey, Q. Meng, M. Berzins. Crash Early, Crash Often, Explain Well: Practical Formal Correctness Checking of Million-core Problem Solving Environments for HPC, In Proceedings of the 35th International Conference on Software Engineering (ICSE 2013), pp. (accepted). 2013.

[26] Q. Meng, A. Humphrey, J. Schmidt, M. Berzins. Preliminary Experiences with the Uintah Framework on Intel Xeon Phi and Stampede, In Proceedings of the 2nd Conference of the Extreme Science and Engineering Discovery Environment (XSEDE 2013). 2013.

[27] M. Hall, J. Beckvermit, C. Wight, T. Harman, and M. Berzins. The Influence of an Applied Heat Flux on the Violence of Reaction of an Explosive Device, Proc. 2013 XSEDE Conf. 2013.

[28] Q. Meng, A. Humphrey, J. Schmidt and M. Berzins. Investigating Applications Portability with the Uintah DAG-based Runtime System on PetaScale Supercomputers, Proc. 2013 the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'13), pp. (accepted), 2013.

[29] J.R. Peterson, C.A. Wight, M. Berzins. Applying high-performance computing to petascale explosive simulations, In Procedia Computer Science, 2013. (Publication from the International Conference on Computational Science June 2013, Barcelona,).

[30] J. Schmidt and M. Berzins and J. Thornock and T. Saad and J. Sutherland., "Large Scale Parallel Solution of Incompressible Flow Problems using Uintah and hypre", "2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)", 458–465,2013.

[31] Z. Fu, H.K. Dasari, M. Berzins, B. Thompson. Parallel Breadth First Search on GPU Clusters, In Proceedings of the IEEE BigData 2014 Conference, Washington DC, October, 2014.

[32] S. Kumar, C. Christensen, P.-T. Bremer, E. Brugger, V. Pascucci, J. Schmidt, M. Berzins, H. Kolla, J. Chen, V. Vishwanath, P. Carns, R. Grout. Fast Multi-Resolution Reads of Massive Simulation Datasets, In Proceedings of the International Supercomputing Conference ISC'14, Leipzig, Germany, June, 2014.

[33] D.C.B. de Oliveira, A. Humphrey, Q. Meng, Z. Rakamaric, M. Berzins, G. Gopalakrishnan. Systematic Debugging of Concurrent Systems Using Coalesced Stack Trace Graphs, In Proceedings of the 27th International Workshop on Languages and Compilers for Parallel Computing (LCPC), September, 2014.

[34] C. Gritton, M. Berzins, R. M. Kirby. Improving Accuracy In Particle Methods Using Null Spaces and Filters, In Proceedings of the IV International Conference on Particle-Based Methods - Fundamentals and Applications, Barcelona, Spain, Edited by E. Onate, M. Bischoff, D.R.J. Owen, P. Wriggers, and T. Zohdi, CIMNE, pp. 202-213. September, 2015. ISBN: 978-84-944244-7-2

[35] J. K. Holmen, A. Humphrey, M. Berzins. Exploring Use of the Reserved Core, In High Performance Parallelism Pearls: Multicore and Many-core Programming Approaches, Vol. 2, Edited by J. Reinders and J. Jeffers, Elsevier, 2015.

[36] A. Humphrey, T. Harman, M. Berzins, P. Smith. A Scalable Algorithm for Radiative Heat Transfer Using Reverse Monte Carlo Ray Tracing, In High Performance Computing, Lecture Notes in Computer Science, Vol. 9137, Edited by Kunkel, Julian M. and Ludwig, Thomas, Springer International Publishing, pp. 212-230. 2015.

[37] A. Humphrey, D. Sunderland, T. Harman, M. Berzins. Radiative Heat Transfer Calculation on 16384 GPUs Using a Reverse Monte Carlo Ray Tracing Approach with Adaptive Mesh Refinement, In Accepted - The 17th IEEE International Workshop on Parallel and Distributed Scientific and Engineering Computing (PDSEC 2016), 2016.

[38] Brad Peterson, Harish Kumar Dasari, Alan Humphrey, James C. Sutherland, Tony Saad, Martin Berzins: Reducing overhead in the Uintah framework to support short-lived tasks on GPU-heterogeneous architectures. 5th International Workshop on Domain-Specific Languages and High-Level Frameworks for High Performance Computing (WOLFHPC'15), 4:1-4:8 ACM SIGHPC

[39] Brad Peterson, Nan Xiao, John Holmen, Sahithi Chaganti, Aditya Pakki, John Schmidt, Dan Sunderland, Alan Humphrey, Martin Berzins. Developing Uintahs Runtime System For Forthcoming Architectures Refereered and selected in competition for the RESPA15 Workshop at SuperComputing15 Conference. Available from the web at http://respa15.rice.edu/files/2015/11/9-2csbfkt.pdf

Publications III. Reports, Books and Other Contributions, 2009-2016.

[40] D. Reed, M. Berzins, R. Lucas, S. Matsuoka, R. Pennington, V. Sarkar, V. Taylor. OE Advanced Scientific Computing Advisory Committee (ASCAC) Report: Exascale Computing Initiative Review, 2015. DOI: DOI 10.2172/1222712

[41] Robert M. Kirby, Martin Berzins and Jan S. Hesthaven (Editors), Spectral and High Order Methods for Partial Differential Equations: Selected Papers from the ICOSAHOM'14 Conference, June 23-27, 2014, Salt Lake City, UT, USA., Lecture Notes in Computational Science and Engineering, Springer, 2015.

[42] Berzins, M. "Status of Release of the Uintah Computational Framework", Technical Report, No. UUSCI-2012-001, SCI Institute, University of Utah, 2012.

Almost all the publications listed below may be downloaded from
`http://www.sci.utah.edu/scipubs.html?groupFilter=SCI`

RESEARCH FUNDING 2009 onwards PIs in bold

2009-2013	NSF	≈ \$1,000,000	PetaApps	**MB**
2010-2013	DOE	≈ \$750,000	Uintah NETL	**C.R. Johnson**
2012-2017	ARL	≈ \$16M	MSME CRA	**M.B.**
2013-2016	NSF	≈ \$700,000	XPS	**MB**
2014-2019	DOE NNSA	≈ \$16M	PSAAP2 Center	**P.Smith** M.B. CS lead
2016-2017	Naval Res. Lab. Engility	≈ \$400K	Code Opt.	**M.B.** Mike Kirby
2018-2023	ARL	≈ \$16M	MSME CRA	**M.B.**

This research from 2010 to 2016 is made possible by about 1 Billion cpu hours of DOE and NSF computer time awaded under peer review

School of Computing, University of Utah
Salt Lake City, UT-84112
http://www.cs.utah.edu/~bhaskara/
Email. *bhaskara@cs.utah.edu*

ADITYA BHASKARA

RESEARCH INTERESTS

Algorithms and Theoretical Computer Science
Machine Learning and Statistics, Distributed Algorithms

EDUCATION

PRINCETON UNIVERSITY, PRINCETON, NJ — **Sep 2007- Aug 2012**
Ph.D. in Computer Science
Thesis: *Finding Dense Sub-structures in Graphs and Matrices*
Advisor: *Prof. Moses Charikar*

INDIAN INSTITUTE OF TECHNOLOGY, BOMBAY, INDIA
Bachelor of Technology in Computer Science and Engineering — **Jul 2003- May 2007**

EMPLOYMENT

UNIVERSITY OF UTAH — **Jan 2016-present**
Assistant Professor, School of Computing

GOOGLE RESEARCH NYC — **Oct 2013- Oct 2015**
Researcher
Team: Large-scale Graph Mining

ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE (EPFL) — **Sep 2012-Aug 2013**
Researcher

RESEARCH PUBLICATIONS

Detecting High Log-densities: An O($n^{1/4}$)- approximation for Densest k-subgraph
(with Moses Charikar, Eden Chlamtac, Uriel Feige and Aravindan Vijayaraghavan)
ACM Symposium on Theory of Computing (STOC), 2010.

Approximating Matrix p-norms
(with Aravindan Vijayaraghavan)
ACM-SIAM Symposium on Discrete Algorithms (SODA), 2011.

Polynomial Integrality gaps for strong relaxations of Densest k-subgraph
(with Moses Charikar, Venkat Guruswami, Aravindan Vijayaraghavan, and Yuan Zhou)
ACM-SIAM Symposium on Discrete Algorithms (SODA), 2012.

Unconditional Differentially Private Mechanisms for Linear Queries
(with Daniel Dadush, Ravishankar Krishnaswamy and Kunal Talwar)
ACM Symposium on Theory of Computing (STOC), 2012.

On Quadratic Programming with a Ratio Objective
(with Moses Charikar, Rajsekar Manokaran and Aravindan Vijayaraghavan)
International Colloquium on Automata, Languages and Programming (ICALP), 2012.

Minimum Makespan Scheduling with Low-rank Processing Times
(with Ravishankar Krishnaswamy, Kunal Talwar and Udi Wieder)
ACM-SIAM Symposium on Discrete Algorithms (SODA), 2013.

Optimal Hitting Sets for Combinatorial Shapes
(with Devendra Desai and Srikanth Srinivasan)

International Workshop on Randomization and Computation (RANDOM), 2012.
Invited to *Theory of Computing*, 2013.

Smoothed Analysis of Tensor Decompositions
(with Moses Charikar, Ankur Moitra and Aravindan Vijayaraghavan)
ACM Symposium on Theory of Computing (STOC), 2014.

Provable Bounds for Learning Some Deep Representations
(with Sanjeev Arora, Rong Ge and Tengyu Ma)
International Conference on Machine Learning (ICML), 2014.

Uniqueness of Tensor Decompositions and Identifiability in Latent Variable Models
(with Moses Charikar and Aravindan Vijayaraghavan)
Conference on Learning Theory (COLT), 2014.

Centrality of Trees for Capacitated k-Center
(with Hyung-Chan An, Chandra Chekuri, Shalmoli Gupta, Vivek Madan and Ola Svensson)
Integer Programming and Combinatorial Optimization (IPCO), 2014.
Invited to *Mathematical Programming*.

More Algorithms for Provable Dictionary Learning
(with Sanjeev Arora, Rong Ge and Tengyu Ma)
Manuscript, 2014.

Distributed Balanced Clustering via Mapping Coresets
(with MohammadHossein Bateni, Silvio Lattanzi and Vahab Mirrokni)
Neural Information Processing Systems (NIPS), 2014.

Optimizing Display Advertising in Online Social Networks
(with Zeinab Abbassi and Vishal Misra)
International World Wide Web Conference (WWW), 2015.

Sparse Solutions to Nonnegative Linear Systems and Applications
(with Ananda Theertha Suresh and Morteza Zadimoghaddam)
International Conference on Artificial Intelligence and Statistics (AISTATS), 2015.

Expanders via Local Edge Flips
(with Zeyuan Allen-Zhu, Silvio Lattanzi, Vahab Mirrokni and Lorenzo Orecchia)
ACM-SIAM Symposium on Discrete Algorithms (SODA), 2016.

Greedy Column Subset Selection: New Bounds and Distributed Algorithms
(with Jason Altschuler, Thomas Fu, Vahab Mirrokni, Afshin Rostamizadeh and Morteza Zadimoghaddam)
International Conference on Machine Learning (ICML), 2016.

Linear Relaxations for Finding Diverse Elements in Metric Spaces
(with Mehrdad Ghadiri, Vahab Mirrokni and Ola Svensson)
Neural Information Processing Systems (NIPS), 2016.

RECENT TALKS (2014 – PRESENT)

Provable Dictionary Learning via Column Signatures

- Theory Seminar, Carnegie Mellon University **Jan 2014**

Smoothed Analysis of Tensor Decompositions

- Theory Seminar, Rutgers university **Mar 2014**
- ACM Symposium on Theory of Computing **Jun 2014**

New Algorithms for Unsupervised Learning

- Colloquium, University of California, Davis **Apr 2014**

Tensor Decomposition for Latent Variable Models

- Machine Learning Seminar, Google Research, Mountain View **Jun 2014**

Provable Bounds for Learning Some Deep Representations

- Machine Learning Seminar, Google Research, NYC **Aug 2014**

Robust Uniqueness of Tensor Decomposition and Polynomial Identifiability

- Simons Institute for the Theory of Computing, UC Berkeley **Oct 2014**

Distributed Balanced Clustering via Mapping Coresets

- Conference on Neural Information Processing Systems (NIPS) **Dec 2014**

Algorithms for Parameter Estimation in Mixture Models

- Brown University **Oct 2014**
- Boston University **Dec 2014**
- University of California, Irvine **Jan 2015**
- University of Pennsylvania **Jan 2015**
- Information Theory and Applications (ITA), UC San Diego **Feb 2015**

New Algorithmic Techniques in Machine Learning

- University of Utah **Feb 2015**
- University of California, Santa Barbara **Feb 2015**
- University of Illinois at Urbana Champaign **Feb 2015**
- Georgetown University **Mar 2015**
- Purdue University **Mar 2015**
- Simon Fraser University, Canada **Mar 2015**

Expanders via Local Edge Flips

- Microsoft Research India **Dec 2015**
- Workshop on Social Impact through Network Science **Jun 2016**

Learning Gaussian Mixtures – A Survey **Dec 2015**

- Workshop on Clustering, Indian Institute of Science (IISc), India

PROFESSIONAL SERVICE

Recent program committees:

- International Conference on Machine Learning (ICML) 2016
- Conference on Neural Information Processing Systems (NIPS) 2016

Recent conference reviewing (2014 - present):

- ACM-SIAM Symposium on Discrete Algorithms (SODA) 2016
- ACM Symposium on Theory of Computing (STOC) 2015
- Conference on Learning Theory (COLT) 2015
- International World Wide Web Conference (WWW) 2015
- European Symposium on Algorithms (ESA) 2015
- International Workshop on Randomization and Computation (RANDOM) 2015
- International Workshop on Approximation Algorithms (APPROX) 2015
- IEEE Symposium on Foundations of Computer Science (FOCS) 2014
- ACM-SIAM Symposium on Discrete Algorithms (SODA) 2014

Recent journal reviewing (2014 - present):

- Journal of Machine Learning Research (JMLR) 2015
- Mathematics of Operations Research (MOR), 2016
- IEEE Transactions in Signal Processing, 2015
- Combinatorica, 2015
- ACM Transactions on Algorithms (TALG), 2015
- ACM Transactions on Database Systems (TODS), 2015

TEACHING

University of Utah

CS 6968: Techniques in Algorithms and Approximation	**Spring 2016**
CS 6150: Advanced Algorithms	**Fall 2016**
CS 7931: Seminar: Large Scale Machine Learning	**Fall 2016**

Princeton University

COS 487: Theory of Computation	**Fall 2008**
COS 126: General Computer Science	**Spring 2009**

Mahdi Nazm Bojnordi

Assistant Professor, School of Computing, University of Utah
Address: 50 S. Central Campus Drive, Rm. 3418, Salt Lake City, UT 84112
E-Mail: bojnordi@cs.utah.edu – URL: `http://www.cs.utah.edu/~bojnordi/`

Research Interests

Computer architecture, novel memory technologies, and energy-efficient computing.

Education

Ph.D. in Electrical and Computer Engineering — University of Rochester
Dissertation: Memory System Optimizations for Energy and Bandwidth Efficient Data Movement — 2010 - 2016

M.Sc. in Electrical and Computer Engineering — University of Tehran
Thesis: Design and Implementation of Efficient Algorithms for Video Coding — 2003 - 2006

B.Sc. in Computer Science and Engineering — Shiraz University
Project: Designing Synchronous Communication Schemes using VHDL — 1999 - 2003

Honors and Awards

- HPCA 2016 Distinguished Paper Award — 2016
- IEEE Micro Top Picks in Computer Architecture — 2013
- Samsung Best Paper Award — 2012
- Grant Award for Master Thesis from Iran Telecommunication Research Center — 2005
- First rank among all graduating CSE students, Shiraz University — 2003

Professional Experience

School of Computing, University of Utah — Salt Lake City, USA
Assistant Professor — 2016 - present

Samsung Information Systems America — San Jose, USA
Summer Intern — 2012 - 2012

SINA Microelectronics — Tehran, Iran
System Design Engineer — 2009 - 2010

NAJI Research and Development — Tehran, Iran
Hardware Design Engineer — 2007 - 2009

SINA Microelectronics — Tehran, Iran
Design and Verification Engineer — 2004 - 2007

Teaching Experience

University of Utah — Salt Lake City, USA
CS/ECE 6810: Computer Architecture — Fall 2016

CE Department, Qazvin Islamic Azad University — Qazvin, Iran
Fundamentals of Computers, Advanced Programming, Computer Architecture — 2009 - 2010

CE Department, School of Virtual University, Shiraz University — Shiraz, Iran
Data Structures, Assembly and Machine Languages, Scientific and Technical Presentation — 2009 - 2010

CE Department, Elmi-Karbordi University — Tehran, Iran
Computer Architecture, Microprocessors — Fall 2008

CE Department, Shiraz University (International Branch) Shiraz, Iran
Data Structures, Data Communications, Design of Compilers, Theory of Machines and Languages, Object Oriented Programming (JAVA), Design and Implementation of Programming Languages, Advanced Programming (C) 2007 - 2008

Publications

— Ph.D. Research at the University of Rochester —

1. **M.N. Bojnordi**, E. Ipek, *Memristive Boltzmann Machine: A Hardware Accelerator for Combinatorial Optimization and Deep Learning*, International Symposium on High Performance Computer Architecture (**HPCA**), 2016. *Distinguished Paper Award.*
2. **M.N. Bojnordi**, E. Ipek, *DESC: Energy-Efficient Data Exchange using Synchronized Counters*, International Symposium on Microarchitecture (**MICRO**), 2013.
3. **M.N. Bojnordi**, E. Ipek, *A Programmable Memory Controller for the DDRx Interfacing Standards*, ACM Transactions on Computer Systems (**ACM TOCS**), 2013. *Invited paper.*
4. **M.N. Bojnordi**, E. Ipek, *Programmable DDRx Controllers*, IEEE Micro Special Issue: Top Picks from 2012 Computer Architecture Conferences (**IEEE MICRO TOP PICKS**), 2013.
5. **M.N. Bojnordi**, E. Ipek, *PARDIS: A Programmable Memory Controller for the DDRx Interfacing Standards*, International Symposium on Computer Architecture (**ISCA**), 2012. *Selected by IEEE Micro as one of the 11 most significant research papers of 2012 based on novelty and potential for long term impact.*
6. Y. Bai, Y. Song, **M.N. Bojnordi**, A. Shapiro, E.G. Friedman, E. Ipek, *Back To the Future: Current-Mode Processor in the Era of Deeply Scaled CMOS*, Transactions on Very Large Scale Integration Systems (**TVLSI**), 2015.
7. S. Wang, Y. Song, **M.N. Bojnordi**, E. Ipek, *Enabling Energy Efficient Hybrid Memory Cube Systems with Erasure Codes*, International Symposium on Low Power Electronics and Design (**ISLPED**), 2015.
8. Y. Song, **M.N. Bojnordi**, E. Ipek, *Energy-Efficient Data Movement with Sparse Transition Encoding*, International Conference on Computer Design (**ICCD**), 2015.
9. Y. Bai, Y. Song, **M.N. Bojnordi**, A. Shapiro, E. Ipek, E.G. Friedman, *Architecting a MOS Current Mode Logic (MCML) Processor for Fast, Low Noise and Energy-Efficient Computing in the Near-Threshold Regime*, International Conference on Computer Design (**ICCD**), 2015.

— M.Sc. Research at the University of Tehran —

10. **M.N. Bojnordi**, O. Fatemi, M.R. Hashemi, *An Efficient Deblocking Filter with Self-transposing Memory Architecture For H.264/AVC*, International Conference on Acoustics, Speech, and Signal Processing (**ICASSP**), 2006.
11. **M.N. Bojnordi**, M.R. Hashemi, O. Fatemi, *A Fast Two Dimensional Deblocking Filter for H.264/AVC Video Coding*, Canadian Conference on Electrical and Computer Engineering (**CCECE**), 2006.
12. M. Hosseinabady, A. Banaiyan, **M.N. Bojnordi**, Z. Navabi, *A concurrent testing method for NoC switches*, Design, Automation, and Test in Europe Conference (**DATE**), 2006.
13. N. Sedaghati, **M.N. Bojnordi**, S.M. Fakhraie, *MDST: Multiprocessor DSP Simulation Toolkit for Voice Processing Applications*, International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (**MASCOTS**), 2007.
14. M. Hosseinabady, **M.N. Bojnordi**, A. Banayian, Z. Navabi, *An Efficient Online BIST Architecture for NoCs* , European Test Symposium (**ETS**), 2006.
15. **M.N. Bojnordi**, O. Fatemi, M.R. Hashemi, *Dual Mode Architecture for Deblocking Filtering in H.264/AVC Video Coding*, Asia Pacific Conference on Circuits and Systems (**APCCAS**), 2006.
16. **M.N. Bojnordi**, O. Fatemi, M. Semsarzadeh, M.R. Hashemi, *Efficient Hardware Implementation for H.264/AVC Motion Estimation*, Asia Pacific Conference on Circuits and Systems (**APCCAS**), 2006.
17. **M.N. Bojnordi**, N. Sedaghati, O. Fatemi, M.R. Hashemi, *An Efficient Self-Transposing Memory Structure for 32-bit Video Processors*, Asia Pacific Conference on Circuits and Systems (**APCCAS**), 2006.
18. **M.N. Bojnordi**, N. Moezzi-Madani, M. Semsarzadeh, A. Afzali-Kusha, *An Efficient Clocking Scheme for On-Chip Communications*, Asia Pacific Conference on Circuits and Systems (**APCCAS**), 2006.
19. **M.N. Bojnordi**, N. Sedaghati, S.M. Fakhraie, *A Self-testing Fully Pipelined Implementation for the Advanced Encryption Standard*, International Conference on Microelectronics (**ICM**), 2005.
20. N. Sedaghati, **M.N. Bojnordi**, A. Hormati, S.M. Fakhraie, *Efficient modeling of VLIW DSP processors*, Communications in Computer and Information Science (**CCIS**), 2008.

21. N. Sedaghati, **M.N. Bojnordi**, A. Farmahini-Farahani, M. Mousavinejad, S.M. Fakhraie, *Simulation of Voice Processing Applications through VLIW DSP Architectures*, International Conference on Electronics, Circuits and Systems (**ICECS**), 2007.

22. **M.N. Bojnordi**, O. Fatemi, M.R. Hashemi, *Implementing an Efficient Encryption Block for MPEG Video Streams*, International Symposium on Electronics in Marine (**ELMAR**), 2005.

Patent Applications

1. A.S. Jagatheesan, Z. Li, **M.N. Bojnordi**, J. Lee, *Application defined computing component configuration*, US Patent Number 2014/0108773, 2014.

2. **M.N. Bojnordi**, E. Ipek, A.S. Jagatheesan, *Programmable Memory Controller*, US Patent Number 2013/0282972, 2013.

Presentations

Memory System Optimizations for Energy and Bandwidth Efficient Data Movement

- Electrical and Computer Engineering Department, University of Central Florida, Orlando, FL — 3/24/2016
- Electrical and Computer Engineering Department, University of Wisconsin, Madison, WI — 3/14/2016
- School of Computing, University of Utah, Salt Lake City, UT — 3/7/2016
- Laboratory for Laser Energetics, University of Rochester, Rochester, NY — 3/4/2016
- Computer Systems Lab, Cornell University, Ithaca, NY — 3/3/2016

Memory System Architecture for RSFQ Processors

- BBN Workshop on Cryogenic Memory Systems, Hayannis, MA — 6/29/2015

DESC: Energy-Efficient Data Exchange using Synchronized Counters

- International Symposium on Microarchitecture, Davis, CA — 12/10/2013

PARDIS: A Programmable Memory Controller for the DDRx Interfacing Standards

- International Symposium on Computer Architecture, Portland, OR — 6/9/2012

Service

Reviewer for TC11, TVLSI12, Micro12, Micro16, CAL12, TACO13, TACO14, ASPLOS11, DAC15, HotPar15, HPCA12, HPCA13, HPCA15, HPCA16, ICCD11, ICCD13, ICS'16, ISCA12, ISCA13, ISCA14, ISCA15, ISCA16, ISPASS12, MICRO11, MICRO12, MICRO13, MICRO14, MICRO15

Erik Brunvand
Biographical Sketch

School of Computing
50 S. Central Campus Dr., Rm MEB 3190 801-581-4345
University of Utah www.cs.utah.edu\~elb
Salt Lake City, UT 84112 elb@cs.utah.edu

I am an Associate Professor of Computer Science at the University of Utah, Salt Lake City, where my research and teaching interests include the design of application-specific computers, graphics processors, asynchronous systems, and VLSI integrated circuit design. I have also spent time as a visiting scholar in the Digital Arts and Experimental Media (DXARTS) program at the University of Washington (2012), and developed collaborative arts/tech classes at the University of Utah.

Professional Preparation

University of Utah	Computer Science, and Mathematics	BS(2), 1982
University of Utah	Computer Science	M.S., 1984
Carnegie Mellon University	Computer Science	Ph.D, 1991

Professional Appointments

University of Utah	University Professor	2014-2016
University of Utah	Director, Computer Engineering Program	2003-2006, 2009-2012, 2015-present
University of Utah	Associate Professor, School of Computing	1996 - present
University of Utah	Adjunct Associate Professor, ECE Department	1996 - present
University of Utah	Assistant Professor, School of Computing	1990-1996
University of Washington	Visiting Professor, DXARTS Program	2012
Columbia University	Visiting Professor, Computer Science	2006

Selected Recent Publications

Complete list includes 1 book, 12 journal articles, and 77 refereed conference publications

E. Brunvand, “CS+X: Cross Campus Collaborations,” SIGGRAPH 2016 (panel), Jul, 2016.

D. Kopta, K. Shkurko, J. Spjut, E. Brunvand, A. Davis. “Memory Considerations for Low-Energy Ray Tracing,” Computer Graphics Forum, Vol34, No. 1, Feb 2015

E. Brunvand, “Kinetic Sculptures: Creating Programmable Art,” SIGGRAPH 2015, Aug. 2015.

E. Brunvand, “A Noise-Based Curriculum for Technological Fluency,” SIGGRAPH 2015, Aug. 2015.

E. Brunvand, “Technological Fluency through Circuit Bending,” International Conference on Microelectronic Systems Education, Pittsburgh, PA, May 2015.

E. Brunvand, “Speculatorum Oculi,” Leonardo, Vol 47, No. 4, Aug 2014

E. Brunvand, N. Chatterjee, D. Kopta, “Why Graphics Programmers Need to Know about DRAM,” SIGGRAPH 2014 (course), Vancouver, B.C., Canada.

E. Brunvand, S. Brunvand, “Drawing Machines: An Arts and Engineering Collaboration,” NAEA conference, San Diego, CA, Mar 2014

E. Brunvand, “Lights Speed Action: Fundamentals of physical computing for programmers,” SIGGRAPH 2013, Aug. 2013.

D. Kopta, K. Shkurko, J. Spjut, E. Brunvand, A. Davis, “An Energy and Bandwidth Efficient Ray Tracing Architecture,” in High-Performance Computer Graphics (HPG 2013), July 2013.

Konstantin Shkurko, Thiago Ize, Christiaan Gribble, Erik Brunvand, and Lee Butler, “Simulating Radio Frequency Propagation via Ray Tracing,” GPU Technology Conference, March 2013.

D. Kopta, T. Ize, J. Spjut, E. Brunvand, A. Davis, A. Kensler, “Fast, Effective BVH Updates for Animated Scenes,” Symposium on Interactive 3D Graphics and Games (I3D), Mar, 2012.

Joseph Spjut, Daniel Kopta, Erik Brunvand, Al Davis, “A Mobile Accelerator Architecture for Ray Tracing,” Workshop on SoCs, Heterogeneous Arch. and Workloads (SHAW-3), Feb 2012.

E. Brunvand, “Games as Motivation in Computer Design Courses: I/O is the Key,” SIGCSE, Dallas, March 2011.

E. Brunvand, P. Stout, “Kinetic Art and Embedded Systems: A Natural Collaboration,” SIGCSE, Dallas, March 2011.

Erik Brunvand, *Digital VLSI Chip Design with Cadence and Synopsys CAD Tools,* Addison-Wesley, 2010.

D. Kopta, J. Spjut, E. Brunvand, A. Davis, “Efficient MIMD Architectures for High-Performance Ray Tracing,” International Conference on Computer Design (ICCD), Oct 2010.

D. Nellans, K. Sudan, E. Brunvand, R. Balasubramonian, “Improving Server Performance on Multi-Cores via Selective Off-loading of OS Functionality, LNCS 2010

D Nellans, K. Sudan, E. Brunvand, R. Balasubramonian, “Hardware Prediction of OS Run-Length for Fine-Grained Resource Customization,: ISPASS, March 2010.

D. Kopta, J. Spjut, and E. Brunvand, “Grid-based Ray Tracing with CUDA,” ACM/Eurographics High Performance Graphics, New Orleans, August 2009.

J. Spjut, A. Kensler, and E. Brunvand, “Hardware-Accelerated Gradient Noise for Graphics,” ACM Great Lakes Conference on VLSI (GLSVLSI09), May 2009.

J. Spjut, A.Kensler, D. Kopta, and E. Brunvand, “TRaX:, A Multicore Hardware Architecture for Real-Time Ray Tracing,” IEEE Transacations on CAD, Vol 28, n12, Dec 2009.

David Nellans, Rajeev Balasubramonian, and Erik Brunvand, “OS Execution on Multi-Cores: Is Out-Sourcing Worthwhile?” ACM Operating Systems Review, Vol 43, No. 2, April 2009.

Academic Activities

Selected Recent Conference Organization/Keynotes:

VLSI Design conference: Keynote speaker, 2016 (Kolkata, India)

GLSVLSI: Keynote speaker, 2015 (Pittsburgh, PA)

IFIP/IEEE Intl. Conference on VLSI: Keynote speaker 2012 (Santa Cruz, CA)

IEEE Symposium on Asynchronous Circuits and Systems: Program Chair 1994 (Salt Lake City), General Chair 2001 (Salt Lake City), General Chair 2011 (Cornell, NY), Chair of Steering Committee 1994—2008, Member of Steering Committee 2008 — present.

Great Lakes Symposium on VLSI (GLSVLSI): Program Chair: 2010 (Providence, RI), General Chair 2012 (Salt Lake City), Member of Steering Committee 2010 — present.

Conference on Advanced Research on VLSI: General Chair 2001 (Salt Lake City).

Member of program committee for many conferences including: IEEE Symposium on Interactive Ray Tracing, ACM High Performance Graphics, IEEE Symposium on Asynchronous Circuits, SIGGRAPH, Eurographics, GLSVLSI, ACM I3D.

Teaching - Regular courses taught include the following:

CS2050: Making Noise - Sound Art and Digital Media

CS3700 / 3710: Digital Hardware Fundamentals, Computer Design Lab

CS3992 / 4710: Computer Engineering Project Planning, and Senior Project

CS5789: Embedded Systems and Kinetic Art (cross listed with Art 4455)

CS6710 / 6712: Digital VLSI Integrated Circuit Design, Digital VLSI Testing

Selected Teaching Honors and Awards

Two-year University Professorship, University of Utah, 2014-2016.

School of Computing Outstanding Teaching Award, 2012.

Dee Fellowship for interdisciplinary course in arts and technology, Fall 2011 - Spring 2012

University of Utah John R. Park Fellowship, 2006

Nominated for Utah Engineers Council, Utah Engineering Educator of the Year award, 2003.

University of Utah Distinguished Teaching Award, 2002

College of Engineering Outstanding Teaching Award, 1997

Recent Course Develpopment Activities

Developed general education course on technological fluency: *Making Noise: Sound Art and Digital Media* (CS/UGS 2050)

Co-developed a collaborative course with a colleague (Paul Stout) in the Art Department at the University of Utah - *Embedded Systems and Kinetic Art* (CS5789)

Received $25,000 equipment grant for installation of laser cutter in the Computer Engineering Senior Hardware Lab.

Received $8,000 equipment grant for installation of surface mount soldering equipment for the Computer Engineering Senior Hardware Lab.

Selected Recent Funded Research

Architectures for Energy Efficient Ray Tracing, National Science Foundation, Oct 2014 - Sep 2018 ($899,992) Co-PI: Cem Yuksel

University Professorship, Undergraduate College, University of Utah, Aug 2014 - Jul 1016 ($35,000)

University Teaching Assistantship (supporting Nina McCurdy), University of Utah, Aug 2015 - Jul 2016 ($15,000)

Recent Past Projects

Radio Frequency Ray Tracing, Army Research Labs (ARL), Sep 2012 - Sep 2013 ($85,000)

The Big Draw: Art and Engineering Collaboration, Council of Dee Fellows, Aug 2011 - Jul 2012 ($8,400) PI: Paul Stout

Flexible Architectures for Future Graphics Processing Systems, National Science Foundation, Aug 2010 - Jul 2014 ($499,709) Co-PI: Al Davis

University Teaching Assistantship (supporting Josef Spjut), University of Utah, Aug 2010 - July 2011 ($15,000)

Radio Frequency Ray Tracing Evaluation, Army Research Labs (ARL), Feb 2011 - May 2011 ($37,384)

Hardware Support for Real Time Ray Tracing, National Science Foundation, June 2006 - June 2010 ($505,382) Co-PIs: Al Davis, Peter Shirley, Steve Parker

Ray Trace Applications to Radio Frequency (RF) Propagation, Army Research Labs (ARL), Sep 2007 - Dec 2008 ($140,087)

High-Performance Asynchronous Computer Architecture, National Science Foundation, Sep 2002 - Jul 2006 ($325,000)

Students

Current Ph.D. Students - Konstantin Shkurko, Tim Grant.

Graduated Ph.D. students - Daniel Kopta, David Nellans, Josef Spjut, John Hurdle, Luli Josephson (M.Phil.), Ajay Khoche, William Richardson, Jung-Lin Yang

Selected Recent University Service

University Teaching Committee (Chair 2015–present, 2001–2005, Member 2013–2014)

Graduate School Review Committee for the Department of Film and Media Studies, 2014.

Creative Campus Steering Committee 2012–2014

University Research Committee, 2006–2009

College of Engineering Curriculum Committee, 2011 - present

Director, Computer Engineering Program (an ABET-accredited BS degree granting program in the College of Engineering), 2002–2005, 2009–2012, 2015–present.

School of Computing - Director, Computing Track in Computer Engineering, 2007–present

School of Computing - Director, Computing Track in Digital Media, 2011–2014

School of Computing Curriculum Committe, 2011–present

School of Computing Scholarship Committee, 2014–present

School of Computing Director of Graduate Studies, 2006–2008

School of Computing Graduate Admissions Committee, 2008–present

Curriculum Vitae

ELAINE COHEN

June 2016

EDUCATION

Ph.D., Mathematics, Syracuse University, 1974
M.S., Mathematics, Syracuse University, 1970
B.A., Mathematics, Vassar College, 1968

EMPLOYMENT

Spring 2014	Visiting Professor, Institute for Computational Engineering and Science, Univ. of Texas, Austin
Fall 2013	Visiting Professor, Department of Aeronautics and Astronautics, MIT
1/07 to 3/07	Visiting Professor, Center of Mathematics for Applications, University of Oslo, Oslo,
11/06	Visiting Professor, Department of Mechanical & Aerospace Engineering, Arizona State University State University
9/06 – 6/07	Visiting Professor, Computer Science, University of North Carolina – Chapel Hill
7/04-6/07	Adjunct Professor, Mechanical Engineering, University of Utah
Winter 00:	Visiting Professor, Department of Informatikk, University of Oslo, Norway
Fall 99:	Visiting Professor, Department of Computer Science, Technion, Israel.
Since 1991 :	Professor of Computer Science, University of Utah
7/85 to 6/91:	Associate Professor of Computer Science, University of Utah
7/85 to 6/91:	Adjunct Associate Professor of Mathematics, University of Utah
Winter 86:	Visiting Professor, Department of Informatikk, University of Oslo, Norway
7/85 to 6/86:	Visiting Associate Professor of Civil Eng and Computer Science,Princeton University
7/82 to 6/85:	Research Associate Professor of Computer Science, University of Utah
7/82 to 6/85:	Adjunct Associate Professor of Mathematics, University of Utah
1/79 to 7/79:	Visiting Research Scientist, Central Inst. for Industrial Research, Oslo, Norway
9/78 to 12/78:	Visiting Assistant Professor of Computer Science, Carnegie-Mellon University
7/74 to 6/82:	Research Assistant Professor of Computer Science, University of Utah
7/74 to 6/82:	Adjunct Assistant Professor of Mathematics, University of Utah

SKETCH

Cohen has co-headed The Geometric Design and Computation (GDC) Research Group for approximately 30 years. She has focused her research in geometric computation methods and algorithms, computer graphics, geometric modeling, and manufacturing, and isogeometric analysis with emphasis on complex sculptured models represented using NURBS (Non-Uniform Rational B-splines) and NURBS-features. Results in manufacturing research focused on automating process planning, automatic toolpath generation for models having many surfaces, optimizing both within and across manufacturing stages and fixture automation. Research in haptics focused on realistic force feedback in distributed haptic systems for complex mechanical models. Algorithms for proximity include developing new approaches to solving geometric computations such as fast and accurate contact and tracking algorithms for sculptured models and determining accessibility of one object by another. Isogeometric analysis related research has been aimed at parameterizing, fitting, modeling, and querying 3D (volumetric) models, and investigating the appropriateness of specific technical design approaches for various simulations.

PROFESSIONAL ACTIVITIES

Editorial Board member: Graphical Models, 2010 -

Selected Research Organization Committees

Conference co-Chair, Geometric Modeling and Processing (GMP), 2015.
Chair, SIAM Special Interest Group on Geometric Modeling, 1/2013-2/2015.
Conference Chair, SIAM/SMA Solid and Physical Modeling 2013.
Organizer and Chair of Steering Committee, "Workshop on Research Challenges in Computer Graphics", sponsored by the National Research Council, Computer Science and Telecollaboration Board, Dec. 2003.

Selected Program Committee Membership

Solid and Physical Modeling, 2016; Geometric Modeling and Processing, 2016; SIAM/SMA Solid and Physical Modeling, 2015; Solid and Physical Modeling, 2014; Solid and Physical Modeling Conference, 2012; Geometric Modeling and Processling (GMP) Conference, 2012; SIAM/ACM Joint Conference on Geometric and Physical Modeling, 2011; ACM Siggraph Symposium on Interactive Graphics & Games, 2012, 2011, 2010,2009,2007, 2005

Advisory Committees

Member, Scientific Committee, Mathematical Methods for Curves and Surfaces 2016, Tonsberg, Norway, 2015-2016.
Chair, Army Research Office Board of Visitors: Review of all Computer Science Programs, 2014.
Invited Participant, ISAT/DARPA Workshop-Rethinking CAD, October 2013.
Chair, Society for Industrial and Applied Mathematics (SIAM) Activity Group on Geometric Design, 2013-2014.
Executive Committee, Solid Modeling Association, 2008-2012
Army Research Office Board of Visitors: Review of all Computer Science Programs, 2012.
Member, Program Review Committee for the Department of Computer Science, University of North Carolina-Chapel Hill, 2009.
Member, National Research Council, Computer Science and Telecommunications Board, 1999-2004.
Member, Board of Directors, Engineering Geometry Systems, 1995-2005.

Member

Association for Computing Machinery, Society for Industrial and Applied Mathematics

HONORS

2016 Pioneer Award, Solid Modeling Association
2014 John Gregory Memorial Award Schloss Dagstuhl, "in appreciation for Outstanding Contributions in Geometric Modeling"
Keynote speaker, Advances in Computational Mechanics, A Conference Celebrating the 70th Birthday of Thomas J. R. Hughes, Invited Symposium on Geometric Modeling and Mesh Generation for FEM and Isogeometric Analysis, Feb. 24-27, 2013.
Best paper finalist: 2011 SIAM Conference of Geometric and Physical Modeling.
2009 Bezier Award for Solid, Geometric and Physical Modeling and Applications, from the Solid Modeling Association;
Best paper award: 2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling
Best paper award: 2008 ACM Solid and Physical Modeling Symposium;
University of Utah Distinguished Research Award, May 2001.
Selected as one of 50 Computer Graphics Pioneers by ACM Siggraph Society on its 25th anniversary (picture in the Computer Museum)
Selected one of 15 speakers for the First Grace Hopper Celebration of Women in Computing, 1994.

ADVISOR

I have been advisor to 19 graduated PhD students (4 since 2009) and additionally am currently advisor to 1 PhD student and co-advisor to 1 additional student. In additioon I serve as an active committee member of 1 PhD student at University of Texas, Austin, who will graduate summer 2016.

I have been advisor to 21 MS students whose research have been published in journals, and advisor to 3 BA Honors students

GRANTS AND CONTRACTS

I have served as **Principal Investigator** on more than $ 10,205,431 in research grants and contracts, $ 901,459 has been within the most recent time period.

I have served as **co-Principal investigator** on more than $ 15,396,473 in research grants and contracts, $ 991,200 within the most recent time period.

Finally, I have served as **Senior Investigator** on more than $ 10,428,492 in research grants and contracts.

SELECTED PUBLICATIONS

Books

Geometric Modeling with Splines: An Introduction, E. Cohen, R. F. Riesenfeld, and G. Elber, A. K. Peters Press, 2001.

Book Chapters – Refereed: 3 in the covered time period

Journal Articles – Refereed

More than 20 refereed journal publications in the covered time period appearing in top journals in computer aided design, computer graphics, and geometric modeling. The articles below include several of them plus several seminal papers from other time periods.

S. Zeng and E. Cohen, "Hybrid volume completion with higher-order Bézier Elements, " Computer Aided Geometric Design, v. 35-36, pp. 180-191, 2015.

R. Riesenfeld, R. Haimes, and E. Cohen, "Initiating a CAD Renaissance: Multidisciplinary Analysis Driven Design Framework for a New Generation of Advanced Computational Design, Engineering, and Manufacturing Environments, " Computer Methods in Applied Mechanics and Engineering, v 284, pp. 1054-1072, 2015.

X. Gao, T. Martin, S. Deng, E. Cohen, Z. Deng, and G. Chen, "*Structured Volume Decomposition Via Generalized Sweeping*, " IEEE TVCG August, 2015, (epub ahead of print).

A. Barteil and E. Cohen, "Animation of Deformable Bodies with Quadratic Bezier Finite Elements," ACM ToGs, May 2014, v.33, n.3, pp. 27:1-27:10.

E. Cohen, T. Lyche, and R. F. Riesenfeld, "A B-spline-like Basis for the Powell-Sabin 12-Split Based on Simplex Splines,", American Mathematical Society Mathematics of Computation, v. 82, #283, (07/2013), pp. 1667-1707.

T. Martin, G. Chen, S. Musuvathy, E. Cohen and C. Hansen, "Generalized Swept Mid-surface for Polygonal Models," Computer Graphics Forum , v. 31, pp. 805-814, and also Eurographics 2012.

T. Martin, E. Cohen, and R. M. Kirby, "Direct Isosurface Visualization of Hex-Based High-Order Geometry and Attribute Representations," IEEE Transactions on Visualization and Computer Graphics, v. 18,n.5, pp. 753-766, May, 2012.

S. Musuvathy, E. Cohen, and J. N. Damon, "Computing medial axes of generic 3D regions bounded by B-spline surfaces, Computer Aided Design (CAD), v.43,(11) Nov. 2011, Also, best paper finalist, SIAM Conference on Geometric and Physical Modeling 2011.

T. Martin and E. Cohen, “Volumetric Parameterization of Complex Objects by Respecting Multiple Materials,” Computers & Graphics, vol. 34, No. 3, pp. 187-197, 2010.

E. Cohen, T. Martin, M. Kirby, T. Lyche, and R. Riesenfeld, “Analysis-aware Modeling: Understanding Quality Considerations in Modeling for Isogeometric Analysis,” Computer Methods in Applied Mechanics and Engineering, v. 199, issues 5-8, p. 334-356, Jan. 2010.

J. Daniels, C. Silva, and E. Cohen, “Semi-regular, Quad-only Remeshing from Simplified Base Domains,” Computer Graphics Forum, v. 28, n.9,,pp. 1427-1435, 2009.

Cohen, and R. M. Kirby, “Volumetric Parameterization and Trivariate B-spline Fitting using Harmonic Functions,” Computer Aided Geometric Design, vol. 26, pp. 648-664, 2009.

X. Chen, R. F. Riesenfeld, and E. Cohen, “An Algorithm for Direct B-spline Multiplication,”IEEE Transactions on Automation Science and Engineering, pp. 433-442, v. 6, n. 3 (Jul. 2009).

J. Daniels, C. Silva, J. Shepherd, and E. Cohen, “Quadrilateral Mesh Simplification”, J, ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2008), vol. 27, n. 5, 2008..

E. Cohen, T. Lyche, and R. F. Riesenfeld, “Cones and Recurrence Relations for Simplex Splines,” *Journal of Constructive Approximation*, v. 3, n. 1, pp. 131-141, 1987.

E. Cohen, T. Lyche, and L. Schumaker, “Degree Raising for Splines,” *Journal of Approximation Theory*, Vol 46, Feb 1986, 170-181.

E. Cohen, T. Lyche, and R. F. Riesenfeld, “Discrete Box Splines and Refinement Algorithms,” *Computer Aided Geometric Design*, with E. Cohen and T. Lyche, 1(2), pp. 131-148, 1984.

E. Cohen, T. Lyche, and R. F. Riesenfeld, “Discrete B-Splines and Subdivision Techniques in Computer Aided Geometric Design and Computer Graphics,” *Computer Graphics and Image Processing*, Vol. 14, No.2, October 1980.

Refereed Proceedings: 3 in the covered time period

Patents: Co-inventor, *Construction of Spline Surfaces to Provide Inter-Surface Continuity,* Provisional submission, February 2016.

Invited Presentations

22 Invited, Keynote, and Invited Plenary Conference Presentations were given during this most recent time period. In addition, Professor Cohen was an invited speaker at The First Grace Murray Hopper Conference on Women in Computing, Washington, DC, 1994

Classes Taught

All classes were developed by Professor Cohen.

CS4963, spring 2016, Computational Additive Fabrication (New class at Utah); CS6600, fall 2015, fall 2014, spring 2013, spring 2012, Mathematics of Computer Graphics; CS6670, spring 2015, Computer Aided Geometric Design; CS6680, spring 2012, Advanced Computer Aided Geometric Design; CS6961/5, spring 2011, Fundamentals of Visual Computing.

Service

Departmental Committees:

Member, Faculty Recruiting Committee, 2014-2015, 2015-2016; Graduate Fellowship website, fall 2014; Volunteer UG scholarship reader for departmental scholarships for fy2013-2014 (spring 2013); Member, Faculty Recruiting Committee, 2007-2008, 2008-2009.

University Committees

Member, Conflict of Interest Committee, 2005- 2014

External

Invited speaker and panelist: Community College Regional Leadership Forum, "Advancing Women's Leadership." 24-26 Septermber 2008, Snowbird Utah; Invited speaker and panelist, "Jewish Women Scientists," National Council of Jewish Women Annual Fall Membership Brunch, 21 September 2008.

TAMARA DENNING

School of Computing
University of Utah
50 S. Central Campus Drive, Room 3190
Salt Lake City, UT 84112

tdenning@cs.utah.edu
http://www.cs.utah.edu/~tdenning/
Fax: 801-581-5843
Phone: 206-605-3160

RESEARCH INTERESTS

My interests are at the intersection of computer security and human-computer interaction.

PROFESSIONAL PREPARATION

- B.S. in Computer Science, University of California at San Diego, 2007.
- M.S. in Computer Science & Engineering, University of Washington, 2009.
- Ph.D. in Computer Science & Engineering, University of Washington, 2014.
  Advisor: Tadayoshi Kohno

APPOINTMENTS

- Assistant Professor, School of Computing, University of Utah (2014—)

SELECTED AWARDS AND HONORS

- Intel PhD Fellow, 2011.
- Honorable Mention: Computers, Privacy and Data Protection (CPDP) Multidisciplinary Privacy Award ("Patients, Pacemakers, and Implantable Defibrillators: Human Values and Security for Wireless Implantable Medical Devices"), 2011.
- Microsoft Research Women's Scholarship, 2009.
- Marilyn Fries Endowed Regental Fellowship, 2007.
- Achievement Rewards for College Scientists (ARCS) Fellowship, 2007.

PUBLICATIONS

Under submission

[1] Aniqua Z. Baset and Tamara Denning. Going Meta: Reflecting on 36 years of Security and Privacy Research. Submitted to IEEE Symposium on Security and Privacy, 2017.

[2] Aniqua Z. Baset and Tamara Denning. IDE Plugins for Secure Coding: Status and Challenges. Submitted to ACM CHI, 2017.

Peer-reviewed conference papers

[1] Tamara Denning, Batya Friedman, Brian Gill, Daniel B. Kramer, Matthew R. Reynolds, and Tadayoshi Kohno. CPS: Beyond Usability: Applying Value Sensitive Design Based Methods to Investigate Domain Characteristics for Security for Implantable Cardiac Devices. In *Proceedings of Annual Computer Security Applications Conference* (ACSAC), 2014.

[2] Tamara Denning, Zakariya Dehlawi, and Tadayoshi Kohno. In Situ with Bystanders of Augmented Reality Glasses: Perspectives on Recording and Privacy-Mediating Technologies. In *Proceedings of the International Conference on Human Factors in Computing Systems* (CHI '14), 2014.

[3] Tamara Denning, Alan Borning, Batya Friedman, Brian Gill, Tadayoshi Kohno, and William H. Maisel. Patients, Pacemakers, and Implantable Debrillators: Human Values and Security for Wireless Implantable Medical Devices. In *Proceedings of the International Conference on Human Factors in Computing Systems* (CHI '10), 2010.

[4] Tamara Denning, Adam Lerner, Adam Shostack, and Tadayoshi Kohno. Control-Alt-Hack: The Design and Evaluation of a Card Game for Computer Security Awareness and Education. In *Proceedings of ACM Conference on Computer and Communications Security* (CCS '13), 2013.

[5] Tamara Denning, Kevin Bowers, Marten van Dijk, and Ari Juels. Exploring Implicit Memory for Painless Password Recovery. In *Proceedings of the International Conference on Human Factors in Computing Systems* (CHI '11), 2011.

[6] Tamara Denning, Cynthia Matuszek, Karl Koscher, Joshua R. Smith, and Tadayoshi Kohno. A Spotlight on Security and Privacy Risks with Future Household Robots: Attacks and Lessons. In *Proceedings of the International Conference on Ubiquitous Computing* (UbiComp '09), 2009.

[7] David Lindquist, Tamara Denning, Michael Kelly, Roshni Malani, William G. Griswold, and Beth Simon. Exploring the Potential of Mobile Phones for Active Learning in the Classroom. In *Proceedings of the Special Interest Group on Computer Science Education Technical Symposium* (SIGCSE '07), 2007.

[8] Tamara Denning, Michael Kelly, David Lindquist, Roshni Malani, William G. Griswold, and Beth Simon. Lightweight Preliminary Peer Review: Does In-Class Peer Review Make Sense? In *Proceedings of the Special Interest Group on Computer Science Education Technical Symposium* (SIGCSE '07), 2007.

[9] Tamara Denning, William Griswold, Beth Simon, and Michelle Wilkerson. Multimodal Communication in the Classroom: What Does It Mean For Us? In *Proceedings of the Special Interest Group on Computer Science Education Technical Symposium* (SIGCSE '06), 2006.

Peer-reviewed journal articles

[1] Tamara Denning, Yoky Matsuoka, and Tadayoshi Kohno. Neurosecurity: Security and Privacy for Neural Devices. *Neurosurgical Focus*, 27(1), July 2009.

Peer-reviewed technical magazine articles

[1] Tamara Denning, Tadayoshi Kohno, Henry M. Levy. Computer Security and the Modern Home. *Communications of the ACM*, 56 (1), January 2013, 94–103.

Technical Magazine Articles/Columns

[1] Jane Cleland-Huang, Tamara Denning, Tadayoshi Kohno, Forrest Shull, and Samuel Weber. Keeping Ahead of Our Adversaries. *IEEE Software* 33(3), 2016.

[2] Jelena Mirkovic, Melissa Dark, Wenliang Du, Giovanni Vigna, and Tamara Denning. Evaluating Cybersecurity Education Interventions: Three Case Studies. *IEEE Security & Privacy* 13(3), 2015.

[3] Mark Gondree, Zachary N.J. Peterson, and Tamara Denning. Security Through Play. *IEEE Security & Privacy*, 11(3), 2013.

Technology & policy primers

[1] Ryan Calo, Tamara Denning, Batya Friedman, Tadayoshi Kohno, Lassana Magassa, Emily McReynolds, Bryce Newell, Franziska Roesner, and Jesse Woo. Augmented Reality: A Technology and Policy Primer. Tech Policy Lab, University of Washington, 2015.

Workshop papers

[1] Franziska Roesner, Tamara Denning, Bryce C. Newell, Tadayoshi Kohno, and Ryan Calo. Augmented Reality: Hard Problems of Law and Policy. In *Proceedings of the Workshop on Usable Privacy & Security for wearable and domestic ubIquitous DEvices* (UPSIDE '14), 2014.

[2] Tamara Denning, Adam Shostack, Tadayoshi Kohno. Practical Lessons From Creating the Control-Alt-Hack Card Game and Research Challenges for Games In Education and Research. In *Proceedings of the USENIX Summit on Gaming, Games and Gamification in Security Education* (3GSE '14), 2014.

[3] Tamara Denning, Adrienne Andrew, Rohit Chaudhri, Carl Hartung, Jonathan Lester, Gaetano Borriello, and Glen Duncan. BALANCE: Towards a Usable Pervasive Wellness Application with Accurate Activity Inference. In *Proceedings of the International Workshop on Mobile Computing Systems and Applications* (HotMobile '09), 2009.

[4] Tamara Denning, Kevin Fu, and Tadayoshi Kohno. Absence Makes the Heart Grow Fonder: New Directions for Implantable Medical Device Security. In *Proceedings of the USENIX Workshop on Hot Topics in Security* (HotSec '08), 2008.

Technical reports

[1] Tamara Denning, Tadayoshi Kohno, and Adam Shostack. Control-Alt-Hack^{TM}: A Card Game for Computer Security Outreach, Education, and Fun. Technical Report UW-CSE-12-07-01, University of Washington Computer Science & Engineering, July 2012.

Dissertation

[1] Tamara Denning. Human-Centric Security and Privacy for Emerging Technologies. Computer Science & Engineering, University of Washington, 2014.

INTERNSHIPS

- Research Intern. RSA Labs (Cambridge, MA), 2010.
- Intern. Naval Research Laboratory (Monterey, CA), 2004.

PROFESSIONAL ACTIVITIES

Program committees

- 24th USENIX Security Symposium (USENIX), 2016
- 24th USENIX Security Symposium (USENIX), 2015
- 2nd USENIX Summit on Games, Gaming, and Gamification in Security Education (3GSE), 2015
- 9th World Conference on Information Security Education (WISE), 2015
- 24th International World Wide Web Conference (WWW), 2015
- NDSS Workshop on Usable Security (USEC), 2015
- Workshop on Home Usable Privacy and Security (HUPS), 2013

Funding Panels

- Secure and Trustworthy Cyberspace (SaTC), National Science Foundation, 2015.

FUNDING

Awarded

- Line-Funding with the Software Engineering Institute (SEI) at Carnegie Mellon University (CMU)
  Title: Evaluation of Threat Modeling Methodologies
  SEI Team Members: Sam Weber (CERT/PI), Forrest Shull (SSD/PI), Nancy Mead (CERT). Other Collaborators: Jane Cleland-Huang (DePaul University), Tadayoshi Kohno (University of Washington, Tamara Denning (University of Utah)
  Submission Date: 2/25/2015
  Funding Duration: 1 year
  Total Proposal Amount: $750,000
  My Amount: $50,000
  Status: Pending
- Principal Investigator on Special Interest Group on Computer Science Education (SIGCSE) Special Projects Grant, 2012.

PRESENTATIONS

Invited panels

1. NSF Secure and Trustworthy Computing (SaTC) PI Meeting: The Future of Privacy, 2015.
2. ACM Symposium on Access Control Models and Technologies (SACMAT), 2014.

Invited talks

1. Security Cards: A Threat Brainstorming Toolkit. Innovations in Cybersecurity Education (ICEW), UMBC, 2016.
2. Human-Centered Computer Security and Privacy. NAS Kavli Frontiers of Science U.S. Symposium, 2015.
3. Security Cards: A Threat Brainstorming Toolkit. GREPSEC, 2015.
4. Human-Centered Computer Security and Privacy. Team for Research in Ubiquitous Secure Technology (TRUST), UC Berkeley, 2015.
5. Control-Alt-Hack^{TM}: White Hat Hacking for Fun and Profit (A Computer Security Card Game). ZonCon, 2013. Presented with Tadayoshi Kohno and Adam Shostack.
6. Control-Alt-Hack^{TM}: White Hat Hacking for Fun and Profit (A Computer Security Card Game). NSF Scholarship for Service Cybersecurity Job Fair, 2013.
7. Human-Centered Design Of Security Systems for Implantable Medical Devices. International Federation for Information Processing Working Group (IFIP WG) 10.4 Meeting, 2012.

TEACHING AND EDUCATION

Instructor

- CS 4964. Introduction to Computer Security. Spring 2016.
- CS 6964. Computer Security Research. Fall 2015.
- CS 4964. Computer Security. Spring 2015.
- CS 6964. Human-Centered Research: Security, Privacy, and Society. Fall 2014.

Eric N. Eide

University of Utah
School of Computing
50 South Central Campus Drive, Room 3190
Salt Lake City, UT 84112–9205

eeide@cs.utah.edu
http://www.cs.utah.edu/~eeide/
Office: +1 801 585 5512
Fax: +1 801 585 3743

Education

Ph.D., Computer Science, University of Utah. Advisor: Prof. Matthew Flatt. Dissertation title: "Software Variability Mechanisms for Improving Run-Time Performance." December 2012.

M.S., Computer Science, University of Utah. Advisor: Prof. Robert R. Kessler. Thesis title: "Valet: An Intelligent UNIX Shell Interface." August 1995.

B.S., Computer Science, University of Utah. Summa cum laude. June 1989.

Seelcted Professional Experience

Research Assistant Professor, School of Computing, University of Utah, July 2013 – present.

Co-Director of the Flux Research Group, School of Computing, University of Utah, September 2008 – present.

Research Associate, Research Staff Member, Project Engineering Manager, and IT Project Manager, School of Computing, University of Utah, June 1996 – June 2013.

Selected Teaching Experience

Instructor, CS 7934, Computer Systems Seminar, School of Computing, University of Utah, 16 semesters. (Fall 2008, Fall 2009, Spring 2010, Fall 2010, Spring 2011, Fall 2011, Spring 2012, Fall 2012, Spring 2013, Fall 2013, Spring 2014, Fall 2014, Spring 2015, Fall 2015, Spring 2016, Fall 2016.) **Received Dean's letter for teaching excellence, Fall 2010.**

Instructor, CS 6950, Independent Study, School of Computing, University of Utah, 2 semesters. (Summer 2015, Fall 2016.)

Selected Grants and Contracts

National Science Foundation Computing and Communication Foundations: Core Programs. *SHF: Small: Xsmith, A Configurable Generator of Highly Effective Fuzz Testers.* Eric Eide (PI) and John Regehr (co-PI). Award CCF–1527638. September 2015 – August 2018. Award amount: $499,998.

National Science Foundation CISE Research Infrastructure Program. *CI–EN: Revitalizing Emulab for Modern Networking and Systems Research.* Eric Eide (PI) and Robert Ricci (co-PI). Award CNS–1513121. July 2015 – June 2018. Award amount: $2,199,450.

National Science Foundation CISE Research Infrastructure Program. *CloudLab: Flexible Scientific Infrastructure to Support Fundamental Advances in Cloud Architectures and Applications.* Robert Ricci (PI), Srinivasa A. Akella (co-PI), Brig "Chip" Elliott (co-PI), Kuang-Ching Wang (co-PI), Michael H. Zink (co-PI), Eric Eide (key personnel), Mike Hibler (key personnel), Linh Ngo (key personnel), James Pepin (key personnel), James von Oehsen (key personnel), and David E. Irwin (key personnel). Award CNS–1419199. October 2014 – September 2017. Award amount: $10,999,999.

National Science Foundation Major Research Instrumentation Program. *MRI: Development of Apt, a Testbed Instrument With Adaptable Profiles for Network and Computational Science.* Robert Ricci (PI), Steve Corbató (co-PI), Eric Eide (co-PI), Julio Facelli (co-PI), and Jacobus Van der Merwe (co-PI). Award CNS–1338155. October 2013 – September 2017. Award amount: $3,400,000 including University of Utah cost-share amount.

National Science Foundation Secure and Trustworthy Cyberspace Program. *TWC: Medium: TCloud: A Self-Defending, Self-Evolving and Self-Accounting Trustworthy Cloud Platform.* Award CNS–1314945. Jacobus Van der Merwe (PI), Eric Eide (co-PI), Feifei Li (co-PI), and Robert Ricci (co-PI). September 2013 – August 2017. Award amount: $999,991.

Selected Refereed Publications

Anton Burtsev, David Johnson, Mike Hibler, Eric Eide, and John Regehr. Abstractions for practical virtual machine replay. In *Proceedings of the 12th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE)*, pages 93–106, Atlanta, GA, April 2016.

Richard Li, Dallin Abendroth, Xing Lin, Yuankai Guo, Hyun-wook Baek, Eric Eide, Robert Ricci, and Jacobus Van der Merwe. Potassium: Penetration testing as a service. In *Proceedings of the 6th ACM Symposium on Cloud Computing (SoCC)*, pages 30–42, Kohala Coast, HI, August 2015.

Xing Lin, Mike Hibler, Eric Eide, and Robert Ricci. Using deduplicating storage for efficient disk image deployment. In *Proceedings of the 10th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM)*, Vancouver, BC, June 2015.

David Johnson, Mike Hibler, and Eric Eide. Composable multi-level debugging with Stackdb. In *Proceedings of the 10th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE)*, pages 213–225, Salt Lake City, UT, March 2014.

Alex Groce, Chaoqiang Zhang, Mohammad Amin Alipour, Eric Eide, Yang Chen, and John Regehr. Help, help, I'm being suppressed! The significance of suppressors in software testing. In *Proceedings of the 24th IEEE International Symposium on Software Reliability Engineering (ISSRE)*, pages 390–399, Pasadena, CA, November 2013.

Anton Burtsev, Nikhil Mishrikoti, Eric Eide, and Robert Ricci. Weir: A streaming language for performance analysis. In *Proceedings of the 7th Workshop on Programming Languages and Operating Systems (PLOS)*, Farmington, PA, November 2013.

Aaron Paulos, Partha Pal, Richard Schantz, Brett Benyo, David Johnson, Mike Hibler, and Eric Eide. Isolation of malicious external inputs in a security focused adaptive execution environment. In *Proceedings of the 8th International Conference on Availability, Reliability and Security (ARES)*, pages 82–91, Regensburg, Germany, September 2013.

Yang Chen, Alex Groce, Chaoqiang Zhang, Weng-Keen Wong, Xiaoli Fern, Eric Eide, and John Regehr. Taming compiler fuzzers. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pages 197–208, Seattle, WA, June 2013.

Partha Pal, Richard Schantz, Aaron Paulos, Brett Benyo, David Johnson, Mike Hibler, and Eric Eide. A3: An environment for self-adaptive diagnosis and immunization of novel attacks. In *Proceedings of the 6th IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW)*, pages 15–22, Lyon, France, September 2012.

Alex Groce, Chaoqiang Zhang, Eric Eide, Yang Chen, and John Regehr. Swarm testing. In *Proceedings of the 2012 International Symposium on Software Testing and Analysis (ISSTA)*, pages 78–88, Minneapolis, MN, July 2012.

John Regehr, Yang Chen, Pascal Cuoq, Eric Eide, Chucky Ellison, and Xuejun Yang. Test-case reduction for C compiler bugs. In *Proceedings of the 33rd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pages 335–346, Beijing, China, June 2012.

Selected Awards and Scholarships

Funded Research Honoree, Celebrate U: A Showcase of Extraordinary Faculty Achievements, 2015.

Dean's letter for teaching excellence, 2010.

Selected Talks

"Finding and Understanding Bugs in C Compilers." Invited talk at the Department of Computer Science and Engineering, New Mexico Institute of Mining and Technology, October 2016.

Invited panelist for discussion of research reproducibility, PHIL 7570 (Research Ethics) at the University of Utah, September 2016.

"Network Testbeds and Repeatable Research." Invited talk at Dagstuhl Seminar 16111 (Rethinking Experimental Methods in Computing), Wadern, Germany, March 2016.

"A Look at Utah's Network Testbeds and Their Support for Repeatable Research." Invited talk at LORIA (Lorraine Laboratory of Research in Computer Science and its Applications), January 2016.

"Random Testing of C Compilers." Guest lecture for CS 5959 (Writing Solid Code) at the University of Utah, November 2015.

Invited guest, KPCW's "Cool Science Radio" program, December 2014.

"Cybersecurity Experimentation of the Future." Invited panelist, 7th Workshop on Cyber Security Experimentation and Test (CSET), August 2014.

Selected Professional Activities

Organizing Committee Member

Finance Chair, 23rd Annual International Conference on Mobile Computing and Networking (MobiCom), 2017.

Program Co-Chair, 4th International Workshop on Computer and Networking Experimental Research Using Testbeds (CNERT), 2017.

Program Co-Chair, 9th Workshop on Cyber Security Experimentation and Test (CSET), 2016.

Artifact Evaluation Committee Co-Chair, 36th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), 2015.

Artifact Evaluation Committee Co-Chair, 35th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), 2014.

Steering Committee Member, 7th Workshop on Programming Languages and Operating Systems (PLOS), 2013.

Program Committee Member

36th IEEE International Conference on Distributed Computing Systems (ICDCS), track on Cloud Computing and Data Center Systems, 2016.

3rd International Workshop on Computer and Networking Experimental Research Using Testbeds (CNERT), 2016.

8th Workshop on Programming Languages and Operating Systems (PLOS), 2015.

2nd International Workshop on Computer and Networking Experimental Research Using Testbeds (CNERT), 2015.

7th Workshop on Cyber Security Experimentation and Test (CSET), 2014.

1st International Workshop on Reproducible Research Methodologies and New Publication Models in Computer Engineering (TRUST), 2014.

1st Conference on Timely Results in Operating Systems (TRIOS), 2013.

32nd IEEE International Conference on Distributed Computing Systems (ICDCS), track on Distributed Operating Systems and Middleware, 2012.

6th Workshop on Programming Languages and Operating Systems (PLOS), 2011.

Editor

Guest editor, *Operating Systems Review*, 49(1), January 2015. Special issue on repeatability and sharing of experimental artifacts.

External Journal Reviewer

Software: Practice and Experience, 2011, 2012, 2013, 2014.

External Conference Reviewer

ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2017.

ACM SIGPLAN Symposium on Principles of Programming Languages (POPL), 2017.

International Conference on Computer Aided Verification (CAV), 2015.

Departmental Service

Member of graduate admissions committee for systems area, University of Utah School of Computing, 2010, 2012, 2013, 2014, 2015, 2016.

Member of faculty recruiting committee for operating systems and distributed systems area, University of Utah School of Computing, 2013–2014.

Other Service

Member of ACM Special Interest Group Governing Board Replication Taskforce, 2015–.

Proposal review panelist, NSF Secure and Trustworthy Cyberspace (SaTC) Program, 2015, 2016.

USENIX Campus Representative to the University of Utah, 2012–.

Matthew Flatt

Education

99	Ph.D. Computer Science Dissertation: *Programming Languages for Reusable Software Components* Advisor: Matthias Felleisen	Rice University
98	M.S. Computer Science	Rice University
93	B.S. Math and Computer Science, B.S. Physics	Carnegie Mellon University

Professional Experience

14-	Professor	University of Utah
06-14	Associate Professor	University of Utah
00-06	Assistant Professor	University of Utah
99-00	Research Assistant Professor	University of Utah
14	Visiting Researcher	Microsoft Research, Cambridge, UK
06-07	Visiting Researcher	Institute of Software, Chinese Academy of Sciences

Recent Publications

Books

[1] 10 SPERBER, M., R.K. DYBVIG, M. FLATT, AND A. VAN STRAATEN, EDITORS. Revised^6 Report on the Algorithmic Language Scheme. Cambridge University Press, 301 pages.

[2] 09 FELLEISEN, M., R.B. FINDLER, AND M. FLATT. *Semantics Engineering with PLT Redex*, MIT Press, 528 pages.

Journal Publications and Book Chapters

[3] 13 KLEIN, C., M. FLATT, AND R.B. FINDLER. The Racket Virtual Machine and Randomized Testing. *Higher Order and Symbolic Programming*.

[4] 12 FLATT, M., R. CULPEPPER, D. DARAIS, AND R.B. FINDLER. Macros that Work Together: Compile-Time Bindings, Partial Expansion, and Definition Contexts. *Journal of Functional Programming*, 22(2), pages 181–216.

Conference Publications

[5] 16 M.FLATT. Binding as Sets of Scopes. In *Proc. ACM Symposium on Principles of Programming Languages* (POPL).

[6] 15 FLORENCE, S., B. FETSCHER, M. FLATT, W.H. TEMPS, T. KIGURADZE, D.P. WEST, C. NIZNIK, P.R. YARNOLD, R.B. FINDLER, AND S.M. BELKNAP. POP-PL: A Patient-Oriented Prescription Programming Language. In *Proc. Conference on Generative Programming: Concepts and Experiences* (GPCE).

[7] 15 TAKIKAWA, A., D. FELTEY, E. DEAN, R.B. FINDLER, M. FLATT, S. TOBIN-HOCHSTADT, AND M. FELLEISEN. Towards Practical Gradual Typing. In *Proc. European Conference on Object-Oriented Programming* (ECOOP).

[8] 15 FELLEISEN, M., R.B. FINDLER, M. FLATT, S. KRISHNAMURTHI, J. MCCARTHY, S. TOBIN-HOCHSTADT. The Racket Manifesto. In *Proc. Summit on Advances in Programming Languages* (SNAPL).

[9] 13 TEW, K., J. SWAINE, M. FLATT, R.B. FINDLER, AND P. DINDA. Distributed Places. In *Proc. Trends in Functional Programming* (TFP).

[10] 13 FLATT, M. Submodules in Racket: You Want it *When*, Again? In *Proc. Conference on Generative Programming: Concepts and Experiences* (GPCE).

[11] 12 STRICKLAND, T.S., S. TOBIN-HOCHSTADT, R.B. FINDLER, AND M. FLATT. Chaperones and Impersonators: Run-time Support for Reasonable Interposition. In *Proc. ACM Conference on Object-Oriented Programming Systems, Languages, and Applications* (OOPSLA).

[12] 12 RAFKIND, J. AND M. FLATT. Honu: Syntactic Extension for Algebraic Notation through Enforestation. In *Proc. Conference on Generative Programming and Component Engineering* (GPCE).

[13] 12 ST-AMOUR, V., S. TOBIN-HOCHSTADT, M. FLATT, AND M. FELLEISEN. Typing the Numeric Tower. In *Proc. Practical Aspects of Declarative Languages* (PADL).

[14] 12 KLEIN, C., J. CLEMENTS, C. DIMOULAS, C. EASTLUND, M. FELLEISEN, M. FLATT, J. MCCARTHY, J. RAFKIND, S. TOBIN-HOCHSTADT, AND R.B. FINDLER. Run Your Research: On the Effectiveness of Lightweight Mechanization. In *Proc. ACM Symposium on Principles of Programming Languages* (POPL).

[15] 11 TEW, K., J. SWAINE, M. FLATT, R.B. FINDLER, AND P. DINDA. Places: Adding Message-Passing Parallelism to Racket. In *Proc. Dynamic Languages Symposium* (DLS).

[16] 11 TOBIN-HOCHSTADT, S., V. ST-AMOUR, R. CULPEPPER, M. FLATT, AND M. FELLEISEN. Languages as Libraries. In *Proc. ACM Conference on Programming Language Design and Implementation* (PLDI).

[17] 10 ATKINSON, K., M. FLATT, AND G. LINDSTROM. Using Macros to Address ABI Compatibility. In *Proc. Conference on Generative Programming and Component Engineering* (GPCE).

[18] 10 SWAINE, J., K. TEW, P. DINDA, R.B. FINDLER, AND M. FLATT. Back to the Futures: Incremental Parallelization of Existing Sequential Runtime Systems. In *Proc. ACM Conference on Object-Oriented Programming Systems, Languages, and Applications* (OOPSLA).

[19] 10 KLEIN, C., M. FLATT, AND R.B. FINDLER. Random Testing for Higher-Order, Stateful Programs. In *Proc. ACM Conference on Object-Oriented Programming Systems, Languages, and Applications* (OOPSLA).

[20] 09 FLATT, M., E. BARZILAY, AND R.B. FINDLER. Scribble: Closing the Book on Ad Hoc Documentation Tools. In *Proc. ACM International Conference on Functional Programming* (ICFP), pages 109–120.

[21] 09 FELLEISEN, M., R.B. FINDLER, M. FLATT, AND S. KRISHNAMURTHI. A Functional I/O System (*or* Fun for Freshman Kids). In *Proc. ACM International Conference on Functional Programming* (ICFP), pages 47–58.

[22] 09 RAFKIND, J., A. WICK, J. REGEHR, AND M. FLATT. Precise Garbage Collection for C. In *Proc. ACM International Symposium on Memory Management* (ISMM), pages 39–48.

[23] 07 FLATT, M., G. YU, R.B. FINDLER, AND M. FELLEISEN. Adding Delimited and Composable Control to a Production Programming Environment. In *Proc. ACM International Conference on Functional Programming* (ICFP).

Workshop Papers

[24] 15 FLORENCE, S., R. CULPEPPER, M. FLATT, AND R.B. FINDLER Check Syntax: An Out-of-the-Box Tool for Macro-Based DSLs. In *Domain Specific Language Design and Implementation.*

[25] 12 SWAINE, J., B. FETSCHER, R.B. FINDLER, AND M. FLATT Seeing the Futures: Profiling Shared-Memory Parallel Racket. In *Workshop on Functional High-Performance Computing.*

[26] 11 ATKINSON, K. AND M. FLATT. Adapting Scheme-Like Macros to a C-Like Language. In *Proc. Scheme Workshop.*

[27] 11 BARZILAY, E., R. CULPEPPER, AND FLATT, M. Keeping it Clean with `syntax-parameterize`. In *Proc. Scheme Workshop.*

[28] 10 BARLAND, I., R.B. FINDLER, AND M. FLATT. The Design of a Functional Image Library. In *Proc. Scheme Workshop*.

[29] 09 FLATT, M. AND E. BARZILAY Keyword and Optional Arguments in PLT Scheme. In *Proc. Scheme Workshop*.

[30] 07 CULPEPPER, R., S. TOBIN-HOCHSTADT, AND M. FLATT. Advanced Macrology and the Implementation of Typed Scheme. In *Proc. Scheme Workshop*.

Recent Research Support

15-18 ROBERT BRUCE FINDLER (LEAD PI), STEVEN M. BELKNAP, MATTHEW FLATT (LOCAL PI), DENNIS P. WEST. NSF CCF SHF program, $120,000 (total for all sites: $400,000).

14-17 ROBERT BRUCE FINDLER (LEAD PI), MATTHIAS FELLEISEN, MATTHEW FLATT (LOCAL PI). NSF CI program, $200,000 (total for all sites: $900,000).

12-14 ROBERT BRUCE FINDLER (LEAD PI), STEVEN M BELKNAP, MATTHEW FLATT (LOCAL PI). SHF: Small: Collaborative Research: Designing a Patient-Oriented Prescription Language: An Executable Medical Algorithm for Gestational Diabetes Mellitus. NSF CCF SHF program, $24,903 (total for all sites: $498,699).

10-14 OLIN SHIVERS (LEAD PI), MATTHIAS FELLEISEN, PETE MANOLIOS, MITCH WAND, MATTHEW MIGHT (LOCAL PI), MATTHEW FLATT. GnoSys: Raising the Level of Discourse in Systems Programming. DARPA CRASH program, $1,500,000 (total for all sites: around $5,000,000).

09-13 MATTHEW FLATT (PI). An Extensible Gradual Type System via Compile-Time Meta-Programming. NSF CCF SHF program, $419,000.

09-11 MATTHIAS FELLEISEN (PI), ROBERT BRUCE FINDLER, MATTHEW FLATT (LOCAL PI), SHRIRAM KRISHNAMURTHI. Infrastructure for the Production of Languages. NSF CI-ANNDO-EN, $134,415 (total for all sites: $660,000).

06-10 STEPHEN BLOCH (PI), JOHN B. CLEMENTS, KATHI FISLER, MATTHEW FLATT (LOCAL PI), VIERA K. PROULX. Redesigning Introductory Computing: The Design Discipline. NSF DUE program, $20,000 (total for all sites: $499,688, mostly to support workshop participants across all sites).

Software

Racket with PLT `http://www.racket-lang.org/`
95-present

Recent Invited Talks and Instruction

16	Keynote	Workshop on Partial Evaluation and Program Manipulation (PEPM)	St. Petersburgh, FL
15	Summer School	DSL Design and Implementation	Lausanne, Switzerland
14	Colloquium	University of Kent	Canterbury, UK
14	Seminar	University of Cambridge	Cambridge, UK
14	Invited Talk	Mozilla	San Francisco, CA
13	Invited Talk	Control Operators and their Semantics (COS)	Eindhoven, Netherlands
13	Invited Talk	Workshop on Trends in Functional Programming in Education (TFPIE)	Provo, UT
13	Keynote	Clojure/West	Portland, OR
13	Colloquium	Vrije Universiteit Brussel	Brussels, Belgium
13	Invited Tutorial	TutorialFest at POPL	Rome, Italy
13	Colloquium	Northwestern University	Evanston, IL
12	Talk	Strange Loop Conference	St. Louis, MO
12	Colloquium	Chinese Academy of Sciences, Institute of Software	Beijing, China
12	Colloquium	Sun-Yat Sen University	Guangzhou, China

11	Invited Tutorial	ECOOP Summer School	Lancaster, England
09	Colloquium	Brigham Young University	Provo, UT

Recent Teaching

S16	Instructor for CS 5965/6965, Functional Programming Studio	University of Utah
S08, S10, S12, S14, S16	Instructor for CS 6510, Functional Programming	University of Utah
F07-11,13,15	Instructor for CS 5510, Programming Languages	University of Utah
F12	Instructor for CS 4400, Computer Systems	University of Utah
S11-12	Instructor for CS 2420-20, Computer Science II (systems)	University of Utah
F10-11	Instructor for CS 1410-20, Computer Science I (systems)	University of Utah
F09	Instructor for CS 5460/6460, Operating Systems	University of Utah
S09, S11, S13	Instructor for CS 6520/7520, Programming Languages and Semantics	University of Utah
F08	Instructor for CS 4960-01, Parallel Programming	University of Utah
S08	Instructor for CS 5967, How to Design Programs	University of Utah

Post-Doc Advising

12-13	Danny Yoo
11-12	Ryan Culpepper

Thesis Committee Charing

15-	Chris Brooks	PhD thesis committee	Chair/Advisor
15-	William Hatch	PhD thesis committee	Chair/Advisor
12-	Xiangqi Li	PhD thesis committee	Chair/Advisor
07-13	Jon Rafkind	PhD thesis committee	Chair/Advisor
07-12	Kevin Tew	PhD thesis committee	Chair/Advisor
06-09	Eric Eide	PhD thesis committee	Chair/Advisor
05-11	Kevin Atkinson	PhD thesis committee	Co-chair/Co-advisor
00-06	Kathy Gray	PhD thesis committee	Chair/Advisor
00-06	Scott Owens	PhD thesis committee	Co-chair/Co-advisor
00-06	Adam Wick	PhD thesis committee	Chair/Advisor

Program and Artifact Evaluation Committee Chairing

18	ACM Intl. Conf. on Functional Programming (ICFP)	Chair
16	SPLASH Doctoral Symposium	Chair
15-16	European Conf. on Object-Oriented Programming (ECOOP)	Co-Chair
14	ACM Conf. on Generative Programming: Concepts & Experiences (GPCE)	Chair
14	Sym. on Practical Aspects of Declarative Languages (PADL)	Co-Chair

Awards

14	Distinguished Scientist	ACM
09	Outstanding Teaching Award	School of Computing, University of Utah
05	Early Career Teaching Award	University of Utah

H. James de St. Germain
germain@cs.utah.edu
February, 2016

OFFICE
University of Utah
School of Computing
50 South Central Campus Drive
3190b Merrill Engineering Building
Salt Lake City, UT 84112
(801) 585-3352

HOME
1465 South, 1600 East
Salt Lake City, UT 84105
(801) 979-1283

Professional Experience	**2012-present**	Associate Professor, Lecturing School of Computing, Utah
	2005-present	Director of Undergraduate Studies, SoC, Utah
	2005-2012	Assistant Professor, Clinical, SoC, Utah
	2005	Consulting – Visual Influence
Professional Service	**2006-present**	College of Engineering Curriculum Committee (Chair 2008-) College of Engineering Articulation Committee UofU Curriculum Policy Review Board Committee UofU Curriculum Fees Committee School of Computing Curriculum Committee Web Learning System Administrator and Promoter
	2006-2012	School of Computing Scholarship Committee
Education	**2002**	Ph.D., Computer Science, University of Utah, School of Computing
	1991	Bachelors of Science, Computer Science, New Mexico State University
Honors	**2005-present**	Numerous Top 15% in Course Evaluations
	2007-2008	Outstanding Teaching Award, University of Utah, SoC
	2005-2006	Honorable Mention, Outstanding Teaching Award, SoC
Presentations	**2010**	Ghost Interruptions - A Digital Dance Animation Performance
	2008	Taught a week long workshop on ActionScript programming for the Utah State Office of Education.
Teaching	**2005-present**	**Sample Courses Taught** CS 1000 – Engineering Computing CS 1410 – Intro to Comp Sci (Entertainment Arts Emphasis) CS 1960 – Freshmen Symposium CS 2420 – Algorithms and Data Structures CS 3500 – Software Practice I CS 3505 – Software Practice II CS 4540 – Web Software Architecture CS 4000, 4500 – Senior Capstone Design/Project ME EN 1010 - Intro to Robotic System Design (Co-Taught)
	2003-2005	Instructor Exercise and Sports Fitness – Ultimate Frisbee

Research	**2000-2005**	Research Scientist, University of Utah Geometric Design and Computation Group
	1994-1999	Research Assistant, University of Utah Computer Vision Group
Community Service	**2010-present** **1992-2004**	Coach – Soccer, U6 →U9 Coach – Boys Competitive Soccer, U7→U18

Publications

Suraj R. Musuvathy, David E. Johnson, H.J. de St. Germain, Elaine Cohen, Chimiao Xu, Richard F. Riesenfeld, and Thomas C. Henderson, "Integrating Multiple Engineering Resources in a Virtual Environment for Reverse Engineering of Mechanical Parts", ASME IDETC 2005.

Rajesh Subramanian, H.J. de St.Germain, and Samuel Drake, "Integrating a Vision System with a Coordinate Measuring Machine to Automate the Datum Alignment Process", ASME IDETC/DAC 2005.

H. J. de St. Germain, D. E. Johnson, and Elaine Cohen, "Integrating Freeform and Feature-Based Fitting Methods", Design Engineering Technical Conference 2004

W.B. Thompson, J.C. Owen, H.J. de St. Germain, Stevan R. Stark, and T.C. Henderson, "Feature-Based Reverse Engineering of Mechanical Parts," *IEEE Transactions on Robotics and Automation*, 15(1), February 1999.

H.J. de St. Germain, S.R. Stark, W.B. Thompson and T.C. Henderson, "Constraint Optimization and Feature-Based Model Reconstruction for Reverse Engineering," *Proceedings of the DARPA Image Understanding Workshop*, May 1997.

W.B. Thompson, H.J. de St. Germain, T.C. Henderson, and J. C. Owen, "Constructing High-Precision Geometric Models from Sensed Position Data," *Proceedings of the ARPA Image Understanding Workshop*, February 1996

J.C. Owen, H.J. de St. Germain, S. Stark, T.C. Henderson and W.B. Thompson, "Calibrated Imagery for Quantitative Evaluation of IU Pose-Estimation and Stereo Algorithms," *Proceedings of the ARPA Image Understanding Workshop*, February 1996.

Calderwood et Al, "First Progress Report of Keypush Timing Group", NMSU-TR-91.CS-06, 1991

H. J. de St. Germain, and H. D. McCoy, "User Identification Through the Use of Classification Statistics and a Hamming-like Distance Measure", Internal Technical Report, 1991

Vita for GANESH GOPALAKRISHNAN
School of Computing, University of Utah, Salt Lake City, UT 84112-9205
(801) 581-3568 (Fax 5843) – ganesh@cs.utah.edu – http://www.cs.utah.edu/~ganesh

EDUCATION

- Ph.D. in Computer Science, State University of New York at Stony Brook, NY, Dec. 1986. Dissertation title: "From Algebraic Specifications to Correct VLSI Systems." Advisors: Profs. D. R. Smith and M. K. Srivas.
- M.Tech. in Electrical Engineering, Indian Institute of Technology, Kanpur, India, Aug. 1980.
- B.Sc. in Electrical Engineering, Regional Engineering College, Calicut, India, Jan. 1978.

EMPLOYMENT

- July 2009 to June 2010: Collaborative Sabbatical Research (with RiSE group), Concurrency Curriculum Development, Microsoft Research, Redmond, WA.
- July 2000 to present: Professor, School of Computing, University of Utah, Salt Lake City, UT.
- August 2002 to June 2003: Sabbatical visitor (Post-Silicon Verification), Intel Corporation, Santa Clara, CA.
- July 1993 to June 2000: Associate Professor, Computer Science, University of Utah, Salt Lake City, UT.
- September 1995 to May 1996: Sabbatical visitor (Dill group), Stanford University.
- August 1988 to July 1989: Visiting Asst. Prof. (Birtwistle group), Computer Science, University of Calgary
- September 1986 to June 1993: Asst. Prof., Computer Science, University of Utah
- August 1980 to July 1981: Research Associate (Rajaraman group), Department of Computer Science, Indian Institute of Technology, Kanpur, India.

LEADERSHIP ROLES

Director, Center for Parallel Computing at Utah (CPU, http://www.parallel.utah.edu).

AREAS OF CURRENT RESEARCH AND FUNDING

- *S12-SSE: Scalable Multifaceted GPU Program Debugging,* NSF ACI 1535032, $417,482, September 1, 2015-August 3, 2017.
- Utah PI (PI is Sriram Krishnamoorthy), *Whole-Program Adaptive Error Detection and Mitigation,* DOE (DE-FOA-0001059 Announcement on Resilience for Extreme Scale Supercomputing Systems), $465,108 (3 years; received one year's funding), July 15, 2015.
- Sole PI: CCF: SHF: Medium: Collaborative: *A Static and Dynamic Verification Framework for Parallel Programming.* (CCF 1302449, 4-15-13 to 4-14-17, $400,000). Collaborators: Vivek Sarkar and Eric Mercer.
- PI: SI2-SSE: *Correctness Verification Tools for Extreme Scale Hybrid Computing.* (ACI-1148127, 6-1-12 to 5-31-15, $444,279). Collaborators: Mary Hall and Rajeev Thakur.
- Faculty Associate: *Institute for Sustained Performance, Energy and Resilience.* (SUPER, 8-1-11 to 7-31-16, 0.5 months and one student per year). PI: Mary Hall.
- PI (co-PI is Rakamarić): CCF FRS (Failure Resistant Systems : *Localized, Layered Formal Hardware/Software Resilience Methods.* (CCF 1255776, 4-1-13 to 3-31-16, $115,500).
- PI (co-PI is Rakamarić): *Localized, Layered Formal Hardware/Software Resilience Methods.* SRC Task associated with above award. Program Officer: William Joyner (SRC Task 2426.001, $77,000).
- Sole PI: CSR: SMALL: *Design Validation Methods for Reliable and Efficient Floating-Point.* (CCF 1421726, 8/1/14 (est) to 7/31/16; $398,341)

- Co-PI (PI is Rakamarić): EAGER: *Memory Models: Specification and Verification in a Concurrency Intermediate Verification Language (CIVL).* (CCF 1346756, 8-31-13 to 8-30-15, $300,000).
- PI (with collaborator Martin Burtscher, Texas State Univ): *Nixing Scale Bugs in HPC Applications*, (CCF 1439002, XPS Program, 9/1/14 (est) to 8/31/16; $149,992)
- PI (Co-PI is Rakamarić): *Nondeterminism Control in Scientific Codes.* LLNL Contract, 1-22-14 to 1-21-15, $61,798).
- PI (Co-PI is Rakamarić): *Data Race Checkers*, LLNL Contract, 8-21-15 to 8-31-16, $61,798).

BOOKS

1. Ganesh Gopalakrishnan, "Computation Engineering: applied automata theory and logic" (505 pages). Springer, ISBN 0-387-24418-2, 2006.
2. Ganesh Gopalakrishnan and Shaz Qadeer, editors. Proceedings of the 23rd International Conference on Computer Aided Verification (CAV), Snowbird, UT, July 2011. LNCS 6806.
3. Slind, Konrad; Bunker, Annette; Gopalakrishnan, Ganesh C. (Eds.) Theorem Proving in Higher Order Logics, 17th International Conference, TPHOLS 2004, Park City, Utah, USA, September 14-17, 2004, Lecture Notes in Computer Science, Vol. 3223. ISBN: 3-540-23017-3
4. Ganesh Gopalakrishnan and Phillip Windley, editors. Formal Methods in Computer-Aided Design. Proceedings of the 2nd International Conference, FMCAD '98. Lecture Notes in Computer Science 1522, Springer-Verlag, 1998, 538 pages. ISSN 0302-9743.

REFEREED JOURNAL ARTICLES

2014 Alan Humphrey, Qingyu Meng, Martin Berzins, Diego Caminha B De Oliveira, Zvonimir Rakamaric, Ganesh C. Gopalakrishnan, "Systematic Debugging Methods for Large Scale HPC Computational Frameworks," Computing in Science & Engineering, no. 1, pp. 1, PrePrints PrePrints, doi:10.1109/MCSE.2014.11

REFEREED PAPERS IN WORKSHOPS AND CONFERENCES

1. Charles Jacobsen, Alexey Solovyev, Ganesh Gopalakrishnan, "A Parametrized Floating-Point Formalization in HOL Light," Electronic Notes in Theoretical Computer Science, Volume 317, 18 November 2015, Pages 101107, (selected from the papers presented at the Seventh and Eighth International Workshops on Numerical Software Verification, NSV). doi:10.1016/j.entcs.2015.10.010.
2. Wei-Fan Chiang, Ganesh Gopalakrishnan, and Zvonimir Rakamaric, "Unsafe Floating-point to Unsigned Integer Casting Check for GPU Programs," NSV 2015, Seattle, WA.
3. Alexey Solovyev, Charles Jacobsen, Zvonimir Rakamaric and Ganesh Gopalakrishnan, "Rigorous Estimation of Floating-Point Round-off Errors with Symbolic Taylor Expansions," pp. 532-550, Nikolaj Bjørner and Frank S. de Boer (ed.), FM 2015, LNCS 9109.
4. Jade Alglave, Mark Batty, Alastair F. Donaldson, Ganesh Gopalakrishnan, Jeroen Ketema, Daniel Poetzl, Tyler Sorensen and John Wickerson, "GPU concurrency: Weak behaviours and programming assumptions," ASPLOS 2015, Istanbul, Turkey.

SOFTWARE RELEASES

1. Vector oriented Utah LLVM Fault Injector (VULFI), `http://formalverification.cs.utah.edu/fmr/vulfi`.
2. **S3FP, a search based floating-point precision analysis framework.** AUTHOR: Wei-Fan Chiang (PhD student).
   Release at `http://www.cs.utah.edu/fv/s3fp`, 2014.

CONFERENCES/WORKSHOP CHAIRED/Co-CHAIRED

- Co-Organizers Sebastian Burkhardt, Azadeh Farzan, Ganesh Gopalakrishnan, Stephen Siegel, Helmut Veith, and Josef Widder, "Exploiting Concurrency Efficiently and Correctly – (EC)^2," CAV 2012 Workshop, July 7-13, 2012, Berkeley, CA `http://forsyte.at/ec2-2012/`
- Co-General-Chair (with Shaz Qadeer) **Computer Aided Verification (CAV), Snowbird, UT, July 2011.**
- Co-Organizers Sebastian Burkhardt, Swarat Chaudhuri, Azadeh Farzan, Ganesh Gopalakrishnan, Stephen Siegel, and Helmut Veith, "Exploiting Concurrency Efficiently and Correctly – (EC)^2," CAV 2011 Workshop, July 14-15, 2011, Snowbird, UT `http://www.cs.utah.edu/ec2`.

PROFESSIONAL ACTIVITIES

- SC15 BoF on "Reproducibility of High Performance Codes and Simulations – Tools, Techniques, Debugging," Birds of a Feather at Supercomputing 2015. `https://gcl.cis.udel.edu/sc15bof.php`.
- PC Member (ERC), Progr. Lang. Design and Implementation (PLDI), 2015.
- PC Member (ERC), Architectural Support for Programming Languages and Systems (ASPLOS), 2015.

INVITED / REFEREED TUTORIALS

1. Matthias S. Müller, David Lecomber, Tobias Hilbrich, Mark OConnor, Bronis R. de Supinski, and Ganesh Gopalakrishnan, "Efficient Parallel Debugging for MPI, Threads, and Beyond" **Full Day Tutorial, Supercomputing**, Austin, November 2015.

PhD COMMITTEES AS CHAIR

1. 2019 est, Vinu Joseph.
2. 2019 est, Arnab Das.
3. 2018 est, Saeed Taheri.
4. 2017 est, Simone Atzeni.
5. 2017 est, Mohammed Saeed Al-Mahfoudh
6. 2016 est, Vishal Sharma
7. 2016 est, Sriram Aananthakrishnan
8. 2016 est, Wei-Fan Chiang (co-advised with Zvonimir Rakamarić)
9. 2015 Peng Li (Fujitsu Research).
10. 2012, Subodh Sharma (Post-Doctoral Fellow at Oxford).
11. 2011, Anh Vo (Microsoft).
12. 2010, Sarvani Vakkalanka (Microsoft).
13. 2010, Guodong Li (Fujitsu Research).
14. 2009, Yu Yang (Software Engineer, SFO Area).
15. 2008, Xiaofang Chen (Pinterest)
16. 2007, Robert Palmer (Tableau Software)
17. 2006, Ritwik Bhattacharya (Microsoft)
18. 2004, Ali Sezgin (Postdoc with Sewell, Cambridge)
19. 2004, Yue Yang (co-advised with Lindström, Microsoft)
20. 2003, Annette Bunker (co-advised with Konrad Slind, Intel)
21. 2001, Mike Jones (Associate Professor, Brigham Young University)
22. 2000, Ravi Hosabettu (Juniper Networks)
23. 1999, Ratan Nalumasu (Google)
24. 1996, Prabhakar Kudva (Research Staff Member, IBM)
25. 1992, Venkatesh Akella (Professor, ECE, UC Davis)

INVITED TALKS / PANELS

1. "Bugs: Black Ice on Parallel Roads," delivered at three locations:
   (a) University of British Columbia.
   (b) Aalto University, Helsinki, Finland.

SUPERVISION OF POST DOCTORAL FELLOWS

1. Dr. Alexey Solovyev, recruited 9/2013.
2. Dr. Diego Oliveira, 7/2012 - 5/2014.
3. Dr. Igor Melatti, from 7/2005 till 12/2005. Also summer of 2006.
4. Dr. Abdel Mokkedem, from 7/1997 till 6/1999 (est). Working for Intel corpn.

UNIV. and COLLEGE COMMITTEES

- University Diversity Committee, 2011-2013.
- College Retention, Promotion, and Tenure Committee Representative from the School of Computing, 2001-02, 2004, 2008, 2010-2011 (as Chair), 2011-2012 (as member).
- 1990-92: College Council representative for CS.

DEPARTMENT COMMITTEES

2015

1. Organizer, Organick Memorial Lecture Committee, 2014, 2015.

Prior to 2015

2. Chairman, Retention, Promotion, and Tenure Committee, 2014.
3. Chairman, Retention, Promotion, and Tenure Committee, 2013.
4. Organizer, Organick Memorial Lecture Series, 2012, 2013.
5. 2009-12: Colloquium Committee Chair
6. 2006-09: Outreach, Colloquium
7. 2007-09: Graduate Admissions
8. 2010-11: Colloquium
9. 2005-06: Colloquium, Undergraduate Studies
10. 2004-05: Undergraduate Studies
11. 2003-04: Graduate Admissions
12. 2001-02: Member, Graduate Studies Committee
13. 2000-01: Member, Graduate Studies Committee
14. 1999-00: Assoc Chair for Research, Department of Computer Science.
15. 1999-00: Member of the Graduate Admissions and Comprehensive Exams Committees.
16. 1998-99: Director of Graduate Admissions, Comprehensive Exams.
17. 1997-98: Departmental RPT Chair, Comprehensive Exams.
18. 1996-97: Computer Engineering, Computer Policies.
19. 1994-95: Undergraduate, Computer Engineering.
20. 1993-94: Undergraduate, Computer Engineering.
21. 1992-93: Colloquium chair.
22. 1990-92: Graduate Committee.
23. 1989-90: Faculty Recruiting, Curriculum.
24. 1987-88: Grad. Admissions.
25. 1986-94: Library representative for the Department.

MARY WOLCOTT HALL

50 S. Central Campus Dr.; School of Computing, University of Utah; Salt Lake City, UT 84103
801-585-1039; mhall@cs.utah.edu; http://www.cs.utah.edu/∼mhall

EDUCATION

Ph.D., Rice University, April 1991 (Computer Science).
M.S., Rice University, March 1989 (Computer Science).
B.A., Rice University, May 1985 (Computer Science/Mathematical Sciences, Magna Cum Laude).

AWARDS

ACM Distinguished Scientist, since 2011.
CRA Leadership in Science Policy Institute, selected for class of 2015.

RECENT PROFESSIONAL ACTIVITIES

Leadership roles

Member, CRA Board of Directors, 2015-present.
Member, External Advisory Board, Institute for Advanced Computational Science, Stonybrook University, 2015-present.
Member, IEEE Computer Society Awards Committee, 2010-2011,2013-2014.
Chair, ACM and IEEE Computer Society Ken Kennedy Awards Committee, 2010, 2014.
Chair, ACM History Committee, 2009-2013.
Member, IEEE Computer Society Cray and Fernbach Awards Committees, 2009, 2010, 2011.
Member, ACM SIGPLAN Robin Milner Young Rseearcher Award Committee, 2013.

Conference Organization

Panel Chair, IEEE Cluster Conference 2016.
Test of Time Award Co-Chair, SC'15.
Technical Papers Co-Chair, SC'14.
Silver Anniversary Chair and Executive Committee member, SC'13.
Exhibits Chair and Executive Committee member, SC'12.
General Chair, ACM SIGPLAN PLDI 2011.
Program Chair, ACM SIGPLAN PPoPP 2010.
Program Committee service omitted.

Mentoring

Co-Organizer/Speaker, Programming Languages Mentoring Workshop (PLMW@PLDI), June 2016.
Invited Speaker, CRA Career Mentoring Workshop, Feb. 2016.
Invited Speaker, Career Workshop for Women and Minorities in Computer Architecture, held in conjunction with ACM MICRO, Dec. 2015.
Panelist, Students@SC Education Panel, SC'15, Nov. 2015.
Dinner with interesting people, Students@SC Program, SC'15, Nov. 2015.
Speed Mentor, Early Career Program, SC'15, Nov. 2015.
Mentor, Grace Hopper Conference Scholarship Lunch, Oct. 2015.
Invited Speaker, CS Lunch and Learn (undergraduates), Rice University, Oct. 2014.
Invited Speaker, Broader Engagement Program, SC12, Nov. 2012.

TEACHING

CS 4961/4230, Parallel Programming, Fall 2009,2010,2011,2012,2013.
CS 4400, Computer Systems, Fall 2015.
CS 6963/6235 Parallel Programming for Many-Core Architectures, Spring 2009,2010,2011,2012,2013,2014,2016.
Autotuning of Dense and Sparse Matrices on GPUs, International Summer School on Parallel High

Performance Computing using Accelerators, University of Minho, Braga, Portugal, June 2014.

Compiler-based autotuning technology, International Summer School on Advanced Computer Architecture and Compilation for High-Performance and Embedded Systems, Fiugi, Italy, July 2011.

ADVISING (since 2010)

Phd graduates:

Anand Venkat (2016), "An Integrated Compiler and Runtime Framework for Sparse Matrix Codes," Intel Research.

Saurav Muralidharan (2016), "Abstractions and Strategies for Adaptive Programming," Nvidia Research.

Protonu Basu (2015), "Compiler Optimizations and Autotuning for Stencil Computations and Geometric Multigrid," Lawrence Berkeley National Laboratory.

Malik Muhammad Murtaza Khan (USC, 2012), "Autotuning, Code Generation and Optimizing Compiler Technology for GPUs," Norwegian Institute of Science and Technology.

Masters graduates (with thesis):

Yu-Jung Lo (2015), Axel Rivera (2014), Shreyas Ramalingam (2012), Gabe Rudy (2010)

Masters graduates (with project):

Amit Roy (2016), Prajakta Mane (2015), Suchit Maindola (2012), Gagandeep Sachdev (2011)

GRANTS (since 2010)

1. "SHF: Medium: Collaborative Research: An Inspector/Executor Compilation Framework for Irregular Applications," PI, NSF CCF-1564074, $400K to Utah, 08/01/16-07/31/20.
2. "EAGER: Application-driven Data Precision Selection Methods," Co-PI, NSF CCF-1643056, $300K, 08/01/16-07/31/18.
3. "Using Active Harmony and CHiLL to Autotune Chapel," Utah PI, University of Maryland (Lead), NSA, $126K to Utah, 1/1/14-10/31/16.
4. "CSR: Medium: Energy-Efficient Architectures for Emerging Big-Data Workloads," Co-PI, $875K, NSF CNS-1302663, 7/1/13-6/30/17.
5. "Osprey: Efficient Embedded Parallel Computing Technologies," Utah PI, Nvidia Corporation (Lead), DARPA PERFECT program, 11/01/12-06/30/16.
6. "Autotuning for Exascale: Self-Tuning Software to Manage Heterogeneity," PI, $600K to Utah, DOE Exascale Software Stack Program, 09/01/12-08/31/16.
7. "SI2-SSE: Correctness Verification Tools for Extreme Scale Hybrid Concurrency," Co-PI, $444K, NSF SI2-1148127, 6/1/12-5/31/16.
8. "Institute for Sustained Performance, Energy, and Resilience (SUPER)," Utah PI, University of Southern California (Lead), DOE SciDAC Institute, ($1.075M to Utah), 09/01/11-08/31/16.
9. "Echelon: Extreme scale Compute Hierarchies with Efficient Locality Optimized Nodes," Utah PI, Nvidia Corporation (Lead), DARPA UCP Program, ($1.256M to Utah), 8/15/10-05/31/14.
10. "SHF Small: A Compiler-Based Auto-Tuning Framework for Many-Core Code Generation," PI, ($316K), 07/01/10-06/30/15.

RECENT REFEREED PUBLICATIONS

Conference Papers

1. "Automating Wavefront Parallelization for Sparse Matrix Codes," A. Venkat, M. Mohamadi, J. Park, R. Barik, H. Rong, M. Strout, M. Hall, *International Conference on Supercomputing, Networking, Storage and Analysis (SC)*, Nov. 2016, **Best Paper Finalist**.
2. "Synchronization Tradeoffs in GPU Implementations of Graph Algorithms," R. Kaleem, A. Venkat, S. Pai, M. Hall, K. Pingali, *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2016.

3. "Architecture-Adaptive Code Variant Tuning," S. Muralidharan, A. Roy, M. Hall, M. Garland, and P. Rai, *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, April 2016.
4. "Generating Efficient Tensor Contractions for GPUs," T. Nelson, A. Rivera, P. Balaprakash, M. Hall, P.D. Hovland, E. Jessup, B. Norris, *Proceedings of the IEEE International Conference on Parallel Processing (ICPP)*, Sept. 2015.
5. "Loop and Data Transformations for Sparse Matrix Code," A. Venkat, M. Hall, M. Strout, *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, June 2015.
6. "Compiler-Directed Transformations for Higher-Order Stencils," P. Basu, S. Williams, B. van Straalen, M. Hall, L. Oliker, P. Collela, *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2015.
7. "Nitro: A Framework for Adaptive Code Variant Tuning," S. Muralidharan , M. Shantharam, M. Hall, M. Garland, B. Catanzaro, *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS)*, May 2014.
8. "Non-affine Extensions to Polyhedral Code Generation," A. Venkat, M. Shantharam, M. Hall, M. M. Strout, *Proceedings of the International Conference on Code Generation and Optimization (CGO)*, Feb. 2014.
9. "Compiler Generation and Autotuning of Communication-Avoiding Operators for Geometric Multigrid," P. Basu, S. Williams, B. Van Straalen, A. Venkat, L. Oliker, M. Hall, *IEEE International Conference on High Performance Computing (HiPC)*, Dec. 2013.
10. "Analyzing the effect of compiler optimizations on application reliability," M. Demertzi, M. Annavaram and M. Hall, *Proceedings of the IEEE International Symposium on Workload Characterization (ISWC)*, Nov. 2011.
11. "EigenCFA: Accelerating Flow Analysis with GPUs," T. Prabhu, S. Ramalingam , M. Might, M. Hall, In *ACM SIGPLAN Principles of Programming Languages (POPL)*, Jan. 2011.
12. "Autotuning and Specialization: Speeding up Nek5000 with Compiler Technology," J. Shin, M. Hall, J. Chame, C. Chen, P. Fischer, P.D. Hovland, *International Conference on Supercomputing (ICS)*, June 2010.

Journal Articles

1. "Towards Making Autotuning Mainstream," P. Basu, M. Hall, M. Khan, S. Maindola, S. Muralidharan, S. Ramalingam, A. Rivera, M. Shantharam, A. Venkat, *International Journal of High Performance Computing Applications*, 27(4), November 2013.
2. "A script-based autotuning compiler system to generate high-performance CUDA code," M. Khan, P. Basu, G. Rudy, M. Hall, C. Chen, and J. Chame. *ACM Transactions on Architecture and Code Optimization*, 9(4), January 2013.
3. "Hierarchical parallelization and optimization of high-order stencil computations on multicore clusters," H. Dursun, M. Kunaseth, K. Nomura, J. Chame, R.F. Lucas, C. Chen, M. Hall, R.K. Kalia, A. Nakano, P. Vashishta, *The Journal of Supercomputing*, 62(2):946-966, December 2012.
4. "Understanding ACM's Past,", M. Hall, *Communications of the ACM*, 55(12), December 2012.
5. "Auto-tuning Full Applications: A Case Study", A. Tiwari, C. Chen, C. Liao, J. Chame, J. Hollingsworth, M. Hall and D. Quinlan, *International Journal of High Performance Computing Applications*, 25(3):286-294, Aug. 2011.
6. "Domain-Specific Optimization of Signal Recognition Targeting FPGAs," M. Demertzi, P.C. Diniz, M.W. Hall, A.C. Gilbert and Y. Wang, *ACM Transactions on Reconfigurable Technology and Systems*, 4(2), May, 2011.

7. “Parameterized specification, configuration and execution of data-intensive scientific workflows,” V.S. Kumar, T. Kurc, V. Ratnakar, J. Kim, G. Mehta, K. Vahi, Y.L. Nelson, P. Sadayappan, E. Deelman, Y. Gil, M. Hall and J. Saltz, *Cluster Computing*, April 2010.

SELECTED INVITED PRESENTATIONS AND KEYNOTES (since 2010)

1. “Compiler Optimization, Specialization and Autotuning: Achieving Productivity and High Performance for Diverse Architectures,” International Workshop on Automatic Performance Tuning, held at IPDPS’16, May 2016.
2. “Domain-Specific Optimization and Autotuning for Performance Portability of Supercomputing Applications,” EE Department Colloquium Series, Brigham Young University, Mar. 2016.
3. “The Role of Compiler Optimization and Autotuning for Reducing Data Movement in High-Performance Applications,” Keynote, Legacy HPC Application Migration Workshop, in conjunction with CANDAR 2015, Sapporo, Japan; Seminar, University of Tokyo, Dec. 2015.
5. “Autotuning Compiler and Library Technology for Sparse Matrix Computations,” Colloquium, Texas A&M University, Oct. 2015.
6. “Compiler Optimization of Computation and Communication in Stencils and Geometric Multigrid,” Intel Research, Santa Clara, Jul. 2015.
7. “The Role of Autotuning Compiler Technology,” Streamlining Application Performance Portability Minisymposium, SIAM CSE, Mar. 2015.
8. “Making Compilers Work: Autotuning for High Performance Applications,” Colloquia speaker, Rice University, Oct. 2014; University of Texas, Oct. 2014.
11. “Leveraging HPC Expertise and Technology in Data Analytics,” Workshop on Clusters, Clouds, and Data for Scientific Computing (CCDSC ’14), Leone, Sept. 2014.
12. “The Creative Process in Inventing Computer Science at the University of Utah,” 62nd Annual Utah State History Conference, Sept. 2014.
13. “Tiling Dense and Sparse Computations for Parallelism and the Memory Hierarchy of GPUs,” SIAM Parallel Processing Symposium, Feb. 2014.
14. “SC13 Silver Anniversary Panel: Retrospective in Supercomputing Technologies,” Panel Organizer and Moderator, SC13, Nov. 2013.
15. “Autotuning Compiler and Language Technology and its Role in Exascale Systems,” 6th International Conference on Automatic Differentiation, July 2012.
16. “Automating Application Mapping with Autotuning: Paving the Way to Exascale,” DOE Salishan Conference on High-Speed Computing, Apr. 2012.
17. “Compiler-Based Autotuning for Productive Parallel Programming and its Relationship to Design Space Exploration,” High-Level Synthesis and Parallel Computation Models Workshop held in conjunction with IEEE FCCM, May 2011.
18. “Compiler-Based Autotuning of Energy Applications,” USC-DOE Conference on Materials for Energy Applications: Experiment, Modeling and Simulations, Mar. 2011.
19. “Compiler-Based Autotuning for Productivity and High Performance,” Colloquia, Ohio State University, Feb. 2011.
20. “Compiler-Based Auto-tuning for Application and Library Code,” DOE SciDAC Workshop on Libraries and Autotuning for Petascale Applications, Aug. 2010.
21. “Paving the Way for Programming Extreme Scale Systems,” DOE Institute for Computing in Science, Future of the Field Workshop, Jul. 2010.
22. “Collaborative Autotuning of Scientific Applications,” SIAM Parallel Processing Symposium, Feb. 2010.

Vita

Charles D. Hansen

May 24, 2016

Current Address:	School of Computing 50 S. Central Campus Drive, 3190 MEB, University of Utah Salt Lake City, Utah 84112	(801) 581-3154 (work)
	hansen@cs.utah.edu	email
	www.cs.utah.edu/˜hansen	web page

Professional Employment

7/1 2005 to present	University of Utah	Professor (School of Computing)
9/1/2003 to 1/1/2016	University of Utah	Associate Director Scientific Computing and Imaging Institute
11/1 2011 to 04/30 2012	University Joseph Fourier	Visiting Professor
7/1/2008 to 6/30/2010	University of Utah	Associate Director School of Computing
8/15 2004 to 7/30 2005	INRIA Rhone-Alpes	Visiting Scientist
9/1 1998 to 6/30 2005	University of Utah	Associate Professor (School of Computing)
7/1 2001 to 6/03 2003	University of Utah	Associate Director School of Computing
2/1 1997 to 8/31 1998	University of Utah	Research Associate Professor (Dept. of Computer Science)
9/18 1989 to 1/31 1997	Los Alamos National Lab	Technical Staff Member Advanced Computing Laboratory
9/1 1994 to 1/31 1997	Univ. of New Mexico	Visiting Research Assistant Professor
9/1 1990 to 1/31 1997	New Mexico Tech	Adjunct Professor
8/1 1988 to 7/31 1989:	University of Utah	Visiting Assistant Professor
7/5 1987 to 7/31 1988:	INRIA	Postdoctoral Research Scientist (Vision and Robotics)
1/1 1987 to 7/5 1987:	University of Utah	Research Assistant (Vision Group)
9/15 1986 to 12/31 1986:	University of Utah	Teaching Assistant (Computer Science Dept)
9/1 1986 to 9/14 1986:	University of Utah	Research Assistant (Vision Group)
9/1 1983 to 8/31 1986:	ARO Research Fellow	
6/16 1983 to 8/31 1983:	University of Utah	Research Assistant (Very Large Text Retrieval Project)
9/15 1982 to 6/15 1983:	University of Utah	Teaching Assistant (Computer Science Dept)
8/1 1979 to 8/31 1982	Fred P. Gattas Co. Memphis, Tn.	Systems Programmer
6/1 1977 to 7/31 1979	D and S Systems, Memphis, Tn.	Systems Analyst

Education

AS	Computer Science Technology	State Technical Institute at Memphis	1977
BS	Applied Computer Science	Memphis State University	1981
PhD	Computer Science	University of Utah	June 1987
		Dissertation title “CAGD-based Computer Vision”	

Areas of Interests

Large Scale Scientific Visualization, Rendering Techniques and Computer Graphics, Parallel Algorithms, Distributed Computation, 3D Shape representation

Fellowships and Honors

IEEE Fellow: Elected to an IEEE Fellow 2012

IEEE Computer Society Technical Achievement Award "For Seminal Work on Tools for Understanding Large-Scale Scientific Data Sets", October 2005, The IEEE VGTC Visualization Technical Achievement Award was established in 2004. It is given every year to recognize an individual for a seminal technical achievement in visualization.

Best Paper Nomination "Visual Analysis of Uncertainties in Ocean Forecasts for Planning and Operation of Off-Shore Structures", IEEE Pacific Visualization 2013, March 2013.

Best Paper Award "Physically-Based Interactive Schlieren Flow Visualization", IEEE Pacific Visualization 2010, March 2010.

Best Paper Award "Non-Photorealistic Volume Rendering Using Stippling Techniques", IEEE Visualization 2002, Oct. 2002.

Best Paper Nomination "Interactive Translucent Volume Rendering and Procedural Modeling", IEEE Visualization 2002, Oct. 2002.

Best Paper Award "Interactive Volume Rendering Using Multi-Dimensional Transfer Functions and Direct Manipulation Widgets", IEEE Visualization 2001, Oct. 2001.

Best Paper Award "Interactive Ray Tracing for Isosurface Extraction", IEEE Visualization '98, Oct. 1998.

Best Panel Award "Tera-Scale Visualization: issues and approaches", IEEE Visualization '97, October 1997.

Bourse Chateaubriand Postdoctoral Research Fellowship, INRIA Rocquencourt France, July 1987 to July 1988

NSF Travel Award NATO Advanced Study Institute on Pattern Recognition Theory and Applications, Spa Belgium June 14 - June 22, 1986

ARO Fellowship Reliable Robotic Architectures, Army Research Office, September 1, 1983 to August 31, 1986

Phi Kappa Phi Member University of Utah

Professional Publications Student co-authors are underlined.

Published Papers: Books

1. "Scientific Visualizaiton: Uncertainty, Multifield, Biomedical, and Scalable Visualization", Charles Hansen, Min Chen, Christopher Johnson, Arie Kaufman, Hans Hagen editors, Springer-Verlag, 2014.

2. "High Performance Visualization: Enabling Extreme-Scale Scientific Insight", E. Wes Bethel, Hank Childs, Charles Hansen, editors, Chapman & Hall/CRC Computational Science, ISBN 9781439875728, 2012.

3. Charles Hansen and Chris Johnson, editors. Visualization Handbook, Elsevier Press, ISBN: 0-12-387582-x, 984 pages, 2004.

Published Papers: Journals

1. "State of the Art in Transfer Functions for Direct Volume Rendering", Patric Ljung, Jens Krge, Eduard Grler, Markus Hadwiger, Charles D. Hansen, Anders Ynnerman, Computer Graphics Forum Journal, Volume 35, Number 3, June 2016

   Pascal Grosset and Manasa Prasad and Cameron Christensen and Aaron Knoll and Charles Hansen, IEEE Transactions on Visualization and Computer Graphics, IEEE Early Access

2. "TOD-Tree: Task-Overlapped Direct send Tree Image Compositing for Hybrid MPI Parallelism and GPUs", Pascal Grosset and Manasa Prasad and Cameron Christensen and Aaron Knoll and Charles Hansen, IEEE Transactions on Visualization and Computer Graphics, IEEE Early Access

3. "A Survey of Colormaps in Visualization", Liang Zhou and Charles Hansen, IEEE Transactions on Visualization and Computer Graphics, IEEE Early Access

4. "A Shot at Visual Vulnerability Analysis", Ethan Kerzner, Charles Hansen, Miriah Meyer, Computer Graphics Forum Journal, Volume 34, Number 3, May 2015, pp. 391-400.

5. "Boundary Aware Reconstruction of Scalar Fields", Stefan Lindholm; Daniel Jnsson and Charles Hansen and Anders Ynnerman, IEEE Transactions on Visualization and Computer Graphics, Volume 20, Number 12, November 2014, pp. 2447 - 2455.

6. "GuideME: Slice-guided Semiautomatic Multivariate Exploration of Volumes", Liang Zhou and Charles Hansen, Computer Graphics Forum Journal, Volume 33, Number 3, June 2014, pp. 151-160.

7. "Ovis: A Framework for Visual Analysis of Ocean Forecast Ensembles", Thomas Hoellt, Ahmed Magdy, Peng Zhan, Guoning Chen, Ganesh Gopalakrishnan, Ibrahim Hoteit, Charles D. Hansen, and Markus Hadwiger, IEEE Transactions on Visualization and Computer Graphics, Volume 20, Number 9, 2014, pp. 1114-1126.

8. "Synthetic Brainbows", Yong Wan, Hideo Otsuna, and Charles Hansen, Computer Graphics Forum Journal, Volume 32, Number 3, June 2013, pp. 471-480.

9. "Similarity Measures for Enhancing Interactive Streamline Seeding", Tony McLoughlin, Mark Jones, Robert Laramee, Charles D. Hansen, IEEE Transactions on Visualization and Computer Graphics, Volume 19, Number 8, 2013, August 2013, pp.1342-1353.

10. "Ambient Occlusion Effects for Combined Volumes and Tubular Geometry", Mathias Schott, Tobias Martin, A.V. Pascal Grosset, Sean T. Smith, Charles D. Hansen, IEEE Transactions on Visualization and Computer Graphics, Volume 19, Number 6, 2013, June 2013, pp. 913-926.

11. "Dye-Based Flow Visualization", Grzegorz K. Karch, Filip Sadlo, Daniel Weiskopf, Charles D. Hansen, Guo-Shi Li, and Thomas Ertl, IEEE Computing in Science & Engineering, Volume 14, Number 6, November 2012, pp. 80-86.

12. "Transfer Function Combinations", Liang Zhou, Charles D. Hansen, Computer and Graphics, Volume 36, Number 6, October 2012, pp. 596-606.

13. "Design of 2D Time-Varying Vector Fields" Guoning Chen, Konstantin Mischaikow, Vivek Kwatra, Li-Yi Wei, Charles D. Hansen, Eugene Zhang, IEEE Transactions on Visualization and Computer Graphics, Volume 18, Number 10, October 2012, pp. 1717-1730.

14. "A Practical Workflow for Making Anatomical Atlases in Biological Research", IEEE Computer Graphics and Applications, Yong Wan, A. Kelsey Lewis, Mary Colasanto, Mark van Langeveld, Gabrielle Kardon, Charles Hansen, Volume 32, Number 5, September 2012, pp. 70-80.

15. "Direct Feature Visualization Using Morse-Smale Complexes". Attila Gyulassy, Natallia Kotava, Mark Kim, Charles Hansen, Hans Hagen, Valerio Pascucci, IEEE Transactions on Visualization and Computer Graphics, Volume 18, Number 9, September, 2012, pp. 1549-1562.

16. "Generalized Swept Mid-structure for Polygonal Models", Tobias Martin, Guoning Chen, Suraj Musuvathy, Elaine Cohen, Charles Hansen, Computer Graphics Forum Journal (proceedings of EuroGraphics), Volume 31, Number 2, May 2012, pp. 805-814.

Editorial Duties:
Guest Editor: IEEE Transactions on Visualization and Computer Graphics, Vol. 22, No. 6, 2016
Guest Editor: IEEE Transactions on Visualization and Computer Graphics, Vol. 14, No. 6, November/December 2008
Guest Editor: IEEE Transactions on Visualization and Computer Graphics, Vol. 13, No. 6, November/December 2007
Guest Editor: IEEE Computer Graphics and Applications, Vol. 23, No. 2, March 2003
Guest Editor: IEEE Computer Graphics and Applications, Vol. 14, No. 4, July 1994
Guest Editor: IEEE Parallel and Distributed Technology, Vol. 2, No. 2, Summer 1994
Editorial Board: IEEE Transactions on Visualization and Computer Graphics, September 2012 - present
Editorial Board: IEEE Computing NOW, September 2012 - present
Editorial Board: Computers and Graphics, September 2011 - present
Editorial Board: IEEE Transactions on Visualization and Computer Graphics, January 2003 - January 2007
Editorial Board: International Journal of High Performance Computer Graphics, Multimedia and Visualization, February 1997 - 1999
Associate Editor-in-Chief: IEEE Transactions on Visualization and Computer Graphics, August 2003 - January 2007
Associate Editor-in-Chief: IEEE Transactions on Visualization and Computer Graphics, June 2014 - Present

University Departmental and College Committees

Department: Director: Graphics and Visualization Track, 2012-present
Faculty Search Committee - Computer Vision 2015-2016
Faculty Search Committee - Data Science 2014-2015
Faculty Search Committee - Visualization 2014-2015
Faculty Search Committee - Computer Vision 2013-2014
Faculty Search Committee - Visualization 2013-2014
Faculty Search Committee - Visualization 2012-2013
Research Grants Review 2011
Associate Director, School of Computing, 2009-2010
Research Grants Review 2010
Research Grants Review 2009
Associate Director, School of Computing, 2008-2009
Graduate Admissions Committee 2008
Research Grants Review 2008

University: UPTAC Committee 2014-2017
Graduate Council, Graduate School, 2014-present
Graduate Admissions Committee, Graduate School, 2014-present
Associate Director of the Scientific Computing and Imaging Institute 2015-2016
Associate Director of the Scientific Computing and Imaging Institute 2014-2015
Associate Director of the Scientific Computing and Imaging Institute 2013-2014
Associate Director of the Scientific Computing and Imaging Institute 2012-2013
Associate Director of the Scientific Computing and Imaging Institute 2010-2011
IT Council 2010-2011
Associate Director of the Scientific Computing and Imaging Institute 2009-2010
IT Council 2009-2010
Associate Director of the Scientific Computing and Imaging Institute 2008-2009
IT Council 2008-2009
Leadership Workshop 2008-2009

Curriculum Vitae

THOMAS C. HENDERSON

Professional Employment

7/89 to present. University of Utah. Professor of Computer Science
8/2011 to 12/2011. University of Karlsruhe, Visiting Professor (sabbatical)
8/2010 to 7/2011. National Science Foundation. Program Director
7/2003 to 12/2003. University of Karlsruhe, Visiting Professor (sabbatical)
7/2000 to 6/2003. University of Utah. Founding Director, School of Computing
12/97 to 7/99. University of Utah. Associate Dean of Engineering
9/91 to 7/1997. University of Utah. Chairman of Computer Science
7/89 to 7/91. University of Utah. Associate Chairman of Computer Science
1/85 to present. University of Utah. Adjunct Professor of Bioengineering
7/84 to 7/89. University of Utah. Associate Prof. of Computer Science
9/88 to 4/89. INRIA, Sophia-Antipolis, Visiting Professor (sabbatical)
7/85 to 7/87. University of Utah. Associate Chairman of Computer Science
1/82 to 6/84. University of Utah. Assistant Prof. of Computer Science
10/80 to 12/81. INRIA, Rocquencourt, France. Visiting Professor
11/79 to 9/80. DFVLR, Oberpfaffenhofen, W. Germany. Research Associate

Education

BS, Math (with Honors), Louisiana State University 1973
PhD, Computer Science, The University of Texas at Austin 1979

Professional Publications

Books

Computational Sensor Networks, Springer-Verlag, May 2009.

Analysis of Engineering Drawings and Raster Map Images, Springer-Verlag, 2014.

Published Papers: Journals

1. "RobotShare: A Google for Robots," T. Henderson and Y. Fan, special issue on "Cognitive Humanoid Robots" of the International Journal of Humanoid Robotics, Vol 5, Bo 2, June 2008, pp. 311-329.

2. “Feature Fusion for Basic Behavior Unit Segmentation from Video Sequences,” Xinwei Xue and Thomas C. Henderson, Robotics and Autonomous Systems, Vol. 57, No. 3, March 2009, pp. 239-248.

3. “Perimeter Detection in Wireless Sensor Networks,” Kyle Luthy, Edward Grant, Nikhil Deshpande, and Thomas C. Henderson, Journal of Robotics and Autonomous Systems, Vol. 60, No. 2, Feb, 2012, pp. 266-277.

4. “Target Localization and Autonomous Navigation using Wireless Sensor Networks - A Pseudo-Gradient Algorithm Approach”, Nikhil Deshpande, Edward Grant, and Thomas C. Henderson, IEEE Systems Journal Special Issue on Sensor Networks for Advanced Localization Systems, Vol. 8, No. 1, pp. 93–103, March 2014.

5. “Model Accuracy Assessment in Reaction-Diffusion Pattern Formation in Wireless Sensor Networks,” Thomas C. Henderson, Anshul Joshi, Karl Rashkeev, and Narong Boonsirisumpun, Kyle Luthy and Edward Grant, International Journal of Unconventional Computing, Vol. 10, No. 4, pp. 317-338, 2014.

Oral Presentations: Professional Meetings

1. “Issues Related to Parameter Estimation in Model Accuracy Assessment,” Thomas C. Henderson and Narong Boonsirisumpun, International Conference on Computational Science, ICCS 2013, Barcelona, Spain, 5-7 June, 2013.

2. “Symmetry Bundles as Affordances,” Thomas C. Henderson, Anshul Joshi and Wenyi Wang, Robot Science and Systems, Workshop: “From Experience to Concepts and Back,” Berlin, Germany, June 27, 2013.

3. “Robot Cognition using Bayesian Symmetry Networks,” Thomas C. Henderson, Anshul Joshi and Wenyi Wang, Internaitonal Conference on Agents and Artificial Intelligence, Angers, France, March 6-8, 2014.

4. “Generative Cognitive Representation for Embodied Agents,” Anshul Joshi and Thomas C. Henderson, IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Beijing, Sept. 28-30, 2014.

5. “SLAMBOT: Structural Health Monitoring using Lamb Waves,” Wenyi Wang, Thomas C. Henderson, Anshul Joshi and Edward Grant, IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Beijing, Sept. 28-30, 2014.

6. “Symmetry Based Semantic Analysis of Engineering Drawings,” Thomas C. Henderson, Narong Boonsirisumpun, and Anshul Joshi, IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Beijing, Sept. 28-30, 2014.

7. "Bayesian Computational Sensor Networks: Small-scale Structural Health Monitoring," Wenyi Wang, Anshul Joshi, Nishith Tirpankar, Philip Erickson, Michael Cline, Palani Thagaraj, and Thomas C. Henderson, International Conference on Computational Science, Reykavik, Iceland, June 1-3, 2015.

8. "Informing Change: Course Content Analysis and Organization," Linda DuHadway and Thomas C. Henderson, Proceedings IEEE Conference on Frontiers in Education, El Paso, TX, Oct 21-24, 2015

9. "Actuation in Perception: Character Classification in Engineering Drawings," Thomas C. Henderson, Narong Boonsirisumpun and Anshul Joshi, Proceedings IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, San Diego, CA, Sept 14-16, 2015.

10. "Gaussian Processes for Multi-Sensor Environment Modeling," Philip Erickson, Michael Cline, Nishith Tirpankar and Thomas C. Henderson, Proceedings IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, San Diego, CA, Sept 14-16, 2015.

11. "Artificial Student Agents and Course Mastery Tracking," Linda DuHadway and Thomas C. Henderson, Proceedings International Conference on Agents and Artificial Intelligence, Rome, Italy, Feb 24-26, 2016.

12. "Actuation-based Shape Representation applied to Engineering Document Analysis," Thomas C. Henderson, Narong Boonsirisumpun and Anshul Joshi, Proceedings International Conference on Agents and Artificial Intelligence, Rome, Italy, Feb 24-26, 2016.

13. "Laying a Foundation for the Graphical Course Map," Linda DuHadway and Thomas C. Henderson, Proceedings of the ACM Web for All Conference, Montreal, CA, 11-13 April 2016.

International Courses and Committees

- European Union Cognitive Robotics Proposal Evaluation Panel, Brussels, Belgium, 2-5 June 2008, 2009, 2012, 2013.
- NSF Committee of Visitors, Reviewer of Intelligent and Information and Systems Division, 19-21 May, 2009.
- NSF Science and Technology Center Site Reviewer, October 2012.
- Co-Program Chair, IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 2013.
- Co-organizer, Workshop at Robot Science and Systems, Berlin, Germany, June 2013.

- Program Chair, IEEE Multisensor Fusion and Integration, San Diego, CA, September 2015.

Research Grants and Contracts

1. AFOSR - STTR (2007-2009) “Exploiting Raster Maps for Imagery Analysis” $223,000 (of $750,000 total to IAVO, Inc.)
2. STC Corp (2008) “Qualitative Modeling for Fault Diagnosis,” $47,610.
3. NSF (2010-2012) “Innate Theories in Robotics,” $36,000.
4. NSF (2010-2011) “IPA,” $273,000.
5. AFOSR (2012-2015) “Bayesian Computational sensor Networks,” $614,796.

Honors and Awards

- 2000 Utah Governor’s Medal for Science and Technology
- 2003 IEEE Fellow
- Best Paper Award, IEEE Conference on Multi-sensor Fusion and Integration, Seoul, Korea, August, 2008.

Doctoral Committees (As Supervisor)

- Chuck Hansen (Computer Science, June 1987) ”CAGD-Based Computer Vision: The Automatic Generation of Recognition Strategies”
- Rod Grupen (Computer Science, Aug. 1988) ”General Purpose Grasping and Manipulation with Multifingered Robot Hands”
- Ashok Samal (Computer Science, Aug. 1988) ”Parallel Split-Level Relaxation”
- Jun Gu (Computer Science, 1989) ”Parallel Algorithms and Architectures for Very Fast AI Search”
- Mohamed Dekhil (Computer Science, 1998) “Instrumented Sensor Systems”
- Xinwei Xue (Computer Science, 2008) “Behavior Analysis”
- Linda DuHadway (Computer Science, May 2016) ”Course Transformation: Content, Structure and Effectiveness Analysis”
- Anshul Joshi (Computer Science, est. 2016) “Symmetry in Robotics”
- Narong Boonsirisumpun (Computing, Robotics Track, est. 2017) ”Semantic Analysis of Engineering Drawings using Symmetry”

Tucker Ryer Hermans

School of Computing
50 S Central Campus Drive Room 3190
Salt Lake City, UT, 84112

thermans@cs.utah.edu
www.cs.utah.edu/ ˜ thermans
+1 (801) 581-8122

Education

Georgia Institute of Technology, School of Interactive Computing *Atlanta, GA*
Ph.D. Robotics, May 2014
› Thesis: Representing and Learning Affordance-Based Behaviors
› Thesis Committee: Aaron Bobick (**advisor**), James M. Rehg (**co-advisor**), Henrik Christensen, Charles C. Kemp, Mike Stilman, and Dieter Fox (Univ. Washington),

Georgia Institute of Technology, College of Computing *Atlanta, GA*
M.S. Computer Science: Computational Perception and Robotics, Aug 2012

Bowdoin College *Brunswick, ME*
A.B. Magna Cum Laude in Computer Science (Hon.) and German, May 2009

Humboldt Universität zu Berlin *Berlin, Germany*
Coursework in Computer Science and German Literature, 2007–2008

Experience

School of Computing, University of Utah *July 2015–Present*
Assistant Professor

Department of Mechanical Engineering, University of Utah *January 2016–Present*
Adjunct Assistant Professor

Technische Universität Darmstadt, Department of Computer Science *April 2014–July 2015*
Postdoctoral Researcher

Georgia Institute of Technology, School of Interactive Computing *Aug 2009–April 2014*
Graduate Research Assistant

Georgia Institute of Technology, School of Interactive Computing *Fall 2011*
Graduate Teaching Assistant: CS 4495 Computer Vision

Awards and Honors

ICDL-Epirob CIS Student Travel Grant	*2013*
Georgia Tech President's Fellowship	*2009–2013*
Phi Beta Kappa, Alpha of Maine	*2009*
Maine State Police Colonel's Award	*2009*
Sarah and James Bowdoin Scholar (Dean's List)	*2006, 2007*

Lectures and Invited Talks

Invited Talk:"Visual and Tactile Learning for Robot Manipulation" *January 2016*
Department of Mechanical Engineering, Brigham Young University

Invited Talk:"Visual and Tactile Learning for Robot Manipulation" *April 2015*
School of Computing, University of Utah

Invited Talk:"Visual and Tactile Learning for Robot Manipulation" *March 2015*
School of Computer Science, McGill University

Invited Talk:"Visual and Tactile Learning for Robot Manipulation" *February 2015*
Department of Computer Science, Drexel University

Guest lecture: "Model Learning" *November 2014*
Robot Learning (Lernende Roboter), Department of Computer Science, TU Darmstadt

Invited talk: "Tactile Sensing for Object Manipulation in Clutter" *September 2014*
Third Workshop on Robotics in Clutter, IROS 2014

Academic Service

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) *2016*
Associate Editor

School of Computing, University of Utah *2016*
Colloquium Chair

Faculty Hiring Committee: Computer Vision, School of Computing, University of Utah *2015-2016*
Committee Member

Graduate Admissions Committee, School of Computing, University of Utah *2015-2016*
Committee Member

Workshop on "Visual and Tactile Learning for Interaction" at Robotics: Science and Systems 2015 *July 2015*
Lead Organizer

Faculty Hiring Committee, School of Interactive Computing, Georgia Tech *2012-2013*
Student Committee Member

Program Committee Member:
ECCV Workshop on Affordances (2014), RSS Workshop on Affordances (2014), IROS Workshop on Cognitive Robotics and Systems (2013), RSS Workshop on Robots in Clutter (2013), ICRA Workshop on Interactive Perception (2013), RoboCup Symposium (2010, 2011)

Reviewer:
International Journal of Robotics Research, IEEE Transactions on Robotics, Autonomous Robots, IEEE Transactions on Cognitive and Developmental Systems, IROS, ICRA, Humanoids, Journal of Intelligent and Robotic Systems, NIPS 2014 Workshop: Autonomously Learning Robots, HRI: Workshops and Tutorials

Publications

Journal Articles

› F. Veiga, H. van Hoof, J. Peters, and T. Hermans. "Stabilizing Novel Objects through Tactile Prediction of Slip." ***Under Review:*** *IEEE Transactions on Robotics*, 2015.

Conference and Workshop Papers

› Z. Yi, R. Calandra, H. van Hoof, F. Veiga, T. Hermans, Y. Zhang, and J. Peters. "Active Tactile Object Exploration with Gaussian Processes," ***Under Review:****IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

› J. Hoelscher, J. Peters, and T. Hermans. "Evaluation of Tactile Feature Extraction for Interactive Object Recognition." *IEEE-RAS International Conference on Humanoid Robotics (Humanoids)*, 2015.

› H. van Hoof, T. Hermans, G. Neumann, and J. Peters. "Learning Robot In-Hand Manipulation with Tactile Features." *IEEE-RAS International Conference on Humanoid Robotics (Humanoids)*, 2015.

› F. Veiga, H. van Hoof, J. Peters, and T. Hermans. "Detecting Slip and Stabilizing Grip of Novel Objects with Tactile Sensing." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.

› T. Hermans, F. Li, J. M. Rehg, A. F. Bobick. "Learning Contact Locations for Pushing and Orienting Unknown Objects." *IEEE-RAS International Conference on Humanoid Robotics (Humanoids)*, Atlanta, GA, USA, October 2013.

› A. Ciptadi, T. Hermans, J. M. Rehg, "An In Depth View of Saliency." *British Machine Vision Conference (BMVC)*, Bristol, United Kingdom, September 2013.

› T. Hermans, F. Li, J. M. Rehg, A. F. Bobick. "Learning Stable Pushing Locations." *IEEE International Conference on Developmental Learning and Epigenetic Robotics (ICDL-Epirob)*, Osaka, Japan, August 2013.

› T. Hermans, J. M. Rehg, A. F. Bobick. "Decoupling Behavior, Perception, and Control for Autonomous Learning of Affordances." *IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

› T. Hermans, J. M. Rehg, A. F. Bobick."Decoupling Behavior, Control, and Perception in Affordance-Based Manipulation." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS): Workshop on Cognitive Assistive Systems*, Vilamoura, Portugal, October 2012.

› T. Hermans, J. M. Rehg, A. F. Bobick."Guided Pushing for Object Singulation." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Portugal, October 2012.

› A. Cosgun, T. Hermans, V. Emeli, M. Stilman, "Push Planning for Object Placement on Cluttered Table Surfaces." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.

› T. Hermans, J. M. Rehg, A. F. Bobick, "Affordance Prediction via Learned Object Attributes," *ICRA Workshop on Semantic Perception, Mapping, and Exploration*, 2011.

› H. Zhou, T. Hermans, A. V. Karandikar, J. M. Rehg, "Movie Genre Classification via Scene Categorization." *ACM Multimedia*, Florence, Italy, November 2010.

› H. Work, E. Chown, T. Hermans, J. Butterfield, M. McGranaghan, "Player Positioning in the Four-Legged League." *RoboCup: Robot Soccer World Cup XII*. Suzhou, China, 2008.

› H. Work, E. Chown, T. Hermans, J. Butterfield, "Robust Team-Play in Highly Uncertain Environments." *International Joint Conference on Autonomous Agents and Multiagent Systems*. Estoril, Portugal, May 2008.

Peer Reviewed Demonstrations

› T. Hermans, F. Veiga, H. van Hoof, J. Hoelscher, J. Peters, "Demonstration: Learning for Tactile Manipulation." *Neural Information Processing Systems (NIPS)*, Montreal, Canada, December 2014.

Funding

Current

› None

Pending

› NSF: NRI: Intuitive Teloperation of Underactuated Tactile Sensing Robot Hands (PI) $649,910

Teaching

› CS 6370 / ME EN 6225 “Motion Planning” - Fall 2015
› CS 7939 / ME EN 7960 “Seminar in Robotics” - Fall 2015
› CS 6300 “Artificial Intelligence” - Spring 2016
› CS 7930 “School of Computing Colloquium” - Spring 2016

John M. Hollerbach

University of Utah, School of Computing
http://www.cs.utah.edu/~jmh

PROFESSIONAL PREPARATION

University of Michigan	Chemistry	B.S.	1968
University of Michigan	Mathematics	M.S.	1969
Massachusetts Institute of Technology	EECS	S.M.	1975
Massachusetts Institute of Technology	EECS	Ph.D.	1978

APPOINTMENTS

University of Utah	School of Computing	Professor	1994-Present
	Mechanical Eng.	Adjunct Professor	1994-Present
McGill University	Biomedical Eng. and Mechanical Eng.	NSERC/CIAR Professor of Robotics	1989 - 1994
M.I.T.	Brain and Cognitive Sciences	Associate Professor	1985 - 1989
M.I.T.	Psychology	Assistant Professor	1982 - 1985
M.I.T.	Dept. Psychology and A.I. Lab	Research Scientist	1978-1982
M.I.T.	EECS and A.I. Lab	Research Assistant	1972-1978

HONORS AND AWARDS

Life Fellow, IEEE, 2016.
Volunteer of the Year Award, Utah Regional FIRST Robotics Competition, 2013.
Distinguished Service Award, IEEE Robotics and Automation Society, 2012.
Outstanding Teacher Award, School of Computing, University of Utah, 2006.
Student Choice Teaching Award, Associated Students of the University of Utah, 2006.

SELECTED SERVICE

Editor-in-Chief, International Journal of Robotics Research, 2000 - Present. Top-rated robotics journal by impact factor.

Director, Robotics Track, University of Utah School of Computing. Since 2005, the Robotics Track has been a joint program of study for M.S. and Ph.D. students in the School of Computing and Department of Mechanical Engineering, and the second such degree program after CMU.

Vice President for Technical Activities 2010-2011, and Vice President for Conference Activities 2008-2009, IEEE Robotics and Automation Society.

Organizing Committee, Utah Regional FIRST Robotics Competition, 2010-Present. As part of our outreach activities for an NSF IGERT on Biocentric Robotics, the robotics group started the

Utah Regional FIRST Robotics Competition (FRC). Until then, there were only 2 or 3 Utah high school teams participating in FRC, and they had to travel out of state. Now there are more than two dozen, and it has been made mandatory for the 5 high schools of the Canyons School District, and the regional has hosted up to 53 teams in total.

co-Founder, University of Utah Robotics Center (URC), January 2016. The URC formalizes the buildup of robotics at Utah over the years, which includes common labs and equipment, the Robotics Track, and the Utah Regional FRC.

Member, Interim Computing Community Consortium (CCC) Council, 2006-2007. This interim committee created the permanent CCC Council of the CRA. Spearheaded the formation of the CCC Iniative on Robotics by asking Henrik Christensen and select members of the robotics community to submit a proposal to the CCC. Participated in a roadmapping activity, which eventually resulted in the creation of the National Robotics Initiative.

COURSES TAUGHT

CS 5310/6310 & ME EN 5220/6220 Introduction to Robotics, 1994-present.
CS 5300/6300 Artificial Intelligence, 2011, 2014-15.
CS 7320 & ME EN 7220 System Identification for Robotics, 2006-present.
CS 7939 & ME EN 7960-002 Robotics Seminar, 2005-present.

RECENT Ph.D. SUPERVISION

Current: 4 Ph.D. students. Graduated 2 Ph.D. students in recent past.

RECENT RESEARCH

Fingertip Force Prediction by Imaging the Fingernail. Normal and shear forces of the fingerpad contact with a flat surface can be estimated with external camera imaging of the fingernail and surrounding skin. This permits contact force to be estimated without instrumented objects, and can be used for scientific studies of grasping and for human-computer interfaces. This is a joint project with Prof. Steve Mascaro of Mechanical Engineering, and is currently supported by an NSF NRI grant.

Treadport Active Wind Tunnel. The Treadport is a large tilting treadmill inside a CAVE-like virtual environment with stereo display and floor projection. A user wears a harness attached to a horizontal mechanical tether, which measures user position to control the speed of the belt and progress and direction in the virtual environment. The tether also exerts a frontal horizontal force, which can simulate collision forces, missing inertial forces, and slope walking. A wind tunnel has been created around the Treadport to generate wind on the cross section of the user with speeds up to 20mph and from a broad frontal direction. This is the world's first steerable wind tunnel in any context. A natural odor display is created by injecting smells into the wind, and an infrared heat source is used for radiant heat display to simulate an outdoor virutal environment in combination.

This project has been funded by two NSF grants. This has been a joint project with Profs. Mark Minor and Eric Pardyjak of Mechanical Engineering.

Gait Rehabilitation with the Treadport. One view of the Treadport is as the most realistic simulation of real-world walking in a safe environment. This capability is being utilized for patients with incomplete spinal cord injuries (iSCI) and Parkinson's patients (currently supported by two separate NSF grants). iSCI patients can regain some ability to walk, but there is a big gap between their clinical training and real-world walking in terms of remaining gait abnormalities and robustness to falling. The Treadport can present realistic challenges and tasks in a safe setting that better prepares patients for the real world. For Parkinson's patients, a challenge is walking on uneven terrain such as on cobblestones. This project is creating an actuated shoe with multiple controllable air bladders to present uneven terrain. The Treadport's stereoscopic floor display can create virtual objects to step over. The iSCI research is in collaboration with Jake Abbott, and the Parkinson's research with Mark Minor and Andrew Merriweather of Mechanical Engineering and Bo Foreman of Physical Therapy.

SELECTED PUBLICATIONS

Grieve, T., Hollerbach, J.M., and Mascaro, S.A., "Optimizing Fingernail Imaging Calibration for 3-D Force Magnitude Prediction," *IEEE Trans. Haptics,* in press.

Grieve, T., Hollerbach, J.M., and Mascaro, S.A., "3-D Fingertip Touch Force Prediction Using Fingernail Imaging with Automated Calibration," *IEEE Trans. Robotics, 31*, 2015, pp. 2116-2129.

Hejrati, B., Crandall, K., Hollerbach, J.M., and Abbott, J.J., "Kinesthetic Force Feedback and Belt Control for the Treadport Locomotion Interface," *IEEE Transactions on Haptics, 8,* 2015, pp. 176-187.

Hejrati, B., Hull, D., Black, J., Abbott, J.J., and Hollerbach, J.M., "Investigation of the Treadport for gait rehabilitation of spinal cord injury," Engineering in Medicine and Biology Conference, Los Angeles, August, 2012, pp. 4553-4558.

Kulkarni, S., Fisher, C.J., Lefler, P., Desai, A., Chakravarthy, S., Pardyjak, E., Minor, M.A., and Hollerbach, J.M., "A Full-Body Steerable Wind Display for a Locomotion Interface," *IEEE Trans. Visualization and Computer Graphics, 21*, 2015, pp. 1146-1159.

Kulkarni, S., Chakravarthy, S., Minor, M.A., Pardyjak, E.R., and Hollerbach, J.M., "Control of a duct flow network for wind display in a virtual environment," *IEEE/ASME Trans. Mechatronics, 17,* 2012, pp. 1021-1030.

Kulkarni, S., Minor, M.A., Deaver, M.W., Pardyjak, E.R., and Hollerbach, J.M., "Design, sensing, and control of a scaled wind tunnel for atmospheric display," *IEEE/ASME Trans. Mechatronics, 17,* 2012, pp. 635-645.

Kulkarni, S., Deaver, M.W., Pardyjak, E.R., Minor, M.A., and Hollerbach, J.M., "Design elements of a novel atmospheric flow simulator," *ASME J. Fluids Engineering, 133,* 2011, pp. 121402-1-10.

Sun, Y., Hollerbach, J.M., and Mascaro, S.A., "Estimation of finger force direction with computer vision," *IEEE Trans. Robotics, 25,* 2009, pp. 1356-1369.

Sun, Y., Hollerbach, J.M., and Mascaro, S.A., “Predicting fingertip forces by imaging coloration changes in the fingernail and surrounding skin,” *IEEE Trans. Biomedical Engineering, 55,* 2008, pp. 2363-2371.

Curriculum Vitae

Peter A. Jensen
Lecturing Associate Professor
School of Computing at the University of Utah

October 2016

Career Focus

My career focus is to use innovative teaching methods to teach college students computer science skills and principles. As a lecturing faculty member in the School of Computing, I primarily work to advance undergraduate education through excellence in teaching and exploration of new methods and technologies for teaching computer science. I also participate in academic governance, student advising, undergraduate administration, and outreach to the broader community. I have interests in computer engineering and computer science related to architecture, number theory, and entertainment and arts engineering.

Education

University of Utah
Doctor of Philosophy: Computer science
Completed February 2008

University of Utah
Bachelor of Science: Computer science
Completed May 1995
Magna cum laude, with honors

Professional background

September 1995 - Present
School of Computing, University of Utah – Salt Lake City, Utah
Lecturing Associate Professor / Lecturing Assistant Professor / Instructor / Teaching Assistant / Research Assistant / Outreach Coordinator

My research experience has included brief forays into operating systems, robotics, scientific computing, but my primary work has been in computer science education. I have implemented dynamic web page tools and databases in support of on-line classes, and I have taught a variety of courses (*see below*).

January 2008 – Present
Clark Planetarium – Salt Lake City, Utah
Consultant / Programmer

I develop and maintain (on an occasional basis) software in support of educational displays at the planetarium. I have created 3D multimedia applications, I have helped develop and implement educational user interfaces, and I have written interface code for use with a Kinect, touchscreens, trackballs, and an industrial scale.

March 1996 – December 2005
NBO Systems, Inc. – Salt Lake City, Utah
Consultant / Programmer

I developed software and hardware for ticketing and sales kiosks and I programmed a robust, high-throughput multithreaded application for credit card transaction processing.

March 1998 – August 1998
Punkinhead L.L.C. – Murray, Utah
Developer / Programmer

I developed a small application called 'Me on a Pumpkin', which used image processing to allow a user to input a picture and convert it into a stencil appropriate for pumpkin carving.

July 1993 – March 1995
Sculptured Software – Salt Lake City, Utah
Programmer

I worked with a development team to create games for the Super Nintendo console system. My responsibilities included programming data compression routines, post-process editing of artwork, and the design, artwork, and programming for animations.

May 1986 – July 1993
Mastery Development – Redmond, Washington
Senior programmer / Consultant

My responsibilities included the programming of educational software and games for elementary school students, the design and implementation of hardware, drivers, and software for a 48-computer network with high-bandwidth multicast support, and the development of utilities to support the software and hardware production assembly lines.

Teaching History

For more than 18 years I have maintained a pattern of teaching excellence, and I continuously work to improve my pedagogy, my use of teaching tools, and my professionalism.

University of Utah Courses

(Course numbering/naming has changed. Courses are listed using current titles and numbers.)

Course Name	Course Number	Semesters Taught
Engineering Computing	CS 1000	Spring 2012, Spring 2011
Introduction to Java (non-major)	CS 1021	Fall 2009, Fall 2004, Summer 2002, Spring 2002, Fall 2001, Fall 1999, Fall 1998, Summer 1998
Creating Interactive Web Content	CS 1040	Spring 2007, Fall 2006, Spring 2006, Summer 2002
Introduction to Computer Science	CS 1400	Summer 2013
Object-Oriented Programming	CS 1410	Fall 2014, Fall 2013, Spring 2013, Fall 2012, Spring 2012, Fall 2011, Spring 2011, Spring 2010, Fall 2009, Spring 2009, Fall 2008, Spring 2008, Fall 2007, Spring 2007, Fall 2006, Spring 2006
Data Structures and Algorithms	CS 2420	Spring 2014, Summer 2010, Spring 2009, Spring 2008, Spring 2005, Summer 1999

Software Practice II	CS 3505	Spring 2015, Spring 2014, Spring 2013, Spring 2010
Computer Design Lab	CS 3710	Fall 2016
Computer Organization	CS 3810	Fall 2016, Spring 2015, Fall 2014, Fall 2013, Fall 2012, Fall 2011, Fall 2010, Fall 2009, Fall 2008, Fall 2007
Algorithms	CS 4150	Spring 2006
Programming Challenges	CS 4190	Fall 2014, Fall 2012, Fall 2011, Fall 2010, Spring 2009, Spring 2008
Mobile Apps: iPhone (Instructor Mentor)	CS 4962	Spring 2012, Spring 2011, Spring 2010
Teaching Introductory CS	CS 5040	Fall 2013, Fall 2006, Spring 2001

Awards

Award	Years received
School of Computing Outstanding Teaching Award	2012, 2010, 2007
Dean's letter of teaching excellence	2013 (x2), 2012, 2011, 2010 (x2), 2009 (x2), 2008 (x2), 2007, 2002, 1999

Service Rolls

June 2016 – Present *Committee Chair*
University of Utah – Senate Advisory Committee on IT

January 2014 – Present *Senator*
University of Utah – Academic Senate

December 2013 – Present *Representative*
College of Engineering – College Council

August 2005 – Present *Advisor / Mentor*
School of Computing – Advising

August 2008 – Present *CE Committee Member / ABET*
Computer Engineering – Service

October 1998 – Present *Team coach*
ACM – Rocky Mountain Regional Programming Contest

January 2015 – May 2016 *Committee Member*
University of Utah – Faculty IT Ad hoc Committee of the Academic Senate

August 2008 – August 2014 *Participant / Member*
School of Computing – Curriculum Committee

September 2013, September 2014 *Meeting Leader*
Utah State Higher Education – Major's Meeting

September 2013 – May 2014 *Committee Member*
University of Utah – Committee for Technology Enhanced Curriculum

January 2011 – December 2013 *Outreach Coordinator*
School of Computing

October 2008 – June 2012 *Event director*
College of Engineering – 'Utah Engineering' (An NSF Grant)

Additional Service - Outreach Events

2014, 2012, 2010, 2009, 2007, 2006 *Utah Site Director*
Rocky Mountain Regional ACM Contest

2012, 2011, 2010, 2009 Organizer / *Director*
Meet an Inventor Night

June 2008 *Session Leader*
Utah Bio Innovation Summit

March 2007 *Organizer / Director*
Utah High School Programming and Design Competition

1996 – 2004 *Chief Judge / Problem writer*
Utah High School Programming Competition

Publications

Peter Jensen. Hybrid Fault Localization in Programs Written By Novice Programmers. Ph.D. dissertation, School of Computing, University of Utah, 2008.

Peter Jensen, edited by Lee Siegel. *Press release – School of Computing dominates again.* In *The Daily Utah Chronicle*, *The Salt Lake Tribune*, and picked up nationally by *ZDNet*, November 2006.

Joseph Zachary and Peter Jensen. Exploiting value-added content in an online course: introducing programming concepts via HTML and JavaScript. In *Proceedings of the 34th SIGCSE technical symposium on computer science education*, pages 396-400, February 2003.

Elizabeth Odekirk, Dominic Jones, and Peter Jensen. Three semesters of CSO using Java: assignments and experiences. In *Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on innovation and technology in computer science education*, pages 144-147, July 2000.

Christopher R. Johnson - Summary Curriculum Vita 2016

Professional Experience – Current Positions:

Director, Scientific Computing and Imaging Institute, University of Utah (2000-)
Distinguished Professor, School of Computing, University of Utah (2003-)
Research Professor of Bioengineering, University of Utah (2002-)
Adjunct Professor of Physics, University of Utah (2002-)
Co-Director, NIH Center for Integrative Biomedical Computing (2000-)

Professional Experience – Selected Past Positions:

Co-Director, DOE Visualization and Analytics Center for Enabling Technologies (2006-11)
Director, School of Computing, University of Utah (2003-05)
Director, Engineering Scholars Program, University of Utah (1999 – 2004)
Director, University ACCESS Program, University of Utah (1993-96)
Co-Director, Computational Engineering and Science Program (1992-98)

Honors and Awards – Selected:

Elected Fellow of the IEEE (2014), SIAM (2009), AAAS (2005), AIMBE (2004)
IEEE Sidney Fernbach Award (2013)
College of Science Outstanding Alumni Award, Wright State University (2013)
IEEE IDPDS Charles Babbage Award (2012)
IEEE Visualization Career Award (2010)
Rosenblatt Award – University of Utah (2010)
Utah Cyber Pioneer Award (2009)
Governor's Medal for Science and Technology (1999)

Scholarship

Books:

1. M. Chen, H. Hagen, C. Hansen, C.R. Johnson, and A. Kauffman, editors. Scientific Visualization: Uncertainty, Multifield, Biomedical, and Scalable Visualization. Springer-Verlag, 416 pages, 2014.
2. C. Hansen and C.R. Johnson, editors. Visualization Handbook. Elsevier/Academic Press, 962 pages, 2004.

Peer Reviewed Articles – Recent from more than 150

1. B. Hollister, G. Duffley, C. Butson, and C.R. Johnson. Visualization for Understanding Uncertainty in Activation Volumes for Deep Brain Stimulation, in *Eurographics Conference on Visualization (EuroVis) 2016*, Editors: K.L. Ma G. Santucci, and J. van Wijk, 2016 (in press).

2. P. Rosen, B. Burton, K. Potter, C.R. Johnson. muView: A Visual Analysis System for

Exploring Uncertainty in Myocardial Ischemia Simulations, In *Visualization in Medicine and Life Sciences III*, Edited by L. Linsen, B. Hamann, and H.C. Hege, Springer, pp. 45-65. 2016.

3. X. Tong, J. Edwards, C. Chen, H. Shen, C. R. Johnson, P. Wong. View-Dependent Streamline Deformation and Exploration, In *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 7, pp. 1788-1801, 2016

4. C. R. Johnson. Visualization of Scalar and Vector Fields. In *Encyclopedia of Applied and Computational Mathematics*, Editor: Bjorn Engquist, pp. 1537-1546, Springer, ISBN: 978-3-540-70528-4 (Print) 978-3-540-70529-1 (Online), 2015.

5. C.R. Johnson. Computational Methods and Software for Bioelectric Field Problems. In *Biomedical Engineering Handbook*, J.D. Bronzino and D.R. Peterson, editors, 4th Edition, vol I, CRC Press, Boca Raton, Chapter 43, 2015.

Keynote, Plenary, and Distinguished Lectures – Recent from more than 90

1. *Image-Based Modeling, Simulation, and Visualization*, IBM Research, Almaden, September 2016, (IBM Distinguished Lecture).
2. *Big Data Visual Analysis*, Platform for Advanced Scientific Computing (PASC16), Lausanne, June 2016 (Plenary Presentation).
3. *Big Data Visual Analysis*, Illinois Institute of Technology, October 2015 (Sigma Xi Lecture).
4. *Visualizing the Future of Medicine*, Fermi Lab, Batavia, September 2015 (Fermi Lecture).
5. *Image-Based Modeling, Simulation, and Visualization*, International Symposium on Big Data and Predictive Computational Modeling, Munich, May 2015 (Keynote Presentation).

Additional Invited Lectures – Recent from more than 170

1. *Image-Based Modeling, Simulation, and Visualization*, Beihang University, Beijing, December 2015.
2. *Big Data Visual Analysis*, Big Data Science Park, Guiyang, December 2015.
3. *Big Data Visual Analysis*, Haite High-Tech, Chengdu, December 2015.
4. *Visualizing the Future of Medicine*, University of Chicago, Chicago, October 2015.
5. *Exploratory Visualization for Pattern Discovery*, Future in Review Conference 2015, Park City, October 2015. 31
6. *The Future of Uncertainty Visualization*, Workshop on Visualization for Decision Making Under Uncertainty, Chicago, October 2015.
7. *Big Data Visual Analysis*, University of Queensland, Brisbane, September 2015.

8. *Big Data Visual Analysis*, Monash University, Melbourne, September 2015.
9. *Big Data Visual Analysis*, ICERM Workshop on Mathematics in Data Science, Providence, July 2015.
10. *Large-Scale Visual Analysis*, National Research Council Board on Atmospheric Sciences and Climate, Boulder, March 2015.

Research Grants and Contracts - Current Support

NIH/NIGMS Center for Integrative Biomedical Computing, 15 years (three 5 year competitive renewals, $17,289,281. Chris R. Johnson (Co-PI), Rob MacLeod (Co-PI), Ross Whitaker (Co-PI), April 30, 2020.

NIH/NIGMS: Predictive Modeling of Bioelectric Activity on Mammalian Multilayered Neuronal Structures in the Presence of Supraphysiologic. 5 years, $3,146,740. Gianluca Lazzi (PI), Chris R. Johnson (Co-PI), Miriah Meyer (Co-PI), August 30, 2017.

DOE: Scalable Data Management Analysis and Visualization (SDAV) Institute. 5 years, $2,250,000. Valerio Pascucci (PI), Chris R. Johnson (Co-PI), Chuck Hansen (Co-PI), February 28, 2017.

NVIDIA Center of Excellence Renewal. 2 years. $324,000. Chris R. Johnson (PI), Chuck Hansen (Co-PI), December 31, 2017.

Research Grants and Contracts - Past Support: 61 awards for more than $72M in funding as PI and Co-PI.

Teaching

Advising: 11 postdocs, 18 Ph.D. students, 12 M.S. students, 74 Ph.D. committees (8 different Departments), 39 M.S. committees, 8 undergraduate projects

New Courses Developed: Advanced Methods in Scientific Computing I & II, Scientific Visualization, Computational Inverse Problems

New Courses Co-Developed: Engineering Computing, Integrated Sciences (ACCESS), Algorithms and Architectures for Scientific Computing

Service to the Profession

Journal and Book Editorial Boards: 7 current, 6 past, 4 special issues

Advisory Boards and National Committees – Current selected from 40

CRA Board of Directors; MATHEON International Advisory Board; IEEE Visualization and Graphics Technical Committee – Executive Committee; HPC Technical Computing Advisory Panel

Other: Chair or Co-Chair – Conferences and Workshops (15); Award Committees, Program Committees, Paper Committees (>100); NSF, NIH, DOE, and CRA National Committee Reports (15)

Service to the University – Selected

Vice President for Research Search Committee (twice); Distinguished Professor Advisory Committee; Advisory Board, UU’s Professional Master of Science and Technology Program; Dean of Dentistry Search Committee; Neuroscience Initiative Executive Committee; Chair of Radiology Search Committee; Entrepreneurial Faculty Advisory – Executive Committee; Department of ECE Chair Search Committee; Center for Quantitative Biology Advisory Board; University Information Technology Committee; Brain Institute, Scientific Advisory Board; Medical Informatics Chair Search Committee; HPC Strategic Planning Committee; University Rhodes Scholarship Committee; University Bachelors in Undergraduate Studies Committee

Presentations (selected): Gould Distinguished Lecture; Frontiers of Science Lecture; Utah Senate Leadership; Utah Legislative Higher-Education Appropriations Committee; Park City Institute; SLC Rotary Club; University Advisory Board; College of Engineering External Advisory Board; Alumni Association, California (multiple); Alumni Association, Seattle (multiple); Centennial Celebration for the College of Engineering; Research Presentations in the Departments of Bioengineering, Computer Science, Mathematics, Physics, Biomedical Informatics, Geology and Geophysics, Radiology, Cardiology, Internal Medicine, School of Medicine Library, and the Cardiovascular Research and Training Institute, Center for High Performance Computing, Energy Geosciences Institute, Huntsman Cancer Institute, Utah Center for Advanced Imaging, Center for Interdisciplinary Art and Technology

DAVID E. JOHNSON

University of Utah, School of Computing
http://www.cs.utah.edu/~dejohnso
dejohnso@cs.utah.edu

EDUCATION

Carleton College	Computer Science/Physics	B.A.	1990
University of Utah	Computer Science	Ph.D.	2005

Dissertation Title: Minimum Distance Queries for Haptic Rendering

PROFESSIONAL APPOINTMENTS

University of Utah School of Computing	Assistant Professor (Lec)	8/2015 - present
University of Utah School of Computing	Associate Instructor	8/2014 – 7/2015
University of Utah School of Computing	Research Associate	6/2005 – 7/2014
University of Utah School of Computing	Research Staff	6/1999 – 5/2005
University of Utah Computer Science	Research Assistant	9/1995 – 5/1999
University of Utah Computer Science	Utah Graduate Fellowship	9/1994 – 8/1995
University of Utah Computer Science	NSF Graduate Fellowship	9/1991 – 8/1994
University of Utah Computer Science	Teaching Assistant	9/1990 – 8/1991

TEACHING EXPERIENCE

Software Practice II, CS3505, University of Utah
Fall 2015, Spring 2016
Introduction to Computer Science, CS/EAE1400/1030, University of Utah
Fall 2014, Spring 2015, Fall 2015, Spring 2016
**Programming for Engineers,* ME6250, University of Utah
Fall 2012-2014
Explorations in Computer Science, CS1060, University of Utah
Fall 2009, Spring 2010-2014
**Geometric Computation for Motion Planning*, CS6370, University of Utah,
Fall 2005-2014, Spring 2015
**Virtual Reality*, CS6360, University of Utah,
Spring 2007, 2008, 2009, 2011, 2014

* Courses that were developed by Dr. Johnson

CURRENT FUNDING

NSF, “EXP: Deepening Conceptual Understanding with Hands-on, Augmented-Reality Experimentation”. PI. 2012-August 2016. $524,051.

NSF, “CHS: Small: Toward a New Generation of Untethered Magnetic Haptic Interfaces”. Senior Personnel. 2014-July 2017. $500,000.

PROFESSIONAL ACTIVITIES

Invited Talks

Course lecturer, "Recent Advances in Haptic Rendering and Applications," SIGGRAPH 2005.
Invited speaker, "Building a Shape Display," Intel Research, 2002.
Invited speaker, "Haptic Rendering of NURBS Models," IMA Haptics 2001.

Conference Organization

Session chair for Graphics Interface 2009.
Demonstrations Chair for World Haptics 2008.
Session co-chair for ASME IDETC/CIE 2005.

External Paper Reviewer

ACM SIGGRAPH
ASME Symposium on Haptic Interfaces
IEEE World Haptics
IEEE VR
IEEE Int. Conf. on Robotics and Automation
3D Symposium on Interactive Computer Graphics
International Journal of Robotics Research
Journal of Computer-Aided Design
IEEE Visualization
Eurographics
ASME International Design Engineering Technical Conference

SERVICE ACTIVITIES

2008-current	Organizing committee and Head Robot Design Judge for FIRST FLL robot competition clubs
Summer 2008-current	Director and instructor, GREAT summer camp. Founded and grew a summer outreach program serving over 600 students in weekly programs.
Summer 1998-2001	Instructor, High School Computing Institute summer camp

STUDENT ADVISING

Seiedmuhammad Yazdian, Ph.D. Mechanical Engineering, committee member
Mohammadreza Mollaei, Ph.D. Mechanical Engineering, committee member
Nathan Nelson, Ph.D. Mechanical Engineering, committee member
Brian Gleeson, Ph.D. Mechanical Engineering, committee member, graduated 2012
Ramya Bandaru, M.S. of Mechanical Engineering, committee member, graduated 2009
Daniel Perry, M.S. Computing, committee member, graduated 2008
Brandt Erickson, Ph.D. Computing, committee member,
Matt Frey, M.S. of Mechanical Engineering, committee member, graduated 2007

JOURNAL PUBLICATIONS

AJ Doxon, DE Johnson, HZ Tan, WR Provancher, "Human Detection and Discrimination of Tactile Repeatability, Mechanical Backlash, and Temporal Delay in a Combined Tactile-Kinesthetic Haptic Display System", IEEE Transactions on Haptics. 6(4), 453-463, 2013.

J Park, AJ Doxon, WR Provancher, DE Johnson, HZ Tan, "Haptic edge sharpness perception with a contact location display", IEEE Transactions on Haptics. 5(4), 323-331, 2012.

AJ Doxon, DE Johnson, HZ Tan, WR Provancher, " Force and contact location shading methods for use within two-and three-dimensional polygonal environments", Presence: Teleoperators and Virtual Environments 20 (6), 505-528, 2011.

Joon-Kyung Seong, David E. Johnson, Gershon Elber, and Elaine Cohen, "Critical Point Analysis Using Domain Lifting for Fast Geometry Queries," *Journal of Computer-Aided Design*, 2010.

Johnson, D., Willemsen, P., and Cohen, E., “6-DOF Haptic Rendering Using Spatialized Normal Cone Search,” Special Issue – Haptics, Virtual and Augmented Reality, *IEEE Transactions on Visualization and Computer Graphics*, 2005.

K. Museth, D. Breen, R. Whitaker, S. Mauch and D. Johnson, "Algorithms for Interactive Editing of Level Set Models," *The International Journal of Eurographics Assoc.: Computer Graphics Forum* Vol. 24, No. 4, pp. 821-841, December 2005.

REFEREED CONFERENCE PUBLICATIONS

D Johnson, “ITCH: Individual Testing Of Computer Homework For Scratch Assignments”, Special Interest Group on Computer Science Education (SIGCSE 2016).

S Yazdian, AJ Doxon, DE Johnson, HZ Tan, WR Provancher. “Compliance display using a tilting-plate tactile feedback device”, Haptics Symposium (HAPTICS), 2014 IEEE, 1-1, 2014.

S Yazdian, AJ Doxon, DE Johnson, HZ Tan, WR Provancher, “Compliance display using a tilting-plate tactile feedback device”, Haptics Symposium (HAPTICS), 2014 IEEE, 13-18 2014.

JB Brink, AJ Petruska, DE Johnson, JJ Abbott, “Factors affecting the design of untethered magnetic haptic interfaces”, Haptics Symposium (HAPTICS), 2014 IEEE, 107-114 2014 (Best Paper Award).

T Taylor, DE Johnson, “Tangible simulations Generalized haptic devices for human-guided computer simulations”, 2013 International Conference Collaboration Technologies and Systems (CTS), 2013.

J Park, WR Provencher, DE Johnson, HZ Tan, “Haptic contour following and feature detection with a contact location display”, World Haptics Conference (WHC), 2013, 7-12, 2013.

S Yazdian, AJ Doxon, DE Johnson, HZ Tan, WR Provancher, “2-DOF contact location display for manipulating virtual objects”, World Haptics Conference (WHC), 2013, 443-448, 2013.

J Park, AJ Doxon, WR Provancher, DE Johnson, HZ Tan. “Edge sharpness perception with force and contact location information”, World Haptics Conference (WHC), 2011 IEEE, 517-522, 2011.

LJ Flemming, DE Johnson, SA Mascaro, “Optimal control of multi-input SMA actuator arrays using graph theory”, Robotics and Automation (ICRA), 2011 IEEE International Conference on, 6109-6114, 2011.

P Willemsen, DE Johnson, J Clark, A Larson, E Pardyjak, “ Adaptive Optimization of Urban Layout for Air Quality using a GPU Cluster“, Ninth Symposium on the Urban Environment. 2010

A Mahoney, J Bross, D Johnson, “Deformable robot motion planning in a reduced-dimension configuration space”, Robotics and Automation (ICRA), 2010 IEEE International Conference on, 5133-5138, 2010.

Andrew J. Doxon, David E. Johnson, Hong Z. Tan, and William R. Provancher, "Force and Contact Location Shading Thresholds for Smoothly Rendering Polygonal Models", 2010 Haptics Symposium, March 25-26, 2010.

Brian Gleeson and David E. Johnson, "Expressive Haptic Rendering with Cartoon-Inspired Effects", 2010 Haptics Symposium, March 25-26, 2010.

Johnson, D., Riesenfeld, R., Cohen, E., and Drake, S., "Interactive Functional Reparameterization of Geometric Models," in ASME IDETC/DAC 2009, August 2009.

David E. Johnson and Elaine Cohen, "Computing Surface Offsets and Bisectors Using a Sampled Constraint Solver," *Graphics Interface 2009*, May 2009.

BOOK CHAPTERS

David E. Johnson and Elaine Cohen, "Haptic Rendering of Sculptured Models," in *Haptic Rendering: Foundations, Algorithms, and Applications* (Ming Lin and Miguel Otaduy, eds.), AK Peters, 2008.

GRADUATE ADVISOR

Dr. Elaine Cohen, School of Computing, University of Utah.

Sneha Kumar Kasera

School of Computing	kasera@cs.utah.edu
University of Utah	http://www.cs.utah.edu/~kasera
50 S Central Campus Drive, Rm 3190	Phone: (801) 581-4541
Salt Lake City, UT 84112	Fax: (801) 581-5843

INTERESTS

Computer networks and systems – technologies, protocols and applications encompassing network security and privacy and reliability, mobile and pervasive systems and wireless networks, Internet of things, crowdsourcing, dynamic spectrum access, network resource management, measurements, and modeling, and social network applications.

EDUCATION

University of Massachusetts Amherst, MA
Doctor of Philosophy in Computer Science 1999

Indian Institute of Science Bangalore, India
Master of Engineering in Electrical Communications Engineering 1990

EMPLOYMENT

University of Utah Salt Lake City, UT
Professor, School of Computing 2015-present
Associate Professor, School of Computing 2009-2015
Assistant Professor, School of Computing 2003-2009
Adjunct Associate Professor, Electrical and Computer Engineering 2009-present

Bell Labs Research, Lucent Technologies Holmdel, NJ
Member of Technical Staff, Mobile Networking Research Department 1999-2003

PUBLICATIONS[*] (2009 – present)

Journal Articles

1. S. N. Premnath, J. Croft, N. Patwari, and S. K. Kasera, "*Efficient High Rate Secret Key Extraction in Wireless Sensor Networks Using Collaboration*," in ACM Transactions on Sensor Networks, November 2014.
2. N. Patwari, J. Wilson, S. Ananthanarayana, S. K. Kasera, D. Westenskow, "*Monitoring Breathing via Signal Strength in Wireless Networks*," in IEEE Transactions on Mobile Computing, vol. 13, no. 8, pages 1774-1786, August 2014.
3. S. N. Premnath, D. Wasden, S. K. Kasera, N. Patwari, B. Farhang-Boroujeny, "*Beyond OFDM: Best Effort Dynamic Spectrum Access Using Filterbank Multicarrier*," in IEEE Transactions on Networking, vol. 21, no. 3, June 2013.
4. S. N. Premnath, S. Jana, J. Croft, P. Lakshmane Gowda, M. Clark, S. K. Kasera, N. Patwari, S. V. Krishnamurthy, "*Secret Key Extraction from Wireless Signal Strength in Real Environments*," in IEEE Transactions on Mobile Computing, vol. 12, no. 5, May 2013.
5. D. Maas, M. H. Firooz, J. Zhang, N. Patwari, S. K. Kasera, "*Channel Sounding for the Masses: Low Complexity GNU 802.11b Channel Impulse Response Estimation*," in IEEE Transactions on Wireless Communications, vol. 11, no. 1, pages 1-8, January 2012.
6. N. Patwari and S. K. Kasera, "*Temporal Link Signature Measurements for Location Distinction*," in IEEE Transactions on Mobile Computing, vol. 10, no. 3, pages 449-462, March 2011.
7. S. Jana and S. K. Kasera, "*On Fast and Accurate Detection of Unauthorized Access Points Using Clock Skews*," in IEEE Transactions on Mobile Computing, vol. 9, no. 3, pages 449-462, March 2010.
8. N. Patwari, J. Croft, S. Jana, and S. K. Kasera, "High-Rate Uncorrelated Bit Extraction for Shared Secret Key Generation from Channel Measurements," in IEEE Transactions on Mobile Computing, vol. 9, no. 1, pages 17-30, January 2010.

[*] Underlined names are those of my students.

Papers in Reviewed Conferences

1. M. Khaledi, M. Khaledi, and Sneha Kumar Kasera, "*Profitable Task Allocation in Mobile Cloud Computing*," in Proc. of the 12th ACM International Symposium on QoS and Security for Wireless Networks (Q2SWINET), November 2016.
2. M. Khaledi, M. Khaledi, Sneha Kumar Kasera, and N. Patwari, "*Preserving Location Privacy in Radio Networks Using a Stackelberg Game Framework*," in Proc. of the 12th ACM International Symposium on QoS and Security for Wireless Networks (Q2SWINET), November 2016.
3. P. Lundrigan, M. Khaledi, M. Kano, N. Subramanyam, and Sneha Kumar Kasera, "*Mobile Live Video Upstreaming*," in Proc. of the 28th International Teletraffic Congress, September 2016.
4. J. Kunz, C. Becker, M. Jamshidy, Sneha Kumar Kasera, R. Ricci, and K. Van der Merwe, "*OpenEdge: A Dynamic and Secure Open Edge Service Network*," in Proc. of IEEE/IFIP Network Operations and Management Symposium (NOMS), April 2016.
5. R. Quinn, J. Kunz, A. Syed, Sneha Kumar Kasera, R. Ricci, and K. Van der Merwe, "*KnowNet: Towards a Knowledge Plane for Enterprise Network Management*," in Proc. of IEEE/IFIP Network Operations and Management Symposium (NOMS), April 2016.
6. A. Banerjee, R. Mahindra, K. Sundresan, Sneha K. Kasera, K. Van der Merwe, and S. Rangarajan, "*Scaling the LTE Control-Plane for Future Mobile Access*," in Proc. of ACM CoNEXT, December 2015.
7. A. Banerjee, B. Nguyen, V. Gopalakrishnan, Sneha K. Kasera, S. Lee, and K. Van der Merwe, "*Efficient, Adaptive, Scalable Device Activation for M2M Communications,*" in Proc. of IEEE International Conference on Sensing, Communications, and Networking (SECON), June 2015.
8. M. Probst, J.C. Park, and S. Kasera, "*Exploiting Altruism in Social Networks for Friend-to-Friend Malware Detection*," in Proc. of 2nd IEEE Conference on Communications and Network Security, October 2014.
9. A. Banerjee, D. Maas, M. Bocca, N. Patwari, and Sneha K. Kasera, "*Violating Privacy Through Walls by Passive Monitoring of Radio Windows*," in Proc. of 7th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec), July 2014.
10. S. N. Premnath, P. Lakshmane Gowda, Sneha K. Kasera, N. Patwari, and R. Ricci, "*Secret Key Extraction Using Bluetooth Wireless Signal Strength Measurements*," in Proceedings of IEEE International Conference on Sensing, Communications, and Networking (SECON), June 2014.
11. M. Khaledi, S. Kasera, N. Patwari, M. Bocca, "*Energy Efficient Radio Tomographic Imaging*," in Proc. of IEEE International Conference on Sensing, Communications, and Networking (SECON), June 2014.
12. S. N. Premnath, S. K. Kasera, N. Patwari, and B. Farhang-Boroujeny, "*Beyond OFDM: Best-Effort Dynamic Spectrum Access Using Filterbank Multicarrier*," in Proc. of COMSNETS, January 2012.
13. D. Maas, N. Patwari, S. K. Kasera, D. Wasden, and M. Jensen, "*Experimental Performance Evaluation of Location Distinction for MIMO Links*," in Proc. of COMSNETS, January 2012.
14. S. N. Premnath, Sneha K. Kasera, N. Patwari, and B. Farhang-Boroujeny, "*Efficient Dynamic Spectrum Access in Vehicular Networks using Filterbank Multicarrier*," in Proc. of the First International Conference on Wireless Technologies for Humanitarian Relief, December 2011.
15. M. Seth, S. K. Kasera, and R. Ricci, "*Emergency Service in WiFi Networks Without Access Point Association*," in Proc. of the First International Conference on Wireless Technologies for Humanitarian Relief, December 2011.
16. J. Zhang, Sneha K. Kasera, N. Patwari, and P. Rai, "*Distinguishing Locations Across Perimeters Using Wireless Link Measurements*," in Proc. of IEEE Infocom, April 2011.
17. M. Probst, J.C. Park, R. Abraham, Sneha K. Kasera, *"SocialSwarm: Exploiting Distance in Social Networks for Collaborative Flash File Distribution,"* in Proc. of IEEE International Conference on Network Protocols (ICNP), October 2010.

Papers in Workshops

1. S. K. Kasera, J. Phillips, and N. Patwari, "*Detecting and Localizing Spectrum Offenders Using Crowdsourcing*," IEEE DySPAN Workshop on Foundations in Spectrum Management Res., September 2015.
2. B. Nguyen, A. Banerjee, V. Gopalakrishna, S.J. Lee, Sneha Kumar Kasera, A. Shaikh, and J. Van der Merwe, "*Towards Understanding TCP Performance on LTE/EPC Mobile Networks*," in 4th ACM Sigcomm Workshop on All Things Cellular: Operations, Applications, and Challenges, August 2014.
3. M. Bocca, S. Gupta, O. Kaltiokallio, B. Mager, Q. Tate, S.K. Kasera, N. Patwari, S. Venkatasubramanian, "*RF-based device-free localization and tracking for ambient assisted living,*" Evaluating AAL Systems through Competitive Benchmarking (EvAAL) Workshop, September 2012.

Invited, Other Papers Published or Presented

1. J. Novak, S. Kasera, N. Patwari, "*Preventing Wireless Network Configuration Errors in Patient Monitoring Using Device Fingerprints*," in the 4th IEEE International Workshop on Data Security and Privacy in Wireless Networks, June 2013.
2. A. Banerjee, M. Maheshwari, N. Patwari, and Sneha K. Kasera, "*Detecting Receiver Attacks in VRTI-based Device Free Localization*," in the 3rd IEEE International Workshop on Data Security and Privacy in Wireless Networks, June 2012.
3. M. Maheshwari, S. Ananthanarayanan, A. Banerjee, N. Patwari, and Sneha K. Kasera, "*Detecting Malicious Nodes in RSS-based Localization*," in the 2nd IEEE International Workshop on Data Security and Privacy in Wireless Networks, June 2011.

Reviewed Poster Papers/Demos in Top Conferences

1. J. Croft, N. Patwari, and Sneha K. Kasera, "*Bit Extraction from Received Signal Strength*," Demonstration paper, in ACM MOBICOM, September 2010.

Disclosures and Patents

1. C. Becker, Sneha Kumar Kasera, and J. van der Merwe, "*Open Access Network Secure Authentication Systems and Methods,*" U.S. Patent filed, January 2016.
2. A. Banerjee, Sneha Kumar Kasera, and J. van der Merwe, "*Adaptive Group Paging for a Communication Network*," U.S. Patent filed, October 2015.
3. Suman Jana and Sneha K. Kasera, "*Method and System for Detecting Unauthorized Wireless Access Points Using Clock Skews*," U.S. Patent 9,049,225, June 2015.
4. Joe Novak and Sneha Kumar Kasera, "*Auto-tuning Active Queue Management*," U.S. Patent filed, May 2015.
5. N. Patwari and Sneha K. Kasera, "*Robust Location Distinction Using Temporal Link Signatures*," U.S. Patent 8,989,764, March 2015.
6. J. van der Merwe, R. Ricci, Sneha Kumar Kasera, M. Strum, R. Peterson, J. Peterson, "*Programmable Data Network Management and Operation*," PCT/U.S. Patent filed, August 2013.
7. N. Patwari, J. Croft, S. Jana, and Sneha K. Kasera, "*Method and System for High Rate Uncorrelated Shared Secret Bit Extraction from Wireless Link Characteristics*," U.S. Patent 8,515,061, August 2013.
8. Sneha K. Kasera, N. Patwari, J. Croft. S. Jana, "*Method and System for Secret Key Exchange Using Wireless Link Characteristics and Random Device Movement*," U.S. Patent 8,503,673, August 2013.
9. Sneha K. Kasera, J. Pinheiro, C. Loader, M. Karaul. A. Hari and T. LaPorta, "*Fast and Robust Signaling Overload Control*," U.S. Patent 7,792,252, June 2010.

TEACHING & MENTORING (2009 – present)

University of Utah Salt Lake City, UT
Assistant/Associate/Full Professor, School of Computing 2003-present

Lectured Classes

- Network Security, CS 5490/6490, Fall 2011 - 2016, Spring 2009
- Wireless & Mobile Networks, CS 6956, Spring 2013 - 2015
- Computer Networks, CS 5480/6480, Spring 2012
- Computer Networks, CS 5480, Spring 2010

Seminar Classes

- Networking Research, CS 7943-001, Spring 2016, Spring 2015
- Mobile Networking, CS 7943-001, Spring 2012
- Advanced Network Security, CS 7943-001, Spring 2010, Spring 2009

Chair of PhD and MS Thesis Student Committees

- **Current PhD/MS Thesis Students (total 9):** Philip Lundrigan (PhD), Mojgan Khaledi (PhD), Jared Plumb (PhD), Joe Novak (PhD), Christopher Becker (PhD), Aniqua Baset (PhD), Shamik Sarkar (PhD), Aarushi Sarbhai (MS Thesis), Shobhi Maheshwari (MS Thesis)
- **Graduated Students (PhD/MS Thesis, Defense date) (total 11):** Arijit Banerjee (PhD, October 2nd 2015), Sriram Premnath (PhD, February 6th 2013), Matthew Probst (PhD, November 7th 2012), Junxing Zhang (PhD,

May 24th 2010), Robert Ricci (PhD, January 5th 2010), Naveen D.S. (MS Thesis, May 1st 2014), Prarthana Lakshmane Gowda (MS Thesis, July 5th 2012), Manas Maheshwari (MS Thesis, May 13th 2011), Manav Seth (MS Thesis, May 9th 2011), Michael Clark (MS Thesis, May 5th 2010), S. Jana (MS Thesis, May 1st 2009)

SERVICE (2010 – present)

Professional External Service

Associate Editor

1. IEEE Transactions on Mobile Computing (2011 - 2015)
2. IEEE/ACM Transactions on Networking (2009 - 2013)
3. ACM/Springer Wireless Networks Journal (2007 - 2011)
4. Computer Networks Journal (COMNET) (2005 - 2010)

Technical Program Committee Chair

1. Technical Program Committee Co-Chair – ACM Sigsac 10th Conference on Security and Privacy in Wireless and Mobile Networks (WiSec), Boston, 2017
2. Technical Program Committee Co-Chair – ACM Sigmobile 21st Annual International Conference on Mobile Computing and Networking (MOBICOM), Paris, France, 2015
3. Technical Program Committee Co-Chair – IEEE International Conference on Networks Protocols (ICNP), Vancouver, BC, 2011
4. Technical Program Committee Co-Chair – IEEE Conference on Sensor, Mesh, and Ad Hoc Communications, and Networks (SECON), Salt Lake City, UT, 2011

Other Conference/Workshop Organization

- Steering Committee Member - IEEE Conference on Sensor, Mesh, and Ad Hoc Communications, and Networks (SECON), 2012-15
- Technical Program Committee Area Chair - IEEE Conference on Sensor, Mesh, and Ad Hoc Communications, and Networks (SECON), 2014
- Technical Program Committee Area Chair - IEEE Conference on Sensor, Mesh, and Ad Hoc Communications, and Networks (SECON), 2012
- Co-chair and organizer, Army Research Office Workshop on Fingerprinting, 2011
- Technical Program Committee Area Chair – IEEE International Conference on Network Protocols, 2010
- Finance Co-Chair - ACM Sigmobile International Conference on Mobile Computing and Networking (MOBICOM), 2010
- Student Poster Co-Chair - IEEE International Conference on Network Protocols, 2010

Technical Program Committee Member (total 21): ACM MOBICOM 2016, ACM WiSec 2016, ACM MOBIHOC 2016, ACM WiSec 2015, ACM MOBIHOC 2014, COMSNETS 2014, IEEE ICNP 2013, ACM MOBICOM 2013, ACM MOBICOM 2012, IEEE ICDCS 2012, ACM SIGMETRICS 2010, IEEE SECON, 2010, IEEE INFOCOM 2010, ACM MOBICOM 2010, IEEE ICNP, 2010, ACM MOBICOM 2009

Review Panels (total 14): National Science Foundation – 13 panels, Department of Energy – 1 panel

K -12 Outreach

- Co-organized three-day Summer Camps for high school students, June 2009-2011.
- Recruited 19 undergraduate students for developing teaching/learning modules for high school students that demonstrate the use of pre-calculus and high school science concepts in Computer Science.
- Visited high schools in the Salt Lake City area for Computer Science sessions.
- Salt Lake Valley Science and Engineer Fair Judging, 2011.

HONORS (2010 – present)

- "Highest Accuracy Award" at the "2012 Evaluating Ambient Assisted Living Systems through Competitive Benchmarking (EvAAL) Competition on Indoor Localization and Tracking for AAL," Madrid, Spain, September 2012.
- IEEE Computer Society 2011 Golden Core Award for leadership and service.

Ladislav Kavan

University of Utah

Research Interests

Computer graphics and animation, deformation modeling, medical applications, numerical methods, physics-based simulation, combining data and physics, applied geometry, real-time rendering.

Education

Ph.D., Computer Science March 2004 – June 2007
Faculty of Electrical Engineering, Czech Technical University, Prague

Mgr. (summa cum laude), Computer Science September 1998 – September 2003
(Combined B.Sc. and M.Sc. degrees)
Faculty of Mathematics and Physics, Charles University, Prague

Recent Journal Articles

1. **Fast and Robust Inversion-Free Shape Manipulation**
   Liu T., Gao M., Zhu L., Sifakis E., Kavan L.
   In: *Computer Graphics Forum (Proceedings of Eurographics 2016).*
2. **Computational Bodybuilding: Anatomically-based Modeling of Human Bodies**
   Saito S., Zhou Z., Kavan L.
   In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2015).*
3. **Linear Subspace Design for Real-Time Shape Deformation**
   Wang Y., Jacobosn A., Barbič J., Kavan L.
   In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2015).*
4. **Skuller: A Volumetric Shape Registration Algorithm for Modeling Skull Deformities**
   Sahillioglu Y., Kavan L.
   In: *Medical Image Analysis*, 2015.
5. **Adaptable Anatomical Models for Realistic Bone Motion Reconstruction**
   Zhu L., Hu X., Kavan L.
   In: *Computer Graphics Forum (Proceedings of Eurographics 2015).*
6. **Reducing Numerical Dissipation in Smoke Simulation**
   Huang Z., Kavan L., Li W., Hui P., Gong G.
   In: *Graphical Models*, 2015.
7. **A Simple Method for Correcting Facet Orientations in Polygon Meshes Based on Ray Casting**
   Takayama K., Jacobson A., Kavan L., Sorkine-Hornung O.
   In: *Journal of Computer Graphics Techniques*, 2014.
8. **Projective Dynamics: Fusing Constraint Projections for Fast Simulation**
   Bouaziz S., Martin S., Liu T., Kavan L., Pauly M.
   In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2014).*
9. **Ink-and-Ray: Global Illumination for Hand-Drawn Animation**
   Sykora D., Kavan L., Cadik M., Jamriska O., Jacobson A., Whited B., Simmons M., Sorkine O.
   In: *ACM Transactions on Graphics (Presented at ACM SIGGRAPH 2014).*

10. **Basis Enrichment and Solid-fluid Coupling for Model-reduced Fluid Simulation**
Gerszewski D., Kavan L., Sloan P.-P., Bargteil A.
In: *Computer Animation and Virtual Worlds*, 2014.

11. **Fast Simulation of Mass-Spring Systems**
Liu T., Bargteil A., O'Brien J., Kavan L.
In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2013).*

12. **Anatomy Transfer**
Dicko A., Gilles B., Liu T., Kavan L., Cani M.-P., Palombi O., Faure F.
In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2013).*

13. **Locally Injective Mappings**
Schüller C., Kavan L., Panozzo D., Sorkine O.
In: *Computer Graphics Forum (Proceedings of SGP 2013).*

14. **Robust Inside-Outside Segmentation using Generalized Winding Numbers**
Jacobson A., Kavan L., Sorkine O.
In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2013).*

15. **Elasticity-Inspired Deformers for Character Articulation**
Kavan L., Sorkine O.
In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2012).*

16. **Fast Automatic Skinning Transformations**
Jacobson A., Baran I., Kavan L., Popovic J., Sorkine O.
In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2012).*

17. **Physics-inspired Upsampling for Cloth Simulation in Games**
Kavan L., Gerszewski D., Bargteil A., Sloan P.-P.
In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2011).*

18. **Least Squares Vertex Baking**
Kavan L., Bargteil A., Sloan P.-P.
In: *Computer Graphics Forum (Proceedings of EGSR 2011).*

Recent Peer-Reviewed Conference Papers

19. **Fast Grid-Based Nonlinear Elasticity for 2D Deformations**
Rajsekhar S., Wang Y., Mitchell N., Kavan L., Sifakis E.
In: *Eurographics/ACM SIGGRAPH Symposium on Computer Animation 2014.*

20. **Sound Localization and Multi-Modal Steering for Autonomous Virtual Agents**
Wang Y., Kapadia M., Huang P., Kavan L., Badler N.
In: *ACM SIGGRAPH I3D 2014 Symposium.*

21. **Ambient Obscurance Baking on the GPU**
Sloan P.-P., Tranchida J., Chen H., Kavan L.
In: *ACM SIGGRAPH Asia Technical Briefs 2013.*

22. **Enhancements to Model-Reduced Fluid Simulation**
Gerszewski D., Kavan L., Sloan P.-P., Bargteil A.
In: *ACM SIGGRAPH Motion in Games 2013.*

Patents

23. **Continuum Model for Position Based Dynamics**
Kavan L.
In: *US Patent 20130103358, 2013.*

24. **Physics-inspired Upsampling for Cloth Simulation**
Kavan L., Sloan P.-P.
In: *US Patent 20130046522, 2013.*

Awards and Funding

NSF CAREER 2014 – 2019
Geometric Shape Deformation with Applications in Medicine $550,000
PI: Kavan

Fulbright Scholarship (MIT) 2007
Algorithms for Real-time Computer Animation approx. $75,000
PI: Kavan (award not accepted due to other offers)

Professional Activities

Associate Editor for

ACM Transactions on Graphics (since 2013)

Graphical Models (since 2012)

Program Committee Member

ACM SIGGRAPH 2015, 2014
ACM SIGGRAPH Asia 2013, 2012
Eurographics 2015
Shape Modeling International 2014, 2013
Mathematical Progress in Expressive Image Synthesis 2014
Eurographics Short Papers 2013
Symposium on Geometry Processing 2015, 2014, 2013, 2012
ACM SIGGRAPH I3D Symposium 2014, 2013, 2012, 2011
ACM/Eurographics Symposium on Computer Animation 2015, 2014, 2013, 2012, 2011
Eurographics 2012 State-of-the-Art Reports
3D Imaging, Modeling, Processing, Visualization & Transmission 2012
Pacific Graphics 2014, 2013, 2010

Panelist

NSF (two panels in 2014)

Chair

ACM/Eurographics Symposium on Computer Animation Posters 2013

Eurographics 2007 (Game Industry Presentations co-chair, with Steven Collins)

Paper Reviewer for

ACM SIGGRAPH, ACM SIGGRAPH Asia, ACM Transactions on Graphics, IEEE TVCG, Eurographics, Computer Graphics Forum, EGSR, Pacific Graphics, SCA, I3D, C&G, GMOD, VRIPhys, WSCG, Web3D, AES, SVE, CASA, 3IA, CEMVRC

Teaching Experience

Instructor Spring 2016
CS 6660: Physically Based Animation University of Utah

Instructor Fall 2015, Fall 2014, Fall 2013
EAS 205: Applications of Scientific Computation University of Pennsylvania

Instructor	Spring 2015, Spring 2014, Spring 2013
CIS 563: Physically Based Animation	University of Pennsylvania
Seminar Co-organizer (with H. Zimmer)	Spring 2012
Advanced Methods in Computer Graphics	ETH Zürich
Instructor	Summer 2005
Real-time Computer Animation	Czech Technical University, Prague
Teaching Assistant	Spring 2005
Theoretical Computer Science	Czech Technical University, Prague
Teaching Assistant	Spring 2004, Fall 2004
C++ Programming Language	Czech Technical University, Prague
Teaching Assistant	Spring 2005
Algorithms	Czech Technical University, Prague

Recent Colloquia, Courses, and Invited Talks

Stanford University	May 2015
Interactive Human Modeling and Simulation	Stanford, CA
University of Utah	April 2015
Interactive Human Modeling and Simulation	Salt Lake City, UT
Walt Disney Animation Studios	April 2015
Real-time Simulation of Deformable Objects	Burbank, CA
University of California, Los Angeles	April 2015
Interactive Human Modeling and Simulation	Los Angeles, CA
Bespoke	April 2015
Interactive Human Modeling and Simulation	San Francisco, CA
Pixar	March 2015
Interactive Human Modeling and Simulation	Emeryville, CA
Carnegie Mellon University	February 2015
Real-time Simulation of Deformable Objects	Pittsburgh, PA
IST Austria	February 2015
Real-time Simulation of Deformable Objects	Klosterneuburg, Austria
IST Austria	February 2015
Special Topics in Interactive Elasticity	Klosterneuburg, Austria
HiVisComp	February 2015
Fast Simulation of Mass-Spring Systems	Tatranska Lomnica, Slovakia
VirtaMed	February 2015
Real-time Simulation of Deformable Objects	Zürich, Switzerland
University of California, San Diego	October 2014
Modeling Deformable Objects using Direct and Physics-based Methods	San Diego, CA
SIGGRAPH 2014 Course (with A. Jacobson, Z. Deng, and J.P. Lewis)	August 2014
Skinning: Real-time Shape Deformation	Vancouver, Canada

Robert Kessler Curriculum Vitae

Education

Ph.D., Computer Science, University of Utah, 1981, M.S., Computer Science, University of Utah, 1977, B.S., Computer Science, University of Utah, 1974 (Magna cum Laude)

Professional Employment

- July 2010 to present, Executive Director, Entertainment Arts and Engineering – EAE designated Qualified Independent Teaching Program (approved by Academic Senate May 2013); Master's of Entertainment Arts and Engineering (with three emphases) (approved by Board of Regents March 2013)
- July 2009 to Dec 2011, Phased retirement (0.75 FTE), Retirement cancelled Jan 1, 2012
- July 2006 to Dec 2010, Associate Director, School of Computing
- July 1997 to June 2000, Chairman of Computer Science
- July 1997 to present, Professor of Computer Science (University of Utah)
- November 1996 to December 2005, Member, Board of Directors, emWare Inc., Salt Lake City, Utah (emWare's assets were sold to a large Japanese company and effectively ceased to exist).
- December 1995 to June 1996, Acting Chairman of Computer Science
- July 1994 to August 1995, June 1996 to September 1996, Visiting Scientist, Hewlett-Packard Research Labs, Palo Alto, CA.
- July 1993 to June 1994, July 1995 to Dec 1995, Associate Chairman of Computer Science.
- December 1991 to December 1994, Chairman of Board and Founder, Hippo Software Inc, Salt Lake City, Utah.
- July 1990 to September 1994, Director, Center for Software Science, University of Utah.
- July 1987 to June 1997, Associate Professor of Computer Science, University of Utah (received tenure beginning July 1990).
- July 1983 to June 1987, Research Assistant Professor of Computer Science, University of Utah.
- November 1982 to February 1996, Chairman of Board and Founder, Medical Software Systems, Salt Lake City, Utah.
- November 1982 to June 1983, Research Computer Scientist, University of Utah.

Research/Scholarship (since 2007)

- Principal Investigator, "EAE Lab Hardware", Intel, $40,000 (equipment gift), CPUs and SSDs for our EAE graduate lab, 7/1/2014.
- Principal Investigator, "Vein Craft", College of Nursing and AVA Foundation, $25,000 (Gift), 1/1/2015 – 5/31/2015.
- Administrator, "Games and Simulations", Rockwell Collins, $142,481, 1/1/2014 – 12/31/2014 (Mark van Langeveld PI, Roger Altizer and Jose Zagal – Co-PIs).
- Principal Investigator, "Reflex Game", Reflexology, $53,604, 9/1/2013 – 6/30/2014.
- Co- Principal Investigator, "Diabetes Game", Center for Medical Innovation, $25,000, 5/1/2013 – TBD.
- Co-P Principal Investigator, "National Energy Foundation Game", $33,000, 5/1/2013 – 8/15/2013 (Roger Altizer, PI)

- Principal Investigator, “Smarty Pants: Pets” TBRIS Technologies, LLC, 09/01/2012 - 11/15/2012. $17,000.
- Principal Investigator, “Cat Siena Game” Siena Entertainment, 02/01/2012 - 06/30/2012. $11,500.00.
- Co-PI, “PE Video Game,” University of Utah Dept of Pediatrics, $35,000, Summer 2011 (Roger Altizer, P.I.), Additional $50K funding received from Jack Britain’s office in 2012.
- Principal Investigator, “EAE Summer Camp”, State ASTEC, $7,500. July 2007-2008.

Note – Prior to 2007, grants totaling $2.4M as PI, cash development grant totaling $2.6M as PI, and equipment/software donations totaling over $8M as PI. Millions of dollars of grants as Co-PI or Faculty Associate.

Honors

- EAE undergraduate program ranked by Princeton Review #1, #2, #2, #1, #3, #2 in 2016, 2015, 2014, 2013, 2012, 2011 respectively.
- EAE graduate program ranked by Princeton Review #3, #1, #4, #2 in 2016, 2015, 2014, 2013 respectively.
- Our paper, “Strengthening the Case for Pair Programming” was chosen as one of the top 30 papers out of over 1000 papers submitted to IEEE Software over the past 25 years. This was described in the Jan/Feb 2009 issue of IEEE Software.
- University of Utah Distinguished Teaching Award, 2001.
- Nominated for University Distinguished Teaching Award, 1999, 2000.
- College of Engineering Outstanding Teaching Award for 2000.

Books

- Laurie Williams and Robert Kessler, “Pair Programming Illuminated,” Addison Wesley, July 2002, 265 pages
- “LISP, Objects, and Symbolic Programming,” Scott, Foresman/Little, Brown, January 1988, 656 pages.

Refereed Publications Since 2007

- “The Intersection of Video Games and Patient Empowerment: case study of a real world application”, Interactive Entertainment 2013 (September 2013) (C. Caldwell; C. Bruggers; R. Altizer; T. D’Ambrosio; R. Kessler; B. Christiansen)
- “Patient-Empowerment Interactive Technologies” *Sci. Transl. Med.* **4**, 152ps16 (2012) (C. S. Bruggers, R. A. Altizer, R. R. Kessler, C. B. Caldwell, K. Coppersmith, L. Warner, B. Davies, W. Paterson, J. Wilcken, T. A. D’Ambrosio, M. L. German, G. R. Hanson, L. A. Gershan, J. R. Korenberg, G. Bulaj)
- “When the games industry and academia collide: How we impact each other” 2012 IEEE International Games Innovation Conference (September 2012), pg. 1-4 (C. Caldwell; R. Kessler; R. Altizer; M. Van Langefeld) [Winner – Best Paper in Education Track]
- “Digital Visualization Tools Improve Teaching 3D Character Modeling,” SIGCSE 2010 (M. van Langeveld and R. Kessler)
- “Educational Impact of Digital Visualization and Auditing Tools On a Digital Character Production Course,” ICFDG 2009 (M. van Langeveld and R. Kessler)

- "Entertainment Arts and Engineering or How to Fast Track A New Interdisciplinary Program)," SIGCSE 2009 (R. Kessler, M. van Langeveld, and R. Altizer).
- "Two in the Middle: Digital Character Production and Machinima Courses," SIGCSE 2009 (M. van Langeveld and R. Kessler)
- "A History of Computing Course with a Technical Focus," SIGCSE 2009 (G. Draper, R. Kessler and R. Riesenfeld).

Patents

- U.S. Patent Pending "Empowering Patients During Disease Therapy Using An Interactive Video Game That Links Exercise and Positive Visualization", Grzegorz Bulaj, Carol S. Bruggers, Roger A. Altizer, Robert Kessler, Craig Caldwell, Wade R. Patterson, Kurt J. Coppersmith, Laura M. Warner, Brandon Davies. Filed: May 2012.

Published Games

- 42 different video games have been published by our senior and master's students since 2010. My role was executive producer as faculty advisor.

Courses Developed since 2007

- CS6070/FILM6711, EAE:MGS Game Projects I – a class for learning how to develop games using a rapid prototyping style of a new game every four weeks.
- CS6071/FILM6712, EAE:MGS Game Projects II – a class where game ideas are pitched and then narrowed down to a game chosen by faculty and industry to be built in a large team of students.
- CS6072/FILM6713, EAE:MGS Game Projects III – this class takes the alpha version of the game and creates a beta version and submits to a festival.
- CS6073/FILM6714, EAE:MGS Game Projects IV – polishing the game and publishing it.

Graduated 10 Doctoral. 1 MPhil, and 24 MS Students as Chairman. Served on many, many doctoral and masters committees.

Entertainment Arts and Engineering

- 10 lecturing and joint appointed faculty
- 4 full time and 2 part time staff
- 130 master's graduate students
- 300 to 400 undergraduate emphasis students
- $3.8M annual budget
- 14K sq ft remodeled space in old Law Library building
- Ranked #1 undergraduate or graduate program in the world three of the last four years

Internal Service Highlights (since 2007)

- Negotiated $20K cash donation female EAE graduate scholarships from Intel 2016
- Created Master's of Entertainment Arts and Engineering graduate degree (2013)
- EAE becomes Qualified Interdisciplinary Teaching Program (2013)
- Co-Founded EAE master's program (2010)
- Driving force behind creation of the Entertainment Arts and Engineering interdisciplinary (CS and Film) undergraduate track (2007)

- Steered the reorganization and revamping of our entire computational infrastructure including eliminating the research charge facility (2007-2011)
- Chairman of University SCAC committee (2005-2013)

External service – typical activities, editorial board, reviewing, vice chairman of SIGPLAN, etc.

Media Appearances – too many to count. The Spring 2013 issue of Continuum includes an excellent spread on the EAE program. (http://continuum.utah.edu/features/game-on)

Biographical Sketch

Dr. Robert M. Kirby

School of Computing	phone: 801-585-3421
University of Utah	fax: 801-585-6513
50 S. Central Campus Dr. RM 3190	kirby@cs.utah.edu
Salt Lake City, UT 84112	http://www.cs.utah.edu/~kirby

A. PROFESSIONAL PREPARATION

Florida State University	Tallahassee, FL	Applied Mathematics and Computer Science	B.S., 1997
Brown University	Providence, RI	Applied Mathematics	Sc.M., 1999
Brown University	Providence, RI	Computer Science	Sc.M., 2001
Brown University	Providence, RI	Applied Mathematics	Ph.D., 2002

B. APPOINTMENTS

University of Utah	Professor of Computing	07/2014 – present
Imperial College London	Leverhulme Visiting Professor in Aeronautics	12/2008 – 06/2009
Imperial College London	Visiting Academic	09/2008 – 11/2008
University of Utah	Associate Professor of Computing	07/2008 – 06/2014
University of Utah	Assistant Professor of Computing	09/2002 – 06/2008
Brown University	Research Associate	09/1997 – 08/2002

D. LEADERSHIP ROLES

- Associate Director, School of Computing, University of Utah, 2014 – present.
- Director and Program Manager, Multi-Scale Multidisciplinary Modeling of Electronic Materials (MSME) Collaborative Research Alliance (CRA), University of Utah, 2016 – present.
- Assistant Program Manager, Multi-Scale Multidisciplinary Modeling of Electronic Materials (MSME) Collaborative Research Alliance (CRA), University of Utah, 2012 – 2016.
- Chair, Scientific Computing Track, Computing PhD Program, School of Computing, University of Utah, 2004 – 2014.
- Director of Graduate Studies, School of Computing, University of Utah, 2012 – 2014.
- Associate Director, Computational Engineering and Science Program, University of Utah, 2004–2005, 2012 – 2014.
- Senator, Academic Senate, University of Utah, 2006 – 2008.
- Director, Computational Engineering and Science Program, University of Utah, 2005 – 2008.
- Associate Director, Scientific Computing and Imaging Institute, University of Utah, 2006–2007.

E. PUBLICATIONS (selected from over 140 publications)

1. Carlo Forestiere, Yanyan He, Ren Wang, Robert M. Kirby and Luca Dal Negro, "Inverse Design of Metal Nanoparticles' Morphology", *ACS Photonics*, In Press, 2015.
2. Hadi Meidani, Justin B. Hooper, Dmitry Bedrov and Robert M. Kirby, "Calibration and Ranking of Coarse-Grained Models in Molecular Simulations Using Bayesian Formalism", *International Journal for Uncertainty Quantification*, In Press, 2015.

3. Yanyan He, Mahsa Mirzargar, Sophia Hudson, Robert M. Kirby and Ross T. Whitaker, "An Uncertainty Visualization Technique Using Possibility Theory: Possibilistic Marching Cubes", *International Journal for Uncertainty Quantification*, In Press, 2015.

4. X. Li, J.K. Ryan, R.M. Kirby and C. Vuik, "Smoothness-Increasing Accuracy-Conserving (SIAC) Filters for Derivative Approximations of discontinuous Galerkin (DG) Solutions over Nonuniform Meshes and Near Boundaries", *Journal of Computational and Applied Mathematics*, In Press, 2015.

5. Mahsa Mirzargar, Jennifer K. Ryan and Robert M. Kirby, "Smoothness-Increasing Accuracy-Conserving (SIAC) Filtering and Quasi-Interpolation: A Unified View", *Journal of Scientific Computing*, In Press, 2015.

6. Sergey Yakovlev, David Moxey, Robert M. Kirby and Spencer J. Sherwin,"To CG or to HDG: A Comparative Study in 3D", *Journal of Scientific Computing*, In Press, 2015.

7. Yanyan He, Mahsa Mirzargar and Robert M. Kirby,"Mixed Aleatory and Epistemic Uncertainty Quantification Using Fuzzy Set Theory", *International Journal of Approximate Reasoning*, Vol. 66, pages 1–15, 2015.

8. Varun Shankar, Grady B. Wright, Robert M. Kirby and Aaron L. Fogelson, "Augmenting the Immersed Boundary Method with Radial Basis Functions (RBFs) for the Modeling of Platelets in Hemodynamic Flows", *International Journal for Numerical Methods in Fluids*, In Press, 2015.

9. Mukund Raj, Mahsa Mirzargar, Robert M. Kirby and Ross T. Whitaker, "Evaluating Alignment of Shapes by Ensemble Visualization", *IEEE Computer Graphics and Applications*, In Press, 2015.

10. C.D. Cantwell, D. Moxey, A. Comerford, A. Bolis, G. Rocco, G. Mengaldo, D. de Grazia, S. Yakovlev, J-E Lombard, D. Ekelschot, B. Jordi, H. Xu, Y. Mohamied, C. Eskilsson, B. Nelson, P. Vos, C. Biotto, R.M. Kirby and S.J. Sherwin, "Nektar++: An open-source spectral/hp element framework", *Computer Physics Communications*, In Press, 2015.

11. Jennifer K. Ryan, Xiaozhou Li, Robert M. Kirby and Kees Vuik, "One-Sided Position-Dependent Smoothness-Increasing Accuracy-Conserving (SIAC) Filtering Over Uniform and Non-Uniform Meshes", *Journal of Scientific Computing*, In Press, 2014

12. Varun Shankar, Grady B. Wright, Robert M. Kirby and Aaron L. Fogelson, "A Radial Basis Function (RBF)-Finite Difference (FD) Method for Diffusion and Reaction-Diffusion Equations on Surfaces", *Journal of Scientific Computing*, Volume 63, pages 745-768, 2015.

13. Mahsa Mirzargar, Ross T. Whitaker and Robert M. Kirby, "Curve Boxplot: Generalization of Boxplot for Ensembles of Curves", *IEEE Transactions on Visualization and Computer Graphics (IEEE Visualization Issue)*, Volume 20, Number 12, pages 2654-2663, 2014.

14. Zhisong Fu, Sergey Yakovlev, Robert M. Kirby and Ross T. Whitaker, "Fast Parallel Solver for Levelset Equations on Unstructured Meshes", *Concurrency and Computation: Practice and Experience*, Volume 27, pages 1639-1657, 2015.

15. Liam C. Jacobson, Robert M. Kirby and Valeria Molinero,"How Short Is Too Short for the Interactions of a Water Potential? Exploring the Parameter Space of a Coarse-Grained Water Model Using Uncertainty Quantification", *Journal of Physical Chemistry B*, Volume 119, pages 8190–8202, 2014.

16. A. Bolis, C.D. Cantwell, R.M. Kirby and S.J. Sherwin, "h to p efficiently: Optimal implementation strategies for explicit time-dependent problems using the spectral/hp element method", *International Journal for Numerical Methods in Fluids*, Volume 75, Issue 8, pages 591-607, 2014.

17. Varun Shankar, Grady B. Wright, Aaron L. Fogelson and Robert M. Kirby, "A Radial Basis Function (RBF)-Finite Difference Method for the Simulation of Reaction-Diffusion Equations on Stationary Platelets within the Augmented Forcing Method", *International Journal for Numerical Methods in Fluids*, Volume 75, Issue 1, pages 1-22, 2014.

18. James King, Sergey Yakovlev, Zhisong Fu, Robert M. Kirby and Spencer J. Sherwin, "Exploiting Batch Processing on Streaming Architectures to Solve 2D Elliptic Finite Element Problems: A Hybridized Discontinuous Galerkin (HDG) Case Study", *Journal of Scientific Computing*, Volume 60, pages 457-482, 2014.

19. C.D. Cantwell, S. Yakovlev, R.M. Kirby, N.S. Peters and S.J. Sherwin, "High-order continuous spectral/hp element discretisation for reaction-diffusion problems on a surface", *Journal of Computational Physics*, Vol. 257, Part A, pages 813-829, 2014.

20. Zhisong Fu, T. James Lewis, Robert M. Kirby and Ross T. Whitaker, "Architecting the Finite Element Method Pipeline for the GPU", *Journal of Computational and Applied Mathematics*, Volume 257, pages 195-211, 2014.

21. Blake Nelson, Robert M. Kirby and Steven Parker,"Optimal Expression Evaluation Through the Use of Expression Templates When Evaluating Dense Linear Algebra Operations", *ACM Transactions on Mathematical Software*, Vol. 40, Issue 3, pages 21:1-21:21, 2014.

22. Ross Whitaker, Mahsa Mirzargar and Robert M. Kirby, "Contour Boxplots: A Method for Characterizing Uncertainty in Feature Sets from Simulation Ensembles", *IEEE Transactions on Visualization and Computer Graphics (IEEE Visualization Issue)*, Vol. 19, Issue 12, pages 2713-2722, 2013.

23. Zhisong Fu, Robert M. Kirby and Ross T. Whitaker, "A Fast Iterative Method for Solving the Eikonal Equation on Tetrahedral Domains", *SIAM Journal of Scientific Computing*, Vol. 35, No. 5, pages C473-C494, 2013.

24. Hanieh Mirzaee, Jennifer K. Ryan and Robert M. Kirby, "Smoothness-Increasing Accuracy-Conserving (SIAC) Filtering for Discontinuous Galerkin Solutions: Applications to Structured Tetrahedral Meshes", *Journal of Scientific Computing*, Vol. 58, No. 3, pages 690-704, 2014.

25. Tiago Etiene, Daniel Jönsson, Timo Ropinski, Carlos Scheidegger, Joao Comba, L. Gustavo Nonato, Robert M. Kirby, Anders Ynnerman and Claudio T. Silva, , "Verifying Volume Rendering Using Discretization Error Analysis", *IEEE Transactions on Visualization and Computer Graphics*, Vol. 20, No. 1, pages 140-154, 2014.

26. Blake Nelson, Robert M. Kirby, Robert Haimes, "GPU-Based Volume Visualization From High-Order Finite Element Fields", *IEEE Transactions on Visualization and Computer Graphics*, Vol. 20, No. 1, pages 70-83, 2014.

27. Dafang Wang, Robert M. Kirby, Rob S. MacLeod and Chris R. Johnson, "Inverse electrocardiographic source localization of ischemia: an optimization framework and finite element solution", *Journal of Computational Physics*, Vol. 250, Issue 1, pages 403-424, 2013.

28. Robert M. Kirby and Miriah Meyer, "Visualization Collaborations: Reflections on What Works and Why", *IEEE Computer Graphics and Applications*, Volume 33, Issue 6, pages 82-88, 2013.

29. Varun Shankar, Grady B. Wright, Aaron L. Fogelson and Robert M. Kirby, "A Study Of Different Modeling Choices For Simulating Platelets With The Immersed Boundary Method", *Applied Numerical Mathematics*, Vol. 63, pages 58-77, 2013.

30. Hanieh Mirzaee, James King, Jennifer K. Ryan and Robert M. Kirby, "Smoothness-Increasing Accuracy-Conserving (SIAC) Filters for Discontinuous Galerkin Solutions Over Unstructured Triangular Meshes", *SIAM Journal of Scientific Computing*, Vol. 35, No. 1, pages 212-230, 2013.

31. Chao Yang, Dongbin Xiu and Robert M. Kirby, "Visualization of Covariance and Cross-Covariance Fields", *International Journal for Uncertainty Quantification*, Vol. 3, Issue 1, pages 25-38, 2013.

32. Torben Patz, Tobias Preusser and Robert M. Kirby, "Ambrosio-Tortorelli Segmentation of Stochastic Images: Model Extensions, Theoretical Investigations and Numerical Methods", *International Journal of Computer Vision*, Vol. 103, Issue 2, pages 190-212, 2013.

33. Blake Nelson, Eric Liu, Robert Haimes and Robert M. Kirby, "ElVis: A System for the Accurate and Interactive Visualization of High-Order Finite Element Solutions", *IEEE Transactions on Visualization and Computer Graphics (IEEE Visualization Issue)*, Vol. 18, No. 12, pages 2325-2334, 2012.

F. INVITED TALKS (2015 Only)

1. (Speaker) Salt Lake City Data Science Meetup, Salt Lake City, UT. Presented a talk entitled "Ensemble Visualization and Uncertainty Characterization Using Generalized Notions of Data Depth", December 2015.
2. (Speaker) Imperial College London School of Computing and Department of Aeronautics, London, UK. Presented a talk entitled "Ensemble Visualization and Uncertainty Characterization Using Generalized Notions of Data Depth", November 2015.
3. (Speaker) University of East Anglia Department of Mathematics, Norwich, UK. Presented a talk entitled: "Ensemble Visualization and Uncertainty Characterization Using Generalized Notions of Data Depth", November 2015.
4. (Speaker) Rensselaer Polytechnic Institute Scientific Computation Research Center, Troy, NY. Presented a talk entitled: "Multiscale modeling and uncertainty quantification as part of 'Materials by Design'", October 2015.
5. (Speaker) National Hurricane Center, Florida International University. Presented a talk entitled: "Ensemble Visualization and Uncertainty Characterization Using Generalized Notions of Data Depth", October 2015.
6. (Speaker) Nektar++ Workshop 2015. Presented a talk entitled: "Nektar ++: A look into the future", July 2015.
7. (Speaker) 2015 MACH Conference (Annapolis, MD). Presented a talk entitled: "Surrogate-Based Bayesian Model Ranking of Atomistic Models", April 2015.

F. TEACHING AND MENTORING (2015 Only)

1. Teaching: Spring 2015: CS 2100 Discrete Structures. Approximately 120 sophomores and juniors.
2. Mentoring: Harshitha PV (MS), M.S. Srivatsa (MS), Vidhi Zala (MS), Ashok Jallepalli (PhD), James King (PhD), Yanyan He (postdoc), Mahsa Mirzargar (postdoc), Shankar Sastry (postdoc)

H. SYNERGISTIC ACTIVITIES

- **Awards:** Leverhulme Visiting Professorship, 2008-2009; Best paper award at International SuperComputing (ISC) 2016; Best paper award at ACM Solid and Physical Modeling Symposium 2008; Best paper award at Parallel and Distributed Systems: Testing and Debugging (PADTAD) 2007; Best paper award at Formal Methods for Industry Critical Systems (FMICS) 2007; Outstanding paper award at EuroPVM-MPI 2006; NSF CAREER Award NSF-CCF0347791 2004; Joukowski Award for Outstanding Dissertation 2003.
- **Software:** Co-Lead Architect and Developer of the Open-Source Simulation Software *Nektar++*: nektar.info
- **Editorial Boards:** Mathematics and Computers in Simulation (2007–2009).
- **Conference Organization:** *SciVis Papers Co-Chair*, IEEE Visualization 2016 and 2017, *General Chair*, ICOSAHOM 2014, *Panels co-chair*, IEEE Visualization 2007–2009. *Panel Organizer (with Prof. Claudio Silva, NYU-Poly)*, IEEE Visualization 2011.
- **Program Committee Member:** International Conference on High Performance Computing 2007; Thread Verification Workshop 2006.

BIOGRAPHICAL SKETCH

Provide the following information for the Senior/key personnel and other significant contributors in the order listed on Form Page 2.
Follow this format for each person. **DO NOT EXCEED FIVE PAGES.**

NAME: Alexander Lex

eRA COMMONS USER NAME (credential, e.g., agency login): ALEXANDERLEX

POSITION TITLE: Assistant Professor of Computer Science Assistant Professor

EDUCATION/TRAINING *(Begin with baccalaureate or other initial professional education, such as nursing, include postdoctoral training and residency training if applicable.)*

INSTITUTION AND LOCATION	DEGREE *(if applicable)*	Completion Date MM/YYYY	FIELD OF STUDY
Graz University of Technology (Graz, Austria)	BSc	09/2006	Computer Science
McMaster University (Hamilton, OT, Canada)	Grad. training	05/2007	Computer Science
Graz University of Technology (Graz, Austria)	MSc	07/2008	Computer Science
Graz University of Technology (Graz, Austria)	PhD	03/2012	Computer Science
Harvard University (Boston, MA, USA)	Postdoctoral	05/2015	Computer Science

A. Personal Statement

I develop interactive data analysis methods for experts and scientists. My primary research interests are interactive data visualization and analysis especially applied to molecular biology and pharmacology.
While certain analytical questions can or will be solvable through automatic means, I concern myself with the challenges that require human reasoning. Data analysis in support of addressing these challenges requires an interactive and visual approach that tightly integrates algorithms, statistics, and machine learning.
I am a co-founder and leader of the Caleydo Project, which is both, software that can be used by life science experts to visualize biomolecular data and pathways, but also a platform for implementing prototypes of radical visualization ideas.
My previous research with respect to visualization of genomic data and my formal training in computer science and visualization, as well as my numerous close collaborations with biologists for visualization in molecular biology are an excellent foundation to address the problems of visualization for biomolecular and clinical data for large patient cohorts and will allow me to significantly contribute to this project.

B. Positions and Honors

<u>Positions and Employment</u>

08/2008 - 03/2012	*Research Assistant*, Graz University of Technology, Austria
08/2010 - 09/2012	*Lecturer*, Graz University of Technology, Austria
08/2011 - 09/2011	*Visiting Researcher*, Center for Biomedical Informatics, Harvard Medical School, USA
03/2012 - 09/2012	*Post-Doctoral Researcher*, Graz University of Technology, Austria
10/2012 - 05/2015	*Post-Doctoral Fellow*, Harvard University
01/2015 - 05/2015	Lecturer, Harvard University
07/2015 -	Assistant Professor of Computer Science, University of Utah

<u>Other Experience and Professional Memberships</u>

2014 - 2015	*Organizing committee*, IEEE VIS
2014 - 2016	*Program committee*, IEEE InfoVis
2015 - 2016	*Program chair,* Symposium on Visualization in Data Science (VDS)
2012 - 2016	*Organizing committee*, IEEE Symposium on Biological Data Visualization (BioVis)
2012 - 2014	*Program committee*, Conference on Human-Computer Interaction & Knowledge Discovery
2008 - 2014	*Member*, Institute of Electrical and Electronics Engineers (IEEE)

<u>Awards and Honors</u>

2006	**Joint Study scholarship** for student exchange with McMaster University, Hamilton, On, Canada.
2007	Research grant for students. Faculty of Computer Science, Graz University of Technology.

2007	Award for excellent performance as a student. Graz University of Technology.
2010	Co-recipient **best student paper award**, ACM Graphics Interface
2011	Co-recipient of **best paper award**, IEEE InfoVis
2012	Recipient of **best dissertation award** by the Forum Technology and Society, TU Graz.
2012	Co-recipient of **best paper award**, IEEE BioVis
2012	Co-recipient of **3rd best paper award**, IEEE/Eurographics EuroVis
2013	**Erwin Schroedinger Postdoctoral Scholarship**, Austrian Science Fund (FWF)
2013	Co-recipient of **best paper award**, IEEE InfoVis
2014	Co-recipient of **honorable mention award**, ACM CHI
2015	Human Technology Interface Award for the work on cancer subtype visualization (StratomeX)

C. Contribution to Science

* indicates joint-first authorship

1. Cancer subtype analysis can improve our understanding of cancers and pave the way to improved patient outcomes. The investigation of cancer subtypes is highly data driven and makes use of large and heterogeneous data sets. These datasets are difficult to analyze and require both computational and advanced visual methods to facilitate human reasoning and exploration. Together with colleagues, I have developed methods to visualize patient stratifications based on various datasets, explore and investigate their impact on outcome, potential confounding factors, and their relationships to alternative stratifications. This body of work has profoundly influenced how cancer subtypes are visually analyzed and communicated.
    a. Streit M*, **Lex A***, Gratzl S, Partl C, Schmalstieg D, Pfister H, Park PJ, Gehlenborg N. Guided visual exploration of genomic stratifications in cancer. Nat Methods. vol 11, no 9, pp 884-5 PMCID: PMC4196637.
    b. **A. Lex**, M. Streit, H.-J. Schulz, C. Partl, D. Schmalstieg, P. J. Park, and N. Gehlenborg, “StratomeX: Visual Analysis of Large-Scale Heterogeneous Genomics Data for Cancer Subtype Characterization,” Computer Graphics Forum (EuroVis ’12), vol. 31, no. 3, pp. 1175–1184, 2012.
    c. C. Turkay, **A. Lex**, M. Streit, H. Pfister, and H. Hauser, “Characterizing Cancer Subtypes using the Dual Analysis Approach in Caleydo,” IEEE Computer Graphics and Applications, vol. 34, no. 2, pp. 38–47, Mar. 2014.
    d. **A. Lex**, H.-J. Schulz, M. Streit, C. Partl, and D. Schmalstieg, “VisBricks: Multiform Visualization of Large, Inhomogeneous Data,” IEEE Transactions on Visualization and Computer Graphics (InfoVis ’11), vol. 17, no. 12, pp. 2291–2300, 2011.

2. Biological networks, such as pathways, provide critical context to experimental data. However, the joint analysis of, for example, gene expression, copy number or mutation data with networks is hindered by the problems of visualizing multivariate networks. To remedy this, I have developed methods to visualize large and highly multivariate networks that convey both the topology of the network and detailed information about experimental data associated with user-selected subsets of the network.
    a. **A. Lex**, C. Partl, D. Kalkofen, M. Streit, A. M. Wasserman, S. Gratzl, D. Schmalstieg, and H. Pfister, “Entourage: Visualizing Relationships between Biological Pathways using Contextual Subsets,” IEEE Transactions on Visualization and Computer Graphics (InfoVis ’13), vol. 19, no. 12, pp. 2536-2545, 2013.
    b. C. Partl, **A. Lex**, Marc Streit, Denis Kalkofen, Karl Kashofer, and Dieter Schmalstieg, “enRoute: Dynamic Path Extraction from Biological Pathway Maps for Exploring Heterogeneous Experimental Datasets.” BMC Bioinformatics, vol. 14, no. Suppl 19, p. S3, Nov. 2013. PMCID: PMC3980897
    c. C. Partl, **A. Lex**, M. Streit, D. Kalkofen, K. Kashofer, and D. Schmalstieg, “enRoute: Dynamic Path Extraction from Biological Pathway Maps for In-Depth Experimental Data Analysis. Proceedings of the IEEE Symposium on Biological Data Visualization (BioVis '12) 2012, pp. 107-114, Seattle, WA, USA, Oct. 2012.
    d. M. Streit, **A. Lex**, M. Kalkusch, K. Zatloukal, and D. Schmalstieg, “Caleydo: Connecting Pathways and Gene Expression,” Bioinformatics, vol. 25, no. 20, pp. 2760–2761, 2009. PMCID: PMC2759551

3. In addition to visualization methods for biological data, I have also developed multiple fundamental data visualization techniques. I developed, for example, a method to visualize and analyze intersections of many

sets, providing an alternative to the often misused Venn diagrams. This technique has been used, for example, to evaluate various algorithms and parameterizations of single nucleotide variant calling algorithms. Other fundamental visualization techniques tackle the challenges of multivariate rankings, high-dimensional and heterogeneous data visualization, and effective highlighting across multiple views.

a. S. Gratzl, **A. Lex**, N. Gehlenborg, H. Pfister, and M. Streit, “LineUp: Visual Analysis of Multi-Attribute Rankings”. IEEE Transactions on Visualization and Computer Graphics (InfoVis ’13), vol. 19, no. 12, pp. 2277–2286, 2013.
b. **A. Lex**, M. Streit, C. Partl, K. Kashofer, D. Schmalstieg, “Comparative Analysis of Multidimensional, Quantitative Data.” IEEE Transactions on Visualization and Computer Graphics (InfoVis'2010), 16(6), pp. 1027-1035, Nov.-Dec. 2010
c. M. Steinberger, M. Waldner, M. Streit, **A. Lex**, and D. Schmalstieg, “Context-Preserving Visual Links,” IEEE Transactions on Visualization and Computer Graphics (InfoVis ’11), vol. 17, no. 12, pp. 2249–2258, Dec. 2011.
d. T. Geymayer, M. Steinberger, M. Streit, **A. Lex**, and D. Schmalstieg, “Show me the Invisible: Visualizing Hidden Content.” In proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14), pp. 3705-3714, 2014.

Complete List of Published Work in My Bibliography:
http://www.ncbi.nlm.nih.gov/sites/myncbi/alexander.lex.1/bibliography/48797429/public/?sort=date&direction=ascending

D. Research Support

Ongoing Research Support

U01 CA198935 Park, Peter (PI) 06/01/15-05/31/18
National Cancer Institute (NCI)
Visual analysis of genomic and clinical data from large patient cohorts
To develop new methods and software tools for integrating genomic and clinical data.
Role: Subcontract / Co-Investigator

Completed Research Support

Erwin Schroedinger Postdoctoral Scholarship Lex (PI) 01/05/12-31/08/15
Austrian Science Fund (FWF)
Visual Analysis of Heterogeneous Data using Semantic Subsets.
The goal of this project was to develop methods for a divide and conquer approach to visualizing large and heterogeneous data, as for example found in cancer subtype analysis.
Role: PI

CURRICULUM VITAE of Feifei Li

Webpage: http://www.cs.utah.edu/∼lifeifei E-mail: lifeifei@cs.utah.edu

Research Interests

Database systems and large-scale data management, systems, and analytics. Security issues in data management and systems.

Education

September 2002 - August 2007 Ph.D. in Computer Science, Computer Science Department, Boston University, PhD Thesis: "Query and Data Security Issues in the Data Outsourcing Model".

September 1998 - Jan 2002 B.S. in Computer Engineering, School of Computer Engineering, Nanyang Technological University, Singapore (transferred from Tsinghua University, China).

Honors and Awards

- SIGMOD Best Paper Award, ACM SIGMOD 2016.
- SIGMOD Best Demonstration Award, ACM SIGMOD 2015.
- Google Faculty Award, 2015.
- IEEE ICDE 2014 10+ Years Most Influential Paper Award.
- Teaching Dean's List, Top 15% Class in the College of Engineering, CS 6530, Fall 2013.
- SIGMOD Best Undergraduate Research Poster Award (Advisor), SIGMOD 2013.
- Google App Engine Education Award, 2012.
- HP Labs Innovation Research Award (61 awards selected from more than 500 worldwide submissions), 2012.
- HP Labs Innovation Research Award (62 awards selected from 626 worldwide submissions), 2011.
- COFRS (Committee on Faculty Research Support) Award, Florida State University, 2011.
- NSF Career Award, 2011.
- First Year Assistant Professor Award, Florida State University, 2008.
- Best Presenter Award, in research summer interns Seminar@Luch Series, IBM T.J. Watson Research Center, August, 2006.
- Best Paper Award, in 20th IEEE International Conference on Data Engineering (ICDE), 2004.
- First Class Honor (Accelerating Honor), School of Computer Engineering, Nanyang Technological University, Singapore, 2002.
- Defense Science & Technology Agency Gold Medal, Singapore, 2001.
- Merit Award in Singapore Advanced e-Business Applications Competition, 2000.
- National Science Class, China, 1997.

Professional Experience

- *ZhiYuan College, Shanghai Jiao Tong University*, Shanghai, China,
  Visiting Professor *July, 2014 - July, 2015*
- *School of Computing, University of Utah*, Salt Lake City, UT, USA
  Associate Professor *July, 2013 - Present*
- *School of Computing, University of Utah*, Salt Lake City, UT, USA
  Assistant Professor *Aug, 2011 - June, 2013*
- *Computer Science Department, Florida State University*, Tallahassee, FL, USA
  Assistant Professor & Director for the SAIT Laboratory *Aug, 2007 - Aug, 2011*
- *Database Research Group, Microsoft Research*, Redmond, WA, USA
  Research Summer Intern *May, 2007 - August, 2007*
- *Database Management Research Department, AT&T Shannon Research Lab*, Florham Park, NJ, USA
  Research Consultant *2007 - 2009*
- *Database Research Group, IBM T.J.Watson Research Center*, Hawthorne, NY, USA
  Research Summer Intern *June, 2006 - September, 2006*
- *SQL Server Group, Microsoft*, Redmond, WA, USA
  Summer Intern *June, 2005 - September, 2005*

Grants

1. "III: Small: Towards a Database Engine for Interactive and Online Sampling and Analytics", *sole PI*, NSF IIS, 09/01/16- 08/31/19, $500,000.
2. "NSF DSSP Workshop: Data Science for Secure and Privacy-Aware Big Data Management and Mining", *sole PI*, NSF, 09/07/16- 09/06/17, $98,000.
3. Research Experience for Undergraduate (for the NSF SEAL project), *PI*, NSF, 09/16-08/17, $16,000.
4. Research Experience for Undergraduate (for the NSF STORM project), *PI*, NSF, 09/16-08/17, $16,000.
5. "TWC: Medium: Collaborative: Seal: Secure Engine for AnaLytics - From Secure Similarity Search to Secure Data Analytics", *PI*, NSF SaTC, co-PI Jeff Phillips, PI at UCSB Rachel Lin, 07/15/15-06/30/19, $942,056: $600,007 at Utah and $342,049 at UCSB.
6. "Spatio-Temporal Online Analytics With Concept Enriched Text", *PI*, Google Research Awards program, 02/2015-02/2016, $57,640.
7. "Automated Query Engine for Large Heterogeneous Data", *PI*, NSFC (NSF China) Oversea Collaboration Grant, co-PI Bin Yao at Shanghai Jiao Tong University, 01/01/2015-12/31/2016, Chinese RMB 200,000.
8. "CIF21 DIBBs: STORM: Spatio-Temporal Online Reasoning and Management of Large Data", *PI*, NSF DIBBs program, co-PI John Horel, Jeff Phillips, Paul Rosen, 11/01/2014-10/31/2017, $1,157,975.
9. Research Experience for Undergraduate (for the NSF BIGDATA grant), *PI*, NSF, 04/14-04/15, $16,006.
10. "BIGDATA: Small: DCM: DA: Building a Mergeable and Interactive Distributed Data Layer for Big Data Summarization Systems", *PI*, NSF BIGDATA program, co-PI Jeff Phillips, 09/15/2013-08/31/2016, $685,380.
11. "TWC: Medium: TCloud: A Self-Defending, Self-Evolving and Self-Accounting Trustworthy Cloud Platform", *co-PI*, NSF, PI Jacobus Van Der Merwe, co-PIs Robert Ricci, Eric Eide, 09/01/2013-08/31/2017, $999,991.
12. "CSR: Medium: Energy-Efficient Architectures for Emerging Big-Data Workloads", *co-PI*, NSF CNS, PI Rajeev Balasubramonian, co-PIs Al Davis, Mary Hall, 07/01/2013-06/30/2017, $873,286.
13. Research Experience for Undergraduate (for the CAREER grant), *sole PI*, NSF, 07/12-07/13, $16,000.
14. "Time-Sensitive and Keyword-Aware Recommendation in Large Data", *sole-PI*, HP, 09/01/2011-08/31/2013, $125,000 (supported by two HP IRP awards at $45,000 in 2011-2012 with an additional $5,000 for collaboration support, and $70,000 in 2012-2013 with an additional $5,000 for collaboration support).
15. "Non-Conventional Search and Identification of Delinquent Parents", *co-PI (PI at Utah)*, PI: Sudhir Aggarwal (FSU), co-PI: Piyush Kumar (FSU), Florida Department of Revenue, 2/16/2011-9/29/2013, $632,465.
16. "Efficient Aggregate Similarity Search", *sole PI*, FSU COFRS (Committee on Faculty Research Support) Award, 2011, $14,000.
17. "CAREER: Novel Query Processing Techniques for Distributed Probabilistic Data", *sole PI*, NSF IIS Program, 02/11-01/16, $498,138.
18. "Scholarship for Service at Florida State University", *co-PI*, PI: Mike Burmester (FSU), co-PIs: Sudhir Aggarwal (FSU), Xiuwen Liu (FSU), NSF Division of Undergraduate Education, 09/10-08/14, $1,853,894.
19. "III:Small: Efficient Ranking and Aggregate Query Processing for Probabilistic Data", *sole PI*, NSF IIS Program, 09/09-09/12, $328,831.
20. Research Experience for Undergraduate (for the CT-ISG grant), *sole PI*, NSF, 06/09-09/11, $12,000.
21. "CT-ISG: Collaborative Research: Towards Trustworthy Database Systems", *sole PI at FSU*, NSF Cyber Trust Program, 10/08-09/11, $150,620.
22. DoD 2009 IASP (Information Assurance Scholarship Program) Grant, *co-PI*, PI: Mike Burmester (FSU), 08/09-08/10, $46,977.
23. "Building Trustworthy Database Systems", *sole PI*, FSU CRC Planning Grant, 12/01/07-11/30/08, $12,000.
24. "Query Verification for Distributed Databases", *sole PI*, FSU CRC First Year Assistant Professor Grant, 05/08-08/08, $16,000.
25. "Ranking and Aggregate Query Processing for Large Scientific Data with Fuzzy Information", Source: FSU Undergraduate Research Award, *sole PI*, Student: Justin DeBrabant, $4,000, 2009.

Selected Recent Publications

1. *Spell: Streaming Parsing of System Event Logs*, by M. Du, **F. Li**. In Proceedings of 16th IEEE International Conference on Data Mining (**IEEE ICDM 2016**), pages TBA, Barcelona, Spain, December, 2016.
2. *Fast And Concurrent RDF Queries With RDMA-Based Distributed Graph Exploration*, by J. Shi, Y. Yao, R. Chen, H. Chen, **F. Li**. In Proceedings of 12th USENIX Symposium on Operating Systems Design and Implebbmentation (**OSDI 2016**), pages TBA, Savannah, USA, November, 2016.
3. *Simba: Spatial In-Memory Big Data Analytics*, by D. Xie, **F. Li**, B. Yao, G. Li, L. Zhou, Z. Chen, M. Guo. Demo Paper. In Proceedings of 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (**ACM SIGSPATIAL 2016**), pages TBA, San Francisco, USA, November, 2016.
4. *Simba: Efficient In-Memory Spatial Analytics*, by D. Xie, **F. Li**, B. Yao, G. Li, L. Zhou, M. Guo In Proceedings of 35th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2016**), pages 1071-1085, San Francisco, USA, June, 2016.
5. *Wander Join: Online Aggregation via Random Walks*, by **F. Li**, B. Wu, K. Yi, Z. Zhao In Proceedings of 35th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2016**), pages 615-629, San Francisco, USA, June, 2016. **SIGMOD Best Paper Award**
6. *Graph Analytics Through Fine-Grained Parallelism*, by Z. Shang, **F. Li**, J. X. Yu, Z. Zhang, H. Cheng. In Proceedings of 35th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2016**), pages 463-478, San Francisco, USA, June, 2016.
7. *Matrix Sketching Over Sliding Windows*, by Z. Wei, X. Liu, **F. Li**, S. Shang, X. Du, J. Wen. In Proceedings of 35th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2016**), pages 1465-1480, San Francisco, USA, June, 2016.
8. *Privacy Preserving Subgraph Matching on Large Graphs in Cloud*, by Z. Chang, L. Zou, **F. Li**, In Proceedings of 35th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2016**), pages 199-213, San Francisco, USA, June, 2016.
9. *Oblivious RAM: A Dissection and Experimental Evaluation*, by Z. Chang, D. Xie, **F. Li**, In Proceedings of 42nd International Conference on Very Large Data Bases (**VLDB 2016**), pages 1113-1124, New Delhi India, 2016.
10. *Spatial Online Sampling and Aggregation*, by L. Wang, R. Christensen, **F. Li**, K. Yi, In Proceedings of 42nd International Conference on Very Large Data Bases (**VLDB 2016**), pages 84-95, New Delhi India, 2016.
11. *ATOM: Automated Tracking, Orchestration, and Monitoring of Resource Usage in Infrastructure as a Service Systems*, by M. Du, **F. Li**, In Proceedings of IEEE International Conference on Big Data (**IEEE BigData 2015**), pages 271-278, Santa Clara CA, November, 2015.
12. *Distributed Online Tracking*, by M. Tang, **F. Li**, Y. Tao, In Proceedings of 34th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2015**), pages 2047-2061, Melbourne, Australia, June 2015.
13. *STORM: Spatio-Temporal Online Reasoning and Management of Large Spatio-Temporal Data*, by R. Christensen, L. Wang, **F. Li**, K. Yi, J. Tang, N. Villa, In Proceedings of 34th ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2015**), system demo, pages 1111-1116, Melbourne, Australia, June 2015 (**SIGMOD Best Demonstration Award**).
14. *Scalable Histograms on Large Probabilistic Data*, by M. Tang, **F. Li**, In Proceedings of 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (**ACM SIGKDD 2014**), pages 631–640, NYC, NY, August 2014.
15. *Continuous Matrix Approximation on Distributed Data*, by M. Ghashami, J. Phillips, **F. Li**, In Proceedings of 40th International Conference on Very Large Databases (**VLDB 2014**), pages 809–820, Hangzhou, China, September 2014.
16. *Optimal Splitters for Temporal and Multi-version Databases*, by W. Le, **F. Li**, Y. Tao, R. Christensen, In Proceedings of 32nd ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2013**), pages 109-120, NYC, NY, June 2013.
17. *Quality and Efficiency for Kernel Density Estimates in Large Data*, by Y. Zheng, J. Jestes, J. Phillips, **F. Li**, In Proceedings of 32nd ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2013**), pages 433-444, NYC, NY, June 2013.
18. *Adaptive Log Compression for Massive Log Data*, by R. Christensen, **F. Li**, In Proceedings of 32nd ACM SIGMOD International Conference on Management of Data (**ACM SIGMOD 2013**, Undergraduate Research Poster), **SIGMOD Best Undergraduate Research Poster Award**, pages 1283-1284, NYC, NY, June 2013.
19. *Secure Nearest Neighbor Revisited*, by B. Yao, **F. Li**, X. Xiao, In Proceedings of 29th IEEE International Conference on Data Engineering, (**IEEE ICDE 2013**), pages 733-744, Brisbane, Australia, April, 2013.

20. *LogKV: Exploiting Key-Value Stores for Log Processing*, by Z. Cao, S. Chen, **F. Li**, M. Wang, X. Sean Wang, In Proceedings of 6th Biennial Conference on Innovative Data Systems Research (**CIDR 2013**), pages NA, Asilomar, California, January 2012.
21. *Ranking Large Temporal Data*, by J. Jestes, J. Phillips, **F. Li**, M. Tang, In Proceedings of 38th International Conference on Very Large Databases (**VLDB 2012**), PVLDB 5(11) pages 1412-1223, Istanbul, Turkey, August 2012.
22. *Building Wavelet Histograms on Large Data in MapReduce*, by J. Jestes, K. Yi, **F. Li**, In Proceedings of 38th International Conference on Very Large Databases (**VLDB 2012**), PVLDB 5(2): 109-120, Istanbul, Turkey, August 2012.
23. *Towards Fair Sharing of Block Storage in a Multi-tenant Cloud*, by X. Ling, Y. Mao, **F. Li**, R. Ricci, In Proceedings of 4th USENIX Workshop on Hot Topics in Cloud Computing (**USENIX HotCloud 2012**, pages TBA, Boston, MA, June 2012.
24. *Approximate Aggregation Techniques for Sensor Databases*, by J. Considine, **F. Li**, G. Kollios, and J. Byers. In Proceedings of the 20th IEEE International Conference on Data Engineering (**IEEE ICDE 2004**), pages 449-460, Boston, MA, March 30 - April 2, 2004. **Best Paper Award**

Selected Professional Service

1. PC Associate Chair, ACM SIGMOD 2015, 34th ACM SIGMOD/PODS International Conference on Management of Data and Principle of Database Systems.
2. Associate Editor, IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE), since November, 2013.
3. PC co-Chair, WAIM'14, 15th International Conference on Web Age Information Management.
4. General co-Chair, SIGMOD'14, 33rd ACM SIGMOD/PODS International Conference on Management of Data and Principle of Database Systems.
5. PC Area Vice Chair, ICDE'14, 30th IEEE International Conference on Data Engineering.
6. Demo PC Chair, VLDB'14, 40th International Conference on Very Large Databases.
7. PC members for the annunal SIGMOD, VLDB, ICDE conferences.
8. Panel members for NSF, HKRGC, Qatar National Research Fund (QNRF).

Teaching Experience

- Track Director, School of Computing, University of Utah. Director for the Data Track Graduate Program. Fall 2011 to Fall 2013.
- Instructor, ZhiYuan College, Shanghai Jiao Tong University. ACM Class, Database Systems and Big Data Management. Fall 2014 and Spring 2015.
- Instructor, School of Computing, University of Utah. Graduate Course: CS6530, Advanced Database Systems. Fall 2012, Fall 2013, Fall 2015, Fall 2016.
- Instructor, School of Computing, University of Utah. Undergraduate Course: CS5530 Database Systems. Spring 2012, Spring 2013, Spring 2014, Spring 2016
- Instructor, School of Computing, University of Utah. Graduate Course: CS7941 Data Reading Group. Spring 2012, Spring 2013, Fall 2013, Fall 2015.
- Instructor, School of Computing, University of Utah. Graduate Course: CS6931 Database Seminar. Spring 2012.
- Instructor, School of Computing, University of Utah. Graduate Course: CS6960 Database Kernels and Large Data Management. Fall 2011.
- Instructor, Computer Science Department, Florida State University. Undergraduate Course: COP4710 Theory and Structure of Databases. Fall 2010, Spring 2010, Fall 2009, Fall 2008, Spring 2008.
- Instructor, Computer Science Department, Florida State University. Graduate Course: COP5725 Database Systems. Spring 2011, Spring 2010, Spring 2009, Fall 2007.
- Instructor, Computer Science Department, Florida State University. Graduate Course: CIS5930 Advanced Topics in Data Management. Fall 2008.

Miriah Meyer

72 Central Campus Dr, Rm 3750, Salt Lake City, UT 84112
http://www.cs.utah.edu/∼miriah
miriah@cs.utah.edu
July 2016

Education

Ph.D. in Computer Science, 2008
University of Utah, Salt Lake City, UT
Thesis: *Dynamic Particle Systems for Adaptive Sampling of Implicit Surfaces*
Advisor: Ross Whitaker

B.S. in Astronomy & Astrophysics with Honors, 1999
Pennsylvania State University, University Park, PA
Minors: Physics, Women's Studies

Professional Appointments

USTAR Assistant Professor, 2011 – present
School of Computing
Scientific Computing and Imaging (SCI) Institute
University of Utah, Salt Lake City, UT

Visiting Scientist, 2010 – 2011
Broad Institute of MIT and Harvard, Cambridge, MA

Postdoctoral Research Fellow, 2008 – 2011
School of Engineering and Applied Sciences
Harvard University, Cambridge, MA
Supervisors: Hanspeter Pfister and Tamara Munzner

Awards

Outstanding Teaching Award, School of Computing, University of Utah, 2015
NSF CAREER Award, 2014
Best Paper Award, ACM AVI Conference, 2014
PopTech Science Fellow, 2013
TED Fellow, 2013
Microsoft Research Faculty Fellowship, 2012
FastCompany Magazine's 100 Most Creative People in Business, 2012
MIT Technology Review TR35: The Top 35 Innovators Under 35, 2011
NSF/CRA Computing Innovation Fellowship, 2009
AAAS Mass Media Fellowship, 2006

Selected Publications[1]

Journal Publications

L. Padilla, P. S. Quinan, M. Meyer, S. Creem-Regehr. *Evaluating the Impact of Binning 2D Scalar Fields*, Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), *accepted.*

A. Bigelow, S. Drucker, D. Fisher, M. Meyer. *Iterating Between Tools to Create and Edit Visualizations*, Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), *accepted.*

[1] underlined names indicate student authors

S. McKenna, D. Staheli, C. Fulcher, M. Meyer. *BubbleNet: A Cyber Security Dashboard for Visualizing Patterns*, Computer Graphics Forum (Proceedings of EuroVis), 35(3):281-290, 2016.

N. McCurdy, J. Lein, K. Coles, M. Meyer. *Poemage: Visualizing the Sonic Topology of a Poem*, IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), 22(1):439-448, 2016.

P. S. Quinan, M. Meyer. *Visually Comparing Weather Features in Forecasts*, IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), 22(1):389-398, 2016.

S. McKenna, M. Meyer, S. Gerber. *s-CorrPlot: An Interactive Scatterplot for Exploring Correlation*, Journal of Computational and Graphical Statistics, 2015.

E. Kerzner, L. Butler, C. Hansen, M. Meyer. *A Shot at Visual Vulnerability Analysis*, Computer Graphics Forum (Proceedings of EuroVis), 34(3):391-400, 2015.

M. Meyer, M. Sedlmair, P. S. Quinan, T. Munzner. *The Nested Blocks and Guidelines Model*, Journal of Information Visualization, 14(3):234-249, 2015.

S. McKenna, D. Mazur, J. Agutter, M. Meyer. *Design Activity Framework for Visualization Design*, IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), 20(12):2191-2200, 2014.

G. McInerny, M. Chen, R. Freeman, D. Gavaghan, M. Meyer, F. Rowland, D. Spiegelhalter, M. Steganer, G. Tessarolo, J. Hortal. *Information Visualization for Science & Policy: Engaging Users & Avoiding Bias*, Trends in Ecology & Evolution, 29(3):148-157, 2014.

R. Kirby, M. Meyer. *Visualization Collaborations: Reflections on What Works and Why*, IEEE Computer Graphics and Applications, 33(6):82-88, 2013.

A. Abdul-Rahman, J. Lein, K. Coles, E. Maguire, M. Meyer, M. Wynne, C. Johnson, A. Trefethen, M. Chen. *Rule-based Visual Mappings with a Case Study on Poetry Visualization*, Computer Graphics Forum (Proceedings of EuroVis), 32(3), 2013.

M. Sedlmair, M. Meyer, T. Munzner. *Design Study Methodology: Reflections from the Trenches and the Stacks*, IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), 18(12):2431-2440, 2012. **Best Paper Honorable Mention**.

C. Fowlkes, K. Eckenrode, M. Bragdon, M. Meyer, Z. Wunderlich, L. Simirenko, C. Luengo, S. Keranen, C. Henriquez, D. Knowles, M. Biggin, M. Eisen, A. DePace. *A Conservered Developmental Patterning Network Produces Quantitatively Different Output in Multiple Species of Drosophila*, PLoS Genetics, 7(10), 2011.

M. Meyer, T. Munzner, A. DePace, H. Pfister. *MulteeSum: A Tool for Comparative Spatial and Temporal Gene Expression Data*, IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), 16(6):908–917, 2010.

M. Meyer, B. Wong, T. Munzner, M. Styczynski, H. Pfister. *Pathline: A Tool for Comparative Functional Genomics*, Computer Graphics Forum (Proceedings of EuroVis), 29(3):1043–1052, 2010.

M. Grabherr, P. Russell, M. Meyer, E. Mauceli, J. Alfoldi, F. DiPalma, K. Lindblad-Toh. *Genome-wide synteny through highly sensitive sequence alignment: Satsuma*, Bioinformatics, 26(9):1145–1151, 2010.

M. Meyer, T. Munzner, H. Pfister. *MizBee: A Multiscale Synteny Browser*, IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis), 15(6):897–904, 2009. **Best Paper Honorable Mention**.

Refereed Conference and Workshop Publications

S. McKenna, D. Staheli, M. Meyer. *Unlocking User-Centered Design Methods for Building Cyber Security Visualizations*, Proceedings of the IEEE Symposium on Visualization for Cyber Security (VizSec), 2015.

N. McCurdy, V. Srikumar, M. Meyer. *RhymeDesign: A Tool for Analyzing Sonic Devices in Poetry*, Workshop on Computational Linguistics for Literature, NAACL HLT, 2015.

A. Bigelow, S. Drucker, D. Fisher, M. Meyer. *Reflections on How Designers Design With Data*, Proceedings of the ACM International Conference on Advanced Visual Interfaces (AVI), 2014. *acceptance rate: 28%.* **Best Paper Award.**

M. Meyer, M. Sedlmair, T. Munzner. *The Four-Level Nested Model Revisited: Blocks and Guidelines*, in Proceedings of the ACM Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV), IEEE VIS 2012.

A. Duchowski, M. Price, M. Meyer, P. Orero. *Aggregate Gaze Visualization with Real-time Heatmaps*, in Proceedings of the ACM Symposium on Eye Tracking Research & Applications (ETRA), 2012. *acceptance rate: 31%*

Book In Progress

M. Meyer and D. Fisher. *Making Sense of Data: Designing Effective Visualizations*, O'Reilly Media, *fall 2016.*

Funding

Current

NIH Grant *PRISMS: Informatics Federation Architecture Center*, K. Sward (PI) *et al*, M. Meyer (Project Lead), NIH (NIBIB), Sept 2015-2019. $5,529,663 ($546,140).

NSF CAREER Grant *Design Decision Patterns for Visualizing Multivariate Graphs*, M. Meyer (PI), NSF Computer Graphics and Visualization (CISE-IIS), July 2014-2019. $400,000 ($400,000).

DARPA Grant *The Visualization Design Environment*, J. Baumes (PI) *et al*, M. Meyer (co-PI), DARPA XDATA, August 2012-2017. $3,038,000 ($550,000).

NIH Grant *Predictive Modeling of Bioelectric Activity on Mammalian Multilayered Neuronal Structures in Presence of Supraphysiological Electric Fields*, G. Lazzi (PI) *et al*, M. Meyer (co-PI), NIH (NIGMS), August 2012-2017. $3,982,276 ($285,089).

NIH Grant *Refining and Testing the Electronic Social Network Assessment Program*, M. Reblin (PI) *et al*, M. Meyer (co-PI), NIH (NCI), September 2015-2016. $250,000 ($19,265).

NSF Large Grant *Modeling, Display, and Understanding Uncertainty in Simulations for Policy Decision Making*, R. Whitaker (PI) *et al*, M. Meyer (co-PI), NSF Computer Graphics and Visualization (CISE-IIS), September 2012-2016. $1,280,000 ($324,095).

Completed

NEH Start-up Grant *Poemage Prototype*, M. Meyer (PI), K. Coles (co-PI), May 2015-2016. $60,000 ($30,000).

University of Utah SEED Grant *The Eye of the Storm: Visualizing Poetry in Space and Time*, K. Coles (PI), M. Meyer (co-PI), August 2013-2014. $28,000 ($14,000).

DoD STTR Grant *SACURE: Situational Awareness for Cyber-secURity Evaluation and training*, R. Pokorny (PI) *et al*, M. Meyer (co-PI), June 2013. $150,000 ($15,000).

State of Utah TCIP Grant *Using Cloud-based Computing as a Channel for Genetics Visualization Software*, M. Meyer (PI), August 2012-2013. $40,000 ($40,000).

Microsoft Research Faculty Fellowship Award July 2012. $200,000 ($200,000).

Keynotes and Distinguished Lectures

Designing Visualizations
International Workshop on Bio-Design Automation Keynote, August 2015
Show and Tell Speaker Series, University of Chicago, May 2015
Strata Conference Keynote, October 2014

Designing Visualizations for Scientists
Rocky Mountain CUWiP Conference Keynote, January 2014

Visualizing Data: Why an (Interactive) Picture is Worth a Thousand Numbers
Park City Institute Lecture Series, August 2013
CI-WATER Symposium Keynote, May 2013
Women's History Month Keynote, Westminster College, March 2013
Gould Distinguished Lecture Series at the University of Utah, September 2012

Designing Visualizations for Biological Research
Design Research Conference Keynote, IIT Institute of Design, October 2012
Arts | Humanities | Complex Networks Keynote – a Leonardo symposium at NetSci, June 2012

Teaching

Introduction to Algorithms and Data Structures (CS 2420), University of Utah, 2015 – 2016

Visualization Seminar (CS 7942), University of Utah, 2012 – 2013, 2015

Visualization (CS 5630/6630), University of Utah, 2012 – 2014

SCI Seminar (CS 7932), University of Utah, 2013 – 2014

Design Studies (CS 7690), University of Utah, 2013

Uncertainty Study Group, University of Utah, 2012 – 2103

Information Visualization (CS 6964), University of Utah, 2012

InfoVis Journal Club, University of Utah, 2011

Visualization (CS 171), Harvard University
Co-instructor and Head Teaching Fellow with Prof. Hanspeter Pfister, 2008 and 2009

Guest Lectures
Undergraduate Research Forum (CS 3020), University of Utah, 2014 – 2015
Business Leadership, University of Utah, 2013
Scientific Computing (CS 3200), University of Utah, 2012 and 2013
Scientific Computing (BIOL/CHEM/PHYS 370), Westminster College, 2013
Introduction to Computer Science (CS 1400), University of Utah, 2012
Data Visualization (CS 171), Harvard University, 2010 and 2011

Students

Current PhD
Alex Bigelow (2012 – 2018)
Ethan Kerzner (2013 – 2017)
Nina McCurdy (2013 – 2018)
Sean McKenna (2012 – 2017)
Jimmy Moore (2015 – 2020)
Sam Quinan (2012 – 2018)

Current Undergraduate
Safia Hassan

Graduated
Alex Bigelow, B.S., "Visualization of Large-Scale Epigenetic Data", 2012
Joshua Dawson, M.S., "Visualizing the UTA Transit System", 2015
Dasha Pruss, B.S., "Toward Interactive Visualization of Connectome Paths", 2016
Zella Urquhart, B.S., 2016

Matthew Might

Professional experience since 2009

- **Executive Office of the President** at The White House. Washington, D.C., U.S.A.
  Strategist. March 2016–Present.
  - Advisor to the President's Precision Medicine Initiative.
  - Member of Office of Management and Budget.
  - Member of United States Digital Service Headquarters.
- **Harvard Medical School**, Department of Biomedical Informatics. Boston, Massachussetts, U.S.A.
  Associate Professor, Visiting. July 2015–Present.
- **University of Utah**, School of Computing. Salt Lake City, Utah, U.S.A.
  Presidential Scholar. July 2014–Present.
  Associate Professor, Tenured. July 2014–Present.
  Assistant Professor, Tenure-Track. Fall 2008–June 2014.

Education

- Ph.D., Computer Science. Fall 2003–Summer 2007.
  Georgia Institute of Technology. Atlanta, Georgia, U.S.A.
  Advisor: Olin Shivers. GPA: 4.00
  Dissertation: *Environment Analysis of Higher-Order Languages*.
  Minor: Economics.
- M.S., Computer Science. Fall 2002–Spring 2003.
  Georgia Institute of Technology. Atlanta, Georgia, U.S.A.
  Specialization: Information security. GPA: 4.00
  (Note: M.S. degree not officially awarded until 2005.)
- B.S., Computer Science. Fall 1999–Fall 2001.
  Georgia Institute of Technology. Atlanta, Georgia, U.S.A.
  Specialization: Systems. Minor: Economics. GPA: 3.94

Publications since 2009

C1. Jason Hemann, William Byrd, Daniel Friedman and Matthew Might. "A Small Embedding of Logic Programming with a Simple Complete Search." *Accepted to Dynamic Languages Symposium 2016.* (**DLS 2016**). October 2016.

C2. Thomas Gilray, Michael Adams and Matthew Might. "Allocation Characterizes Polyvariance." *Accepted to International Conference on Functional Programming* (**ICFP 2016**). September 2016.

C3. Michael Adams, Celeste Hollenbeck and Matthew Might. "On the Complexity and Performance of Parsing with Derivatives." *Proceedings of the 37th Annual Conference of Programming Language Design and Implementation* (**PLDI 2016**). Santa Barbara, California. June 2016.
[16% acceptance rate. 48 accepted. 304 submitted. Celeste is my student.]

C4. James King, Thomas Gilray, Robert M. Kirby, and Matthew Might "Dynamic Sparse-Matrix Allocation on GPUs." *International Supercomputing Conference* (**ISC 2016**). Istanbul, Turkey. June 2016. [42% acceptance rate. 25 accepted. 60 submitted. **Winner of PRACE ISC Best Paper Award.**]

C5. Thomas Gilray, Steven Lyde, Michael D. Adams, Matthew Might and David Van Horn. "Pushdown Control-Flow Analysis for Free." *Symposium on Principles of Programming Languages* (**POPL 2016**). St. Petersburg, Florida. January 2016. pages 691 – 703.
[23% acceptance rate. 59 accepted. 253 submitted. Thomas and Steven are my students.]

C6. David Darais, Matthew Might and David Van Horn. "Galois Transformers and Modular Abstract Interpreters: Reusable Metatheory for Program Analysis." *Conference on Object-Oriented Programming, Systems, Languages and Applications* (**OOPSLA 2015**). Pittsburgh, Pennsylvania. October, 2015. pages 552 – 571.
[25% acceptance rate. 53 accepted. 210 submitted.]

C7. Steven Lyde, William E. Byrd and Matthew Might. "Control-Flow Analysis of Dynamic Languages via Pointer Analysis." *Dynamic Languages Symposium* (**DLS 2015**). Pittsburgh, Pennsylvania. October, 2015. pages 54 – 62.
[35% acceptance rate. 14 accepted. 40 submitted. Steven is my student.]

C8. Peter Aldous and Matthew Might. "Static Analysis of Non-interference in Expressive Low-Level Languages." 22nd International Static Analysis Symposium (**SAS 2015**). Saint-Malo, France. 9 September 2015.
[41% acceptance rate. 18 accepted. 44 submitted. Peter is my student.]

C9. Shuying Liang, Weibin Sun and Matthew Might. "Fast Flow Analysis with Gödel Hashes." 14th IEEE International Working Conference on Source Code Analysis and Manipulation (**SCAM 2014**). Victoria, BC, Canada. 29 September 2014.
[32% acceptance rate. 26 accepted. 82 submitted. **Best Paper Award**. Shuying is my Ph.D. student.]

C10. Shuying Liang, Weibin Sun, Matthew Might, Andrew Keep and David Van Horn. "Pruning, Pushdown Exception-Flow Analysis." 14th IEEE International Working Conference on Source Code Analysis and Manipulation (**SCAM 2014**). Victoria, BC, Canada. 29 September 2014.
[32% acceptance rate. 26 accepted. 82 submitted. Shuying is my Ph.D. student.]

C11. J. Ian Johnson, Nicholas Labich, Matthew Might, David Van Horn. "Optimizing Abstract Abstract Machines." *Proceedings of the International Conference on Functional Programming 2013* (**ICFP 2013**). Boston, Massachusetts. September, 2013.
[30% acceptance rate. 40 accepted. 133 submitted.]

C12. Steven Lyde, Matthew Might. "Extracting Hybrid Automata from Control Code." *Proceedings of the 5th Annual NASA Formal Methods Symposium* (**NFM 2013**). Short paper category. Moffet Field, CA. May, 2013.
[37% (37% short, 37% long) acceptance rate. 37 (9 short, 28 long) accepted. 99 (24 short, 75 long) submitted. Steven Lyde is my Ph.D. student.]

C13. Ilya Sergey, Dominique Devriese, Matthew Might, Jan Midtgaard, David Darais, Dave Clark, Frank Piessens. "Monadic Abstract Interpreters." *Proceedings of the 34th Annual Conference of Programming Language Design and Implementation* (**PLDI 2013**). Seattle, Washington. June, 2013.
[17% acceptance rate. 46 accepted. 267 submitted.]

C14. Christopher Earl, Ilya Sergey, Matthew Might and David Van Horn. "Introspective Pushdown Analysis of Higher-Order Programs." *Proceedings of the International Conference on Functional Programming 2012* (**ICFP 2012**). Copenhagen, Denmark. September, 2012. pages 177–188.
[36% acceptance rate. 32 accepted. 88 submitted. Christopher Earl is my Ph.D. student. Invited for submission to special issue of *Journal of Functional Programming*: **Best Papers of ICFP 2012**.]

C15. Jan Midtgaard, Michael D. Adams and Matthew Might. "A Structural Soundness Proof for Shivers's Escape Technique: A Case for Galois Connections." *Static Analysis Symposium 2012* (**SAS 2012**). Deauville, France. September, 2012. pages 352–369.
[40% acceptance rate. 25 accepted. 62 submitted.]

C16. Michael D. Adams, Andrew W. Keep, Jan Midtgaard, Matthew Might, Arun Chauhan and R. Kent Dybvig. "Flow-Sensitive Type Recovery in Linear-Log Time." *Conference on Object-Oriented Programming, Systems, Languages and Applications* (**OOPSLA 2011**). Portland, Oregon. October, 2011. page 483–498.
[37% acceptance rate. 61 accepted. 165 submitted.]

C17. Matthew Might, David Darais and Daniel Spiewak. “Parsing with Derivatives: A Functional Pearl.” *Proceedings of the 16th ACM International Conference on Functional Programming* (**ICFP 2011**). Tokyo, Japan. September, 2011. pages 189–195.
[34% acceptance rate. 38 accepted. 112 submissions. David Darais was a B.S. student.
Note: The reported acceptance rate for the submission category “Functional Pearl” was 25%.]

C18. Matthew Might and David Van Horn. “A family of abstract interpretations for static analysis of concurrent higher-order programs.” *Proceedings of the 2011 Static Analysis Symposium* (**SAS 2011**). Venice, Italy. September, 2011. pages 180–197.
[32% acceptance rate. 22 accepted. 67 submitted.]

C19. Tarun Prabhu, Shreyas Ramalingam, Matthew Might and Mary Hall. “EigenCFA: Accelerating flow analysis with GPUs.” *Proceedings of the 38th Annual ACM Symposium on the Principles of Programming Languages* (**POPL 2011**). Austin, Texas. January, 2011. pages 511–512.
[23% acceptance rate. 49 accepted. 209 submitted. Tarun Prabhu was my M.S. student.]

C20. David Van Horn and Matthew Might. “Abstracting abstract machines.” *Proceedings of the 15th ACM International Conference on Functional Programming* (**ICFP 2010**). Baltimore, Maryland. September, 2010. pages 51–62.
[33% acceptance rate. 33 accepted. 99 submitted. Invited for submission to special issue of *Journal of Functional Programming*: **Best Papers of ICFP 2010**.]

C21. Matthew Might. “Abstract interpreters for free.” *Proceedings of the 2010 Static Analysis Symposium* (**SAS 2010**). Perpignan, France. September, 2010. pages 407–421.
[30% acceptance rate. 25 accepted. 82 submitted.]

C22. Matthew Might, Yannis Smaragdakis and David Van Horn. “Resolving and exploiting the *k*-CFA paradox: Illuminating functional v. object-oriented program analysis.” *Proceedings of the 31st Annual Conference on Programming Language Design and Implementation* (**PLDI 2010**). Toronto, Canada. June, 2010. pages 305–315.
[20% acceptance rate. 41 accepted. 204 submitted.]

C23. Matthew Might. “Shape analysis in the absence of pointers and structure.” *Proceedings of the 11th Annual Conference on Verification, Model-Checking and Abstract Interpretation* (**VMCAI 2010**). Madrid, Spain. January, 2010. pages 263–278.
[37% acceptance rate. 21 accepted. 57 submitted.]

C24. Matthew Might and Panagiotis Manolios. “*A posteriori* soundness for non-deterministic abstract interpretations.” *Proceedings of the 10th Annual Conference on Verification, Model-Checking and Abstract Interpretation* (**VMCAI 2009**). Savannah, Georgia. January, 2009. pages 1–15.
[33% acceptance rate. 24 accepted. 72 submitted.]

J1. Christopher Earl, Matthew Might, Abhishek Bagusetty and James Sutherland. “An efficient, parallel, and portable domain-specific language for numerically solving partial differential equations.” *Journal of Systems and Software*. pages 1–12. January 2016.

J2. Katherine F. Lambertson, Stephen A. Damiani, Matthew Might, Robert Shelton and Sharon F. Terry. “Participant-Driven Matchmaking in the Genomic Era.” *Journal of Human Mutation*. Volume 36. Issue 10. pages 965–973. October 2015.

J3. Kimball Germane and Matthew Might (2014). “Deletion: The curse of the red-black tree.” *Journal of Functional Programming*, 24(4), pp 423-433. July 2014.

J4. J. Ian Johnson, Ilya Sergey, Christopher Earl, Matthew Might and David Van Horn. “Pushdown flow analysis with abstract garbage collection.” *Journal of Functional Programming*. 24, pp 218-283. May 2014.
[Extended report invited as a Best Paper of ICFP 2012.]

J5. Matthew Might and Matt Wilsey. “The shifting model in clinical diagnostics: how next-generation sequencing and families are altering the way rare diseases are discovered, studied, and treated.” *Genetics in Medicine*. Peer-reviewed commentary. 20 March 2014.

J6. David Van Horn and Matthew Might. "Systematic Abstraction of Abstract Machines." *Journal of Functional Programming*. September 2012. Volume 22. Special Issue 4-5. pages 705–746.
[Extended report invited as a Best Paper of ICFP 2010.]

J7. David Van Horn and Matthew Might. "Abstracting Abstract Machines: A Systematic Approach to Higher-Order Program Analysis." *Communications of the ACM*. 2011. September, 2011. pages 101–109.
[Nominated and selected by CACM to appear as a Research Highlight. Only two research highlights per month are selected across computer science.]

E1. Jan Midtgaard and Matthew Might, Editors. *2012 Proceedings of the Workshop on Numeric and Symbolic Abstract Domains* (NSAD 2012). Electronic Notes in Theoretical Computer Science. Elsevier. Volume 287, pages 1-100. Deauville, France. 10 September 2012.
[77% acceptance rate. 7 papers accepted. 9 papers submitted.]

E2. Matthew Might, Editor. *2011 Proceedings of the Workshop on Scheme and Functional Programming.* Electronic Proceedings. Portland, Oregon. 23 October 2011.
[77% acceptance rate. 7 papers accepted. 9 papers submitted.]

E3. Dixie Baker, Matthew Might, Pearl O'Rourke, Laura Lyman Rodriguez, Tania Simoncelli, John Wilbanks. "Participant Engagement, Data Privacy, and Novel Ways of Returning Information to Participants." Working Group Report for NIH Large Cohort Precision Medicine Workshop. 11 February 2015. Bethesda, Maryland.
[Presented by Pearl O'Rourke at the kick-off NIH Precision Medicine Workshop.]

Teaching since 2009

- Instructor, CS5470: "Compilers." 60 students. Spring 2015.
- Instructor, CS6475: "Advanced topics in compilation." 21 students. Fall 2013.
- Instructor, CS5470: "Compilers." 53 students. Spring 2013.
- Instructor, CS5959: "Scripting language design and implementation." 14 students. Spring 2012.
- Instructor, CS7938: "Static analysis seminar." 7 students. Spring 2012.
- Instructor, CS5470: "Compilers." 35 students. Spring 2011.
- Instructor, CS7938: "Static analysis seminar." 7 students. Spring 2011.
- Instructor, CS7938: "Static analysis seminar." 3 students. Spring 2010.
- Instructor, CS6470: "Advanced topics in compilation." 15 students. Fall 2009.
- Instructor, CS6969: "Programming language analysis." 15 students. Spring 2009.
- Instructor, CS7938: "Static analysis seminar." 4 students. Spring 2009.

Awards

A1. Quora Top Writer 2016. Quora. January 2016.
A2. Presidential Scholar. University of Utah. July 2014.
A3. Top 15% of College Teaching (Undergraduate Lecturing). AY 2012-2013.
A4. Outstanding Instructor Award, School of Computing. AY 2012-2013.
A5. Top 15% of College Teaching (Undergraduate Lecturing). AY 2011-2012.
A6. Nominee for ACM SIGPLAN Dissertation Award. 2007.
A7. Nominee for ACM Doctoral Dissertation Award. 2007.
A8. Outstanding Doctoral Dissertation, Computing, Georgia Tech. 2007.

Chris John Myers

Department of Electrical and Computer Engineering, University of Utah
50 S. Central Campus Dr. Rm. 4112, Salt Lake City, UT 84112-9206
(801) 581-6490, *myers@ece.utah.edu*, *http://www.async.ece.utah.edu/∼myers*

Education

Stanford University, Stanford, California (1991-1995)
Ph.D. degree in Electrical Engineering (1995)
Thesis: Computer-Aided Synthesis and Verification of Gate-Level Timed Circuits
M.S. degree in Electrical Engineering (1993)

California Institute of Technology, Pasadena, California (1987-1991)
B.S. degree with honor in Electrical Engineering and History (1991)

Professional Experience (Selected)

Professor of Electrical and Computer Engineering, *University of Utah* (2006-present)
Adjunct Professor of Computer Science, *University of Utah* (2007-present)
Adjunct Professor of Bioengineering, *University of Utah* (2007-present)

Awards (Selected)

Fellow of the IEEE (2013)
ECE Departmental Service Award, *University of Utah* (2013)

Service (Selected)

Computer Engineering Committee Member, *University of Utah* (1995-present)

Recruiting Committee Member, *University of Utah* (1995-1998,1999-2000, 2015-present)

Member of the IEEE, S'91-M'96-SM'04-Fellow'13 (1991-present)

Member of the ACM (1996-present)

Technology Editor, *ACS Synthetic Biology* (2017-present)

Member of the Editorial Board, *Engineering Biology* (2016-present)

Member of the Editorial Board, *Synthetic Biology* (2016-present)

Member of the Steering Committee, *Synthetic Biology Open Language* (2015-present)

Member of the Steering Committee, *Synthetic Biology Standards Consortium* (2015-present)

Guest Editor, *IEEE Design & Test Magazine* (2015-present)

Organizer, *COMBINE Forum 2015* (2015)

COMBINE (COmputational Modeling in Biology NEtworks) Coordinator (2014-present)

Associate Editor, *IEEE Life Sciences Letters*, (2014-present)

Member of the Steering Committee, *Frontiers in Analog CAD Workshop* (2010-present)

Courses Developed (Selected)

1. Modeling and Analysis of Biological Networks - In 2009, I published the textbook, *Engineering Genetic Circuits*, used for this course.
2. Asynchronous Circuit Design - In 2001, I published the textbook, *Asynchronous Circuit Design*, used for this course.
3. Computer Aided Design of Digital Circuits - This course provides an introduction to algorithms for the synthesis and optimization of digital designs.
4. Embedded System Design - I completely redesigned this course to focus on embedded system design issues rather than just interfacing with a PC.
5. Formal Verification - This course presents state-of-the-art methods for the formal verification of hardware and software systems. A new version of this class was developed in 2012.

Publications (Selected)

Books

1. C. J. Myers, *Engineering Genetic Circuits*, Chapman & Hall/CRC Press, July, 2009.
2. C. J. Myers, *Asynchronous Circuit Design*, John Wiley and Sons, July, 2001.

Book Chapters and Books Edited

1. C. Myers, K. Clancy, G. Misirli, E. Oberortner, M. Pocock, J. Quinn, N. Roehner, and H. Sauro, "The Synthetic Biology Open Language", in *Computational Methods in Synthetic Biology*, Methods in Molecular Biology, Volume 1244, pages 323-336, 2015.
2. C. Madsen, C. Myers, N. Roehner, C. Winstead, and Z. Zhang, "Efficient Analysis Methods in Synthetic Biology", in *Computational Methods in Synthetic Biology*, Methods in Molecular Biology, Volume 1244, pages 217-257, 2015.
3. A. Fisher, D. Kulkarni, and C. Myers, "A new assertion property language for analog/mixed-signal circuits", in *Languages, Design Methods, and Tools for Electronic System Design - Selected Contributions from FDL 2013*, Lecture Notes in Elec. Eng., Vol. 311, pages 45-65, 2015.
4. C. Myers, "Platforms for Genetic Design Automation", in *Methods in Microbiology 2013: Microbial Synthetic Biology*, November, 2013.

Journal Articles

1. N. Roehner, J. Beal, K. Clancy, B. Bartley, G. Misirli, R. Grunberg, E. Oberortner, M. Pocock, M. Bissell, C. Madsen, T. Nguyen, M. Zhang, Z. Zhang, Z. Zundel, D. Densmore, J. Gennari, A. Wipat, H. Sauro, and C. Myers, "Sharing structure and function in biological design with SBOL 2.0", to appear in *ACS Synthetic Biology.*
2. D. Waltemath, J.. Karr, F. Bergmann, V. Chelliah, M. Hucka, M. Krantz, W. Liebermeister, P. Mendes, C. Myers, P. Pir, B. Alaybeyoglu, N. Aranganathan, K. Baghalian, A. Bittig, P. Burke, M. Cantarelli, Y. Chew, R. Costa, J. Cursons, T. Czauderna, A. Goldberg, H. Gómez, J. Hahn, T. Hameri, D. Kazakiewicz, I. Kiselev, V. Knight-Schrijver, C. Knüpfer, M. König, D. Lee, A. Lloret-Villas, N. Mandrik, J. Medley, B. Moreau, H. Meshkin, S. Palaniappan, D. Priego-Espinosa, M. Scharm, M. Sharma, K. Smallbone, N. Stanford, J. Song, T. Theile, M. Tokic, N. Tomar, V. Touré, J. Uhlendorf, T. Varusai, L. Watanabe, F. Wendland, M. Wolfien, J. Yurkovich, Y. Zhu, A. Zardilis, A. Zhukova, and F. Schreiber, "Toward community standards and software for whole-cell modeling", to appear in *IEEE Trans. on Bio. Eng.*.
3. V. Dubikhin, D. Sokolov, C. Myers, and A. Yakovlev, "Design of Mixed-signal Systems with Asynchronous Control", to appear in *IEEE Design and Test.*

4. Z. Zhang, T. Nguyen, N. Roehner, G. Misirli, M. Pocock, E. Oberortner, M. Samineni, Z. Zundel, J. Beal, K. Clancy, A. Wipat, C. Myers, "libSBOLj 2.0: A Java Library to Support SBOL 2.0", to appear in *IEEE Life Sciences Letters*.

5. L. Watanabe and C. Myers, "Efficient Analysis of SBML Models of Cellular Populations Using Arrays", to appear in *ACS Synthetic Biology*.

6. T. Nguyen, N. Roehner, Z. Zundel, and C. Myers, "A Converter from the Systems Biology Markup Language to the Synthetic Biology Open Language", to appear in *ACS Syn. Bio.*.

7. Z. Zheng, W. Serwe, J. Wu, T. Yoneda, H. Zheng, and C. Myers, "An Improved Fault-Tolerant Routing Algorithm for a Network-on-Chip Derived with Formal Analysis", to appear in *Science of Computer Programming*.

8. C. Myers, "Computational Synthetic Biology: Progress and the Road Ahead", in *IEEE Transactions on Multi-scale Computing Systems*, 1(1): 19-32, 2015.

9. J. Quinn, R. Cox, A. Adler, J. Beal, S. Bhatia, Y. Cai, J. Chen, K. Clancy, M. Galdzicki, N. Hillson, N. Le Novère, A. Maheshwari, J. Alastair, C. Myers, Umesh P, M. Pocock, C. Rodriguez, L. Soldatova, G.-B. Stan, N. Swainston, A. Wipat, and H. Sauro, "SBOL Visual: A Graphical Language for Genetic Designs", in *PLOS Biology*, 13(12): e1002310, 2015.

10. N. Rodriguez, A. Thomas, L. Watanabe, I. Vazirabad, V. Kofia, H. Gómez, F. Mittag, J. Rudolph, F. Wrzodek, E. Netz, A. Diamantikos, J. Eichner, R. Keller, C. Wrzodek, S. Fröhlich, N. Lewis, C. Myers, N. Le Novère B. Palsson, M. Hucka, and A. Dräger, "JSBML 1.0: providing a smorgasbord of options to encode systems biology models", in *Bioinfo.*, 31(20):3383-6, 2015.

11. F. Schreiber, G. Bader, M. Golebiewski, M. Hucka, B. Kornmeier, N. Le Novère, C. Myers, D. Nickerson, B. Sommer, D. Waltemath and S. Weise, "Specifications of Standards in Systems and Synthetic Biology", *Journal of Integrative Bioinformatics*, 12(2):258, 2015.

12. L. Smith, M. Hucka, S. Hoops, A. Finney, M. Ginkel, C. Myers, I. Moraru and W. Liebermeister, "SBML Level 3 package: Hierarchical Model Composition, Version 1 Release 3", *Journal of Integrative Bioinformatics*, 12(2):268, 2015.

13. M. Hucka, F. Bergmann, A. Dräger, S. Hoops, S. Keating, N. Le Novère, C. Myers, B. Olivier, S. Sahle, J. Schaff, L. Smith, D. Waltemath and D. Wilkinson. "Systems Biology Markup Language (SBML) Level 2 Version 5: Structures and Facilities for Model Definitions", *Journal of Integrative Bioinformatics*, 12(2):271, 2015.

14. B. Bartley, J. Beal, K. Clancy, G. Misirli, N. Roehner, E. Oberortner, M. Pocock, M. Bissell, C. Madsen, T. Nguyen, Z. Zhang, J. Gennari, C. Myers, A. Wipat and H. Sauro. "Synthetic Biology Open Language (SBOL) Version 2.0.0", *Journal of Integrative Bioinformatics*, 12(2):272, 2015.

15. N. Roehner, Z. Zhang, T. Nguyen, C. Myers, "Generating Systems Biology Markup Language Models from the Synthetic Biology Open Language", in *ACS Synthetic Biology*, 4(8), 867-943, August 21, 2015.

16. M. Hucka, D. Nickerson, G. Bader, F. Bergmann, J. Cooper, E. Demir, A. Garny, M. Golebiewski, C. Myers, F. Schreiber, D. Waltemath, N. Le Novère, "Promoting coordinated development of community-based information standards for modeling in biology: the COMBINE initiative", in *Frontiers in Bioengineering and Biotechnology*, 3(19), 2015.

17. H. Zheng, Z. Zhang, C. Myers, E. Rodriguez, and Y. Zhang, "Compositional Model Checking of Concurrent Systems", in *IEEE Transactions on Computers*, 64(6), June, 2015.

18. N. Roehner, E Oberortner, M. Pocock, J. Beal, K. Clancy, C. Madsen, G. Misirli, A. Wipat, H. Sauro, C. Myers, "A Proposed Data Model for the Next Version of the Synthetic Biology

Open Language", in *ACS Synthetic Biology*, 4(1), 57-71, January 16, 2015.

19. C. Myers, H. Sauro, and A. Wipat, "Introduction to the Special Issue on Computational Synthetic Biology", in *ACM Journal on Emerging Technologies in Computing Systems*, 11(3), December, 2014.
20. C. Madsen, Z. Zhang, N. Roehner, C. Winstead, and C. Myers, "Stochastic Model Checking of Genetic Circuits", in *ACM Journal on Emerging Technologies in Computing Systems*, 11(3), December, 2014.
21. L. Watanabe and C. Myers, "Hierarchical Stochastic Simulation Algorithm for SBML Models of Genetic Circuits", in *Frontiers in Bioengineering and Biotechnology*, 2(55), 2014.
22. N. Roehner and C. J. Myers, "Directed Acyclic Graph-Based Technology Mapping of Genetic Circuit Models.", in *ACS Synthetic Biology*, 3(8), 543-555, August 15, 2014.
23. M. Galdzicki, K. Clancy, E. Oberortner, M. Pocock, J. Quinn, C. Rodriguez, N. Roehner, M. Wilson, L. Adam, J. C. Anderson, B. Bartley, J. Beal, D. Chandran, J. Chen, D. Densmore, D. Endy, R. GruÌĹnberg, J. Hallinan, N. Hillson, J. Johnson, A. Kuchinsky, M. Lux, G. Misirli, J. Peccoud, H. Plahar, E. Sirin, G.-B. Stan, A. Villalobos, A. Wipat, J. Gennari, C. Myers, H. Sauro, "SBOL: A community standard for communicating designs in synthetic biology", in *Nature Biotechnology*, 32(6): 545-550, June, 2014.
24. D. Waltemath, F. Bergmann, C. Chaouiya, T. Czauderna, P. Gleeson, C. Goble, M. Golebiewski, M. Hucka, N. Juty, O. Krebs, N. Le Novère, H. Mi, I. Moraru, C. Myers, D. Nickerson, B. Olivier, N. Rodriguez, F. Schreiber, L. Smith, F. Zhang, and E. Bonnet, "Meeting report from the fourth meeting of the Computational Modeling in Biology Network (COMBINE)", in *Standards in Genomic Sciences*, 9(3), 2014.
25. N. Roehner and C. J. Myers, "A Methodology to Annotate Systems Biology Markup Language Models with the Synthetic Biology Open Language", in *ACS Synthetic Biology*, 3(2): 57-66, Feb. 21, 2014.
26. J. Stevens and C. Myers, "Dynamic Modeling of Cellular Populations within iBioSim," in *ACS Synthetic Biology*, 2(5): 223-229, Nov. 21, 2012.
27. C. Madsen, C. Myers, T. Patterson, N. Roehner, J. Stevens, and C. Winstead, "Design and test of genetic circuits using iBioSim," in *IEEE Design and Test*, 29(3): 32-39, June, 2012.
28. S. Little, D. Walter, C. Myers, R. Thacker, S. Batchu, and T. Yoneda, "Verification of analog/ mixed-signal circuits using labeled hybrid Petri nets," in *IEEE Transactions on CAD*, 30(4): 617-630, April, 2011.
29. N. Barker, C. Myers, and H. Kuwahara, "Learning genetic regulatory network connectivity from time series data," in *IEEE Transactions on Computational Biology and Bioinformatics*, 8(1):152-165, Jan-Mar, 2011.
30. N. Nguyen, C. Myers, H. Kuwahara, C. Winstead, and J. Keener, "Design and analysis of a robust genetic Muller C-element," in *Journal of Theor. Bio.*, 264(2): 174-187, May, 2010.
31. S. Little, D. Walter, K. Jones, C. Myers, and A. Sen, "Analog/mixed-signal circuit verification using models generated from simulation traces," in *The International Journal of Foundations of Computer Science*, 21(2): 191-210, 2010.
32. H. Kuwahara, C. Myers, and M. Samoilov, "Temperature control of fimbriation circuit switch in uropathogenic *escherichia coli*: quantitative analysis via automated model abstraction," in *PLoS Computational Biology*, 6(3): e1000723, March 2010.

Erin Parker

Associate Professor (Lecturer)
School of Computing
University of Utah

50 S. Central Campus Drive
Room 3190 MEB
Salt Lake City, UT 84112

parker@cs.utah.edu

www.cs.utah.edu/∼parker

Experience

2015-present	Associate Professor, Lecturer School of Computing, University of Utah
2008-2015	Assistant Professor, Lecturer School of Computing, University of Utah
2005-2008	Assistant Professor, Adjunct School of Computing, University of Utah
2005	Instructor School of Computing, University of Utah
2002	Instructor Department of Computer Science, University of North Carolina at Chapel Hill

Teaching — Over 3700 students in sixteen courses taught fifty times.

CS 1050: Computers in Society
Sp08

CS 1060: Explorations in Computer Science
Fa08, Sp09, Fa09

CS/EAE 1400: Intro to Computer Science†
Fa10, Sp11, Fa11, Sp12, Su12, Fa12, Sp13, Fa13, Sp14

CS 1410: Intro to Object-Oriented Programming
Fa13, Sp14, Fa15, Sp16

CS 2000: Intro to Program Design in C
Fa05, Fa06, Fa07

CS 2420: Intro to Data Structures & Algorithms
Su06, Su07, Su08, Sp10, Sp11, Sp12, Fa15

CS 3020: Research Forum†
Fa09, Fa10, Fa11, Fa12, Fa13

CS 3500: Software Practice I
Sp06

† denotes courses created

CS/ECE 3810: Computer Organization
Sp16

CS 4000: Senior Capstone Design
Fa12

CS 4400: Computer Systems
Fa07, Fa08, Fa09, Fa10, Fa11

CS 4500: Senior Capstone Project
Sp13

CS 5040: Teaching Introductory Computer Science
Fa15

CS 5470: Compiler Principles & Techniques
Sp05, Sp06, Sp07, Sp08, Sp09, Sp10

CS 5510: Programming Language Concepts
Fa06

COMP 014: Intro to Programming (UNC)
Su02

Supervising — Guidance, mentorship, and evaluation of graduate student instructors.

Summer 2013-14	CS 2420, Paymon Saebi
Spring 2013	CS 2420, Daniel Kopta

Summer 2012	CS 2420, Stephen Ward
Summer 2011	CS 1400, Jake Van Alstyne and Dan Maljovec; CS 2420, Daniel Kopta

Service

School of Computing

2008-present Diversity Committee member

Work to diversify the School of Computing primarily through recruitment and retention of women and underrepresented minority students. Specific duties include maintaining mailing lists of women undergraduate students, graduate students, and faculty; facilitating and selecting student representatives for diversity conferences; and semesterly meetings with undergraduate women students.

2010-present Undergraduate Studies Committee member

Review and improve the undergraduate computer science curriculum (particularly focused on the courses of the first, "pre-major" year).

2011-present Scholarship Committee member (2011-2012) and chair (2013-present)

Organize SoC scholarship applications for review by committee members. Match high-ranking applicants to awards for which they are eligible. Coordinate with CoE and ECE, for which the pool of applications overlaps with SoC.

2011-present Undergraduate Committee member

Organize and administer admission to full-major status. Decide on probationary actions for students who do not meet the minimum requirements for continuing performance. Approve BS Theses. Decide on exceptions to degree requirements and determine equivalency of courses completed elsewhere. Manage the CS 1410 Proficiency Test, by which students may waive the CS/EAE 1030 (formerly 1400) pre-major requirement. Advise students academically.

2016-present ACM's Committee on Women in Computing (ACM-W) faculty advisor

Advise the members of University of Utah's ACM-W student chapter, particularly the chapter leaders, through attending meetings, handling logistics, and recruiting new members.

2010-2015 CS Undergraduate Student Advisory Council (UgSAC) faculty advisor

Supervised a small group of students committed to enhancing the community of CS undergraduates by organizing social activities, professional development workshops, and collecting valuable feedback on all aspects of the student experience in the School of Computing.

2008-2011	Education Outreach coordinator Created and led outreach programs designed to expose K-12 students to the exciting field of computer science.

College of Engineering

2011-present	College of Engineering Scholarship Committee member Review and rate hundreds of applications for scholarships awarded by the College.
2011-present	Society of Women Engineers (SWE) faculty advisor Advise the members of University of Utah's SWE section, particularly the section leaders, through attending regular meetings (leadership and general member), participating in events (K-12 outreach, professional development, and social), and attending SWE conferences (national and regional).
2012-present	Women in Engineering (WIE) faculty mentor Mentor the student members of WIE and help coordinate WIE events.
2009-2014	Hi-GEAR (Girls Engineering Abilities Realized) Camp Created and often presented the computer science activity in this week-long camp that exposes high-school girls to all kinds of engineering.

University of Utah

2014-present	Honors Faculty Advisor Advise computer science students on the Honors track and connect students to faculty in the department for supervising their Honors thesis work.

External

2014-present	National Center for Women & Information Technology (NCWIT) Aspriations in Computing award presenter and application reviewer
2009-present	Utah Computer Science Teachers Association (CSTA) member
2014	Underrepresented Women in Computing (UWiC) mentor at the Grace Hopper Celebration of Women in Computing
2009-14	Expanding Your Horizons presenter (annual conference for middle-school girls interested in math and science)
2013	Scratch Day presenter (annual workshop for elementary- and middle-school students interested in computer science)

2010	NSF Transforming Undergraduate Education in Science Program (TUES) reviewer
2008	Principles and Practice of Parallel Programming (PPoPP) local arrangements chair

Awards

2011	College of Engineering Outstanding Teaching Award
2011, 2006	School of Computing Outstanding Teaching Award
2006-present	Dean's letters for excellent teaching

Education

2004	Ph.D. in Computer Science University of North Carolina at Chapel Hill *Analyzing the Behavior of Loop Nests in the Memory Hierarchy: Methods, Tools, and Applications*
2001	M.S. in Computer Science University of North Carolina at Chapel Hill.
1999	B.S. in Computer Science and Mathematics The College of William and Mary, Williamsburg, Virginia.

Publications

Philip J. Hanlon, Dean Chung, Siddhartha Chatterjee, Daniela Genius, Alvin R. Lebeck, and Erin Parker. "The Combinatorics of Cache Misses during Matrix Multiplication." *Journal of Computer and System Sciences*. 2001.

Erin Parker and Siddhartha Chatterjee. "An Automata-Theoretic Algorithm for Counting Solutions to Presburger Formulas." In *Proceedings of CC 2004 International Conference on Compiler Construction*, Barcelona, Spain. April, 2004. 32% acceptance rate.

Siddhartha Chatterjee, Erin Parker, Philip J. Hanlon, and Alvin R. Lebeck. "Exact Analysis of Cache Misses in Nested Loops." In *Proceedings of the International Symposium on Programming Language Design and Implementation (PLDI)*, Snowbird, Utah. June, 2001. 21% acceptance rate.

References

Prof. Joe Zachary, School of Computing (zachary@cs.utah.edu)

Prof. Jim de St. Germain, School of Computing (germain@cs.utah.edu)

Biographical Sketch
Valerio Pascucci

Center for Extreme Data Management Analysis and Visualization
Scientific Computing and Imaging (SCI)
University of Utah
72 So. Central Campus Drive
Salt Lake City, Utah 84103

Voice: 801-587-9885
Fax: 801-585-6513
E-mail: pascucci@sci.utah.edu
Web: http://CEDMAV.COM

Professional Preparation

- "Laurea" degree in Electrical Engineering, University of Rome, Italy, 1993.
- Qualification for the membership to the Italian Order of Engineers, 1995.
- PhD Computer Science, Purdue University, West Lafayette, IN, USA, 2000.

Appointments

- 2011- Present Director, Center for Extreme Data Management Analysis and Visualization (CEDMAV), University of Utah
- 2013- Present Director, Data Center Engineering Certificate, University of Utah
- 2011- Present DoE Laboratory Fellow, Pacific Northwest National Laboratory, WA
- 2011- Present Professor, School of Computing, University of Utah.
- 2011- Present Founder and CEO, ViSUS LLC.
- 2014- Present Technical Adviser, nView medical.
- 2008- Present Faculty, Scientific computing and Imaging Institute, University of Utah.
- 2015- Present Associate Editor, IEEE Transactions on Visualization & Computer Graphics (also 2006-2010)
- 2014-2015 Visiting Prof., King Abdullah Univ. of Sci. and Technology (KAUST).
- 2011-2013 Associate Director, Scientific Computing and Imaging Inst., Univ. of Utah
- 2008-2011 Associate Professor, School of Computing, University of Utah.
- 2000-2008 Computer Scientist, Project Leader, Group Leader, CASC, LLNL.
- 2005-2009 Adjunct Assistant Professor, Computer Science, Davis.

Publications

Related Publications

1. S. Kumar, C. Christensen, P.-T. Bremer, E. Brugger, V. Pascucci, J. Schmidt, M. Berzins, H. Kolla, J. Chen, V. Vishwanath, P. Carns, R. Grout. "Fast Multi-Resolution Reads of Massive Simulation Datasets," In Proceedings of the International Supercomputing Conference ISC'14, Leipzig, Germany, June, 2014.
2. S. Kumar, J. Edwards, P.-T. Bremer, A. Knoll, C. Christensen, V. Vishwanath, P. Carns, J.A. Schmidt, V. Pascucci. "Efficient I/O and storage of adaptive-resolution data," In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE Press, pp. 413--423. 2014.

3. S. Kumar, A. Saha, V. Vishwanath, P. Carns, J.A. Schmidt, G. Scorzelli, H. Kolla, R. Grout, R. Latham, R. Ross, M.E. Papka, J. Chen, V. Pascucci. “Characterization and modeling of PIDX parallel I/O for performance optimization,” In Proc. of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 67. 2013.
4. A. G. Landge, V. Pascucci, A. Gyulassy, J. C. Bennett, H. Kolla, J. Chen, and P.-T. Bremer. In- situ feature extraction of large scale combustion simulations using segmented merge trees. In Proc. of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '14, pp.1020-1031, NJ, USA, 2014. IEEE Press.
5. W. Widanagamaachchi, P.-T. Bremer, C. Sewell, L.-T. Lo, J. Ahrens, and V. Pascucci. Data- parallel halo finding with variable linking lengths. In Large Data Analysis and Visualization (LDAV), 2014, IEEE 4th Symposium on, pp. 27-34, Nov. 2014.

Other Significant Publications

1. S. Liu, B. Wang, J. J. Thiagarajan, P.-T. Bremer, and V. Pascucci. Multivariate volume visualization through dynamic projections. In Large Data Analysis and Visualization (LDAV), 2014, IEEE 4th Symposium on, pages 35-42, Nov 2014.
2. V. Pascucci, G. Scorzelli, B. Summa, P.-T. Bremer, A. Gyulassy, C. Christensen, S. Philip, S. Kumar. “The ViSUS Visualization Framework,” In High Performance Visualization: Enabling Extreme-Scale Scientific Insight, Chapman and Hall/CRC Comp. Science, Ch. 19, Edited by E. Wes Bethel and Hank Childs (LBNL) and Charles Hansen (UofU), 2012.
3. V. Pascucci, P.-T. Bremer, A. Gyulassy, G. Scorzelli, C. Christensen, B. Summa, S. Kumar. “Scalable Visualization and Interactive Analysis Using Massive Data Streams,” In Cloud Computing and Big Data, Advan. in Parallel Comp., Vol. 23, IOS Press, pp. 212--230. 2013.
4. S. Kumar, B. Summa, G. Scorzelli, V. Pascucci, V. Vishwanath, P. Carns, R. Ross, J. Chen, H. Kolla, and R. Grout. Pidx: Efficient parallel I/O for multi-resolution multi-dimensional scientific datasets. In IEEE Cluster 2011, Austin Texas, 2011.

Synergistic Activities

- Organizer of Dagstuhl Perspective Workshop on Connecting Performance Analysis and Visualization to Advance Extreme Scale Computing
- Member of the IEEE 3333-2WG - Standardization of 3D Based Medical Application Working group. Chair of the Subcommittee of the IEEE 3333-2 WG - P3333.2.3 - Standard for Three-Dimensional (3D) Medical Data Management.
- Organizer and Program Co-Chair of TopoInVis 2013 and 2009. Co-editor of the book that followed the meetings: ”Topological Methods in Data Analysis and Visualization: Theory, Algorithms, and Applications”, Math+Visualization book series, Springer, 2010.
- Program Committee member for over 60 international conferences

Jeff M. Phillips

Assistant Professor | School of Computing | University of Utah
Director, Data Management and Analysis Track | Big Data Program
Co-Organizer, Data Group
Center for Extreme Data Management, Analysis, and Visualization
50 S Central Campus Dr., Salt Lake City, UT 84112 | (801) 585-7775
`http://www.cs.utah.edu/~jeffp | jeffp@cs.utah.edu`

Education

Duke University

Ph.D. in Computer Science, January, 2009.
Thesis Title: *Small and Stable Descriptors of Distributions for Geometric Statistical Problems.*
Advisor: Pankaj K. Agarwal.

Rice University

Bachelor of Science in Computer Science, May 2003.
Bachelor of Arts in Mathematics, May 2003.

Research Experience

University of Utah, School of Computing (Assistant Professor)	(2011-present)
University of Utah, School of Computing (Postdoctoral CI Fellow)	(2009-2011)
Duke University, Department of Computer Science (Postdoctoral Associate)	(2009)
Duke University, Department of Computer Science (Research Assistant)	(2003-2009)
Yahoo! Research (Research Intern)	(Summer 2007)
AT&T Research (Visiting Researcher)	(Summer/Winter 2005)
Rice University, Department of Computer Science (Research Assistant)	(2000-2003)
The Charles Stark Draper Laboratory, Inc. (Research Scientist)	(2002-2003)

Fellowships and Awards

NSF CAREER Award. (2014)

Best Paper Award at MultiClust Workshop: Discovering, Summarizing and Using Multiple Clusterings. (2011)

CCC-CRA-NSF Computing Innovation Fellowship. (2009)
2 year postdoctoral fellowship — 60 awarded among all graduating computer scientists in US

Best Student Paper at International Conference on Automata, Languages and Programming (ICALP). (2008)

Distinguished Department Service Award (Duke Computer Science). (2008)
For 5 years of department service — never before awarded

Outstanding Department Service Award (Duke Computer Science). (2006)

NSF Graduate Research Fellowship. (2004-2007)
3 year full graduate fellowship

James B. Duke Fellowship. (2003-2007)

Book Chapters and Surveys

[S1] Coresets and Sketches (to appear).
Jeff M. Phillips.
Handbook of Discrete and Computational Geometry, 3rd edition, CRC Press, Chapter 49. 2016.

[S2] A Gentle Introduction to the Kernel Distance.
Jeff M. Phillips and Suresh Venkatasubramanian.
arXiv:1103.1625, March 2011.

Conference and Journal Publications (since 2011)

[C1] Streaming Principal Component Analysis.
Mina Ghashami and Daniel Perry, and Jeff M. Phillips.
International Conference on Artificial Intelligence and Statistics (AISTATS), [31%] May 2016.

[C2] Subsampling in Smooth Range Spaces.
Jeff M. Phillips and Yan Zheng.
Algorithmic Learning Theory (ALT), [50%] October 2015.

[C3] L_∞ Error and Bandwidth Selection for Kernel Density Estimates of Large Data.
Jeff M. Phillips and Yan Zheng.
ACM Conference on Knowledge Discovery and Data Mining (KDD), [19%] August 2015.

[C4] Geometric Inference on Kernel Density Estimates.
Jeff M. Phillips, Bei Wang, and Yan Zheng.
International Symposium on Computational Geometry (SoCG), [38%] June 2015.

[C5] Improved Practical Matrix Sketching with Guarantees.
Mina Ghashami, Amey Desai, and Jeff M. Phillips.
22nd Annual European Symposium on Algorithms (ESA), [25%] September 2014.
Transactions on Knowledge and Data Engineering (TKDE) (to appear) 2016.

[C6] Continuous Matrix Approximation on Distributed Data.
Mina Ghashami, Jeff M. Phillips, and Feifei Li.
40th International Conference on Very Large Data Bases (VLDB), [22%] September 2014.

[C7] Relative Errors for Deterministic Low-Rank Matrix Approximations.
Mina Ghashami and Jeff M. Phillips.
25th Annual ACM-SIAM Symposium on Discrete Algorithms (SoDA), [28%] January 2014.
extended version as Frequent Directions: Simple and Deterministic Matrix Sketching.
Mina Ghashami, Edo Liberty, Jeff M. Phillips and David P. Woodruff.

[C8] Quality and Efficiency for Kernel Density Estimates in Large Data.
Yan Zheng, Jeffrey Jestes, Jeff M. Phillips, and Feifei Li.
ACM Conference on Management of Data (SIGMOD), [20%] June,2013.

[C9] Nearest Neighbor Searching Under Uncertainty II.
Pankaj K. Agarwal, Boris Aronov, Sariel Har-Peled, Jeff M. Phillips, Ke Yi, Wuzhou Zhang.
32nd ACM Symposium on Principles of Database Systems (PoDS), [25%] June 2013.

[C10] Range Counting Coresets for Uncertain Data.
Amirali Abdullah, Samira Daruki, Jeff M. Phillips.
29th Annual ACM Symposium on Computational Geometry (SoCG), [35%] June 2013.

[C11] Radio Tomographic Imaging and Tracking of Stationary and Moving People via Kernel Distance.
Yang Zhao, Neal Patwari, Jeff M. Phillips, and Suresh Venkatasubramanian.
12th ACM-IEEE Conference on Information Processing in Sensor Networks (IPSN), [21%] April 2013.

[C12] ε-Samples for Kernels.
Jeff M. Phillips.
24th Annual ACM-SIAM Symposium on Discrete Algorithms (SoDA), [30%] January 2013.

[C13] Efficient Protocols for Distributed Classification and Optimization.
Hal Daume III, Jeff M. Phillips, Avishek Saha, and Suresh Venkatasubramanian.
23rd International Conference on Algorithmic Learning Theory (ALT), [49%] October 2012.

[C14] Ranking Large Temporal Data.
Jeffrey Jestes, Jeff M. Phillips, Feifei Li, and Mingwang Tang.
38st International Conference on Very Large Databases (VLDB), [20%] August 2012.
PVLDB 5:1412-1423, 2012.

[C15] Mergeable Summaries.
Pankaj K. Agarwal, Graham Cormode, Zengfeng Huang, Jeff M. Phillips, Zhewei Wei, and Ke Yi.
31st ACM Symposium on Principals of Database Systems (PODS), [26%] May 2012.
invited to *ACM Transactions on Database Systems (TODS)* 38:26, 2013.

[C16] Protocols for Learning Classifiers on Distributed Data.
Hal Daume III, Jeff M. Phillips, Avishek Saha, and Suresh Venkatasubramanian.
15th International Conference on Artificial Intelligence and Statistics (AISTATS), [30%] April 2012.

[C17] Efficient Threshold Monitoring for Distributed Probabilistic Data.
Mingwang Tang, Feifei Li, Jeff M. Phillips, and Jeffrey Jestes.
28th IEEE International Conference on Data Engineering (ICDE), [24%] April 2012.

[C18] Uncertainty Visualization in HARDI based on Ensembles of ODFs.
Fangxiang Jiao, Jeff M. Phillips, Yaniv Gur, and Chris R. Johnson.
5th IEEE Pacific Visualization Symposium (PacificVis), [34%] February 2012.

[C19] Lower Bounds for Number-in-Hand Multiparty Communication Complexity, Made Easy.
Jeff M. Phillips, Elad Verbin, and Qin Zhang.
23th Annual ACM-SIAM Symposium on Discrete Algorithms (SoDA), [31%] January 2012.
SIAM Journal of Computing (SICOMP) (to appear) 2015.

[C20] Geometric Computation on Indecisive Points.
Allan G. Jørgensen, Maarten Löffler, and Jeff M. Phillips.
Algorithms and Data Structures Symposium (WADS), [42%] August 2011.

[C21] Computing Hulls, Centerpoints, and VC-Dimension in Positive Definite Space.
P. Thomas Fletcher, John Moeller, Jeff M. Phillips, and Suresh Venkatasubramanain.
Algorithms and Data Structures Symposium (WADS), [42%] August 2011.

[C22] Comparing Distributions and Shapes Using the Kernel Distance.
Sarang Joshi, Raj Varma Kommaraju, Jeff M. Phillips, and Suresh Venkatasubramanain.
ACM Symposium on Computational Geometry (SoCG), [39%] June 2011.

[C23] Spatially-Aware Comparison and Consensus for Clusterings.
Jeff M. Phillips, Parasaran Raman, and Suresh Venkatasubramanain.
SIAM International Conference on Data Mining (SDM), [25%] April 2011.

[C24] (Approximate) Uncertain Skylines.
Peyman Afshani, Pankaj K. Agarwal, Lars Arge, Kasper Dalgaard Larsen, and Jeff M. Phillips.
14th International Conference on Database Theory (ICDT), [41%] March 2011. *Theory of Computing Systems (TOCS)* 52:342–366, 2013. (Special Issue: ICDT 2011)

Students

Current Students Supervising

Yan Zheng (PhD).	(entered 2012)
Mina Ghashami (PhD).	(entered 2012)
Jian Ying (PhD).	(part time, entered 2013)
Pingfan Tang (PhD).	(entered 2014)
Michael Matheny (PhD).	(entered 2014)
WaiMing Tai (PhD).	(entered 2015)
Kaiqiang Wang (MS Project).	(expected Spring 2016)
Sierra Allred (Bachelors in Undergraduate Studies in *Data Science*)	(expected Spring 2016)

Graduated Students

Liang Zhang (MS Project). - first job : Microsoft. (Fall 2015)
Raghvendra Singh (MS Thesis). - first job : InsideSales. (2015)
"*Scalable Spatial Scan Statistics*"
Jamie Iong (BS Thesis). - first job : EMC. (2015)
"*Solving K-depth Coverage problem using Sweep Line Algorithm and Red Black Tree*"
Tami Y. Porter-Jones (BS Thesis). - first job : Myriad Genetics. (2015)
"*Detecting Large DNA Rearrangements Using NGS Data*"
Amey Desai (MS Thesis) - first job: UrbanEngines (Bay Area startup). (2014)
"*Streaming Algorithms for Matrix Approximation*"
Sitaram Gautum (Bachelors in Undergraduate Studies in *Data Science*) (2014)
Shashank Krishnaswamy (MS Project) - first job: Amazon. (2013)
"*Quality Control in Weather Data with Quantiles*"
Alex Clemmer (BS Thesis) - first job: Microsoft. (2013)
"*Streaming LDA*"

Funding

Total: **$1,792,996** (roughly)

- *Algorithmic Problems in Geometric Statistical Problems on Spatial Datasets.* NSF Computing Innovations Postdoctoral Fellow (Sep. 2009-Aug 2011) **$246,250**.
  - NSF 0937060 to CRA, subaward CIF-32 to the University of Utah (Sep 2009 - Aug 2010) **$140,000**
  - NSF 1019343 to CRA, subaward CIF-A-32 to the University of Utah (Sep 2010 - Aug 2011) **$106,250**
- *Synopsis Data Structures for Data Analysis in Shape Space.* (senior personnel)
  NSF-CCF 1115677 (Sep. 2011-Aug 2014) **$127,673** (out of **$347,716**).
- *Building a Mergeable and Interactive Distributed Data Layer for Big Data Summarization Systems.* (co-PI)
  NSF-BIGDATA 1251019 (Sep 2013 - Aug 2016). *about* **$308,421** (out of **$685,380**).
- *CAREER: Foundations for Geometric Analysis of Noisy Data.* (PI)
  NSF-CCF CAREER 1350888: (May 2014 - May 2019). **$521,156**
- *STORM: Spatio-Temporal Online Reasoning and Management of Large Data.* (co-PI)
  NSF-ACI 1443046 (CIF21 DIBBs) (Sep 2014 - Aug 2018). *about* **$289,493** (out of **$1,157,975**).
- *Seal: Secure Engine for AnaLytics - From Secure Similarity Search to Secure Data Analytics.* (co-PI)
  NSF-TWC 1514520 (Medium: Collaborative Research): (July 2015 - June 2019). *about* **$300,003** (out of **$600,007**).

Teaching

Data Mining (cs5140/6140) **[self-developed]**

Spring 2015: 98 | 29 undergrad + 69 graduate students.
Spring 2014: 69 | 14 undergrad + 55 graduate students.
Spring 2013: 40 | 10 undergrad + 30 graduate students.
Spring 2012: 35 | 8 undergrad + 27 graduate students.

Probability and Statistics for Engineers (cs3130/ece3530)

Fall 2014: 93 undergrad students.

Models of Computation for Massive Data (cs7960) **[self-developed]**

Fall 2013: 27 students.
Fall 2011: 29 students.

Data Mining Seminar (cs7931)

Spring 2015: *Matrix Sketching.* 15 students [self-developed]
Fall 2012: *Sampling.* 16 students [self-developed]
Fall 2010: *Modeling Data with Uncertainty.* (taught while a postdoc at the U) [self-developed]

Data Reading Group (cs7941)

Fall 2014: *Data Group Meeting.* 6 students.
Spring 2014: *Data Group Meeting.* 5 students.
Spring 2012: *Data Group Meeting.* 11 students.

External Service

Editorial Board

Associate Editor for *IEEE Transactions on Knowledge and Data Engineering (TKDE).* (2016)
SIAM Journal of Scientific Computing, Special Section for CSE15 on CSE Software and Big Data in CSE. (2015-16)

Program Committees

ACM Symposium on Principles of Database Systems (PODS). (2015,2017)
International Conference on Database Theory (ICDT). (2017)
International Symposium on Computational Geometry (SOCG). (2016)
IEEE International Conference on Data Engineering (ICDE). (2014,2016)
ACM-SIAM Symposium on Discrete Algorithms (SODA). (2015)
Workshop on Massive Data Algorithmics (MASSIVE). (2014,2015)
Fall Workshop in Computational Geometry (FWCG). (2012,2014)
International Conference on Database Systems for Advanced Applications. (2014)
ACM IKDD Conference on Data Science. (2014)
ACM International Conference on Information and Knowledge Management (CIKM). (2013)
European Symposium on Algorithms (ESA). (2013)
International Workshop on Big Dynamic Distributed Data, a VLDB Workshop. (2013)
ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). (2012)
Robotics: Science and Systems (RSS). (2006)

Journal Reviewing

Computational Geometry: Theory and Applications (CGTA), Computational Statistics and Data Analysis (CSDA), Computational Statistics (COST), Discrete and Computational Geometry (DCG), Distributed and Parallel Databases (DaPD), International Journal of Computational Geometry (IJCGA), Journal of Computational Geometry (JoCG), Journal of Discrete Algorithms (JDA), The London Mathematical Society (LMS), Wiley Journal on Statistical Analysis and Data Mining (SADM), SIAM Journal of Computing (SICOMP), SIAM Journal of Discrete Mathematics (SIDMA), SIAM Journal of Scientific Computing (SISC), ACM Transactions on Algorithms (TALG), ACM Transactions on Database Systems (TODS), IEEE Transactions on Knowledge Discovery and Data Engineering (TKDE), The Visual Computer (VisComp), IEEE Transactions on Multimedia, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), IEEE Signal Processing Letters, BMC Systems Biology.

Workshop/Conference Organization

Workshop on *Geometric Algorithms for Machine Learning* at jointly *ACM STOC* and SOCG. (2016)
SIAM CSE co-organized and served on panel on "Data Science: What is It and How to Teach It." (2015)
ACM SIGMOD demo and workshop chair / *ACM PODS* local arrangements chair. (2014)
ACM Symposium on Computational Geometry, Workshop on Computational Geometry, program committee. (2014)
Workshop on Computational Geometry in the Field (8F-CG) at SoCG 2012, organizer. (2012)
Symposium on Computational Geometry (SoCG), local arrangements. (2010)

Zvonimir Rakamarić **March, 2016**

CONTACT INFORMATION

Address: School of Computing, 50 South Central Campus Drive, Rm 3424
University of Utah, Salt Lake City, UT 84112-9205, USA
Phone: +1 (801) 581-6139
E-mail: zvonimir@cs.utah.edu
WWW: www.zvonimir.info, www.soarlab.org

EDUCATION

University of British Columbia, Vancouver, BC, Canada

Ph.D. in Computer Science, Mar 2011

- Thesis: Modular Verification of Shared-Memory Concurrent System Software
- Supervisor: Alan J. Hu

M.Sc. in Computer Science, Aug 2006

- Thesis: A Logic and Decision Procedure for Verification of Heap-Manipulating Programs
- Supervisor: Alan J. Hu

Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia

Dipl. ing. (5-year degree) in Computer Science, Jun 2002

- Thesis: Java Assembler
- Supervisor: Danko Basch

HONORS AND AWARDS

SMACK+Corral verifier combo wins 2 silver and 2 bronze medals in the 5th International Competition on Software Verification (SV-COMP), 2016

SMACK+Corral verifier combo wins 2 gold, 1 silver, and 1 bronze medal in the 4th International Competition on Software Verification (SV-COMP), 2015

NSF/TCPP CDER Center Early Adopter Award, 2013

Microsoft Research Software Engineering Innovation Foundation (SEIF) Award, 2012

TEACHING EXPERIENCE

University of Utah, Salt Lake City, UT, USA

- CS 5959 — Writing Solid Code, Fall 2015 (students: 20, instructor: 5.35, course: 4.83)
- CS 6110 — Formal Methods in System Design, Spring 2015 (students: 9, instructor: 5.64, course: 5.36). Received top teaching award from the College of Engineering.
- CS 2100 — Discrete Structures, Fall 2014 (students: 142, instructor: 5.36, course: 5.24)
- CS 2100 — Discrete Structures, Spring 2014 (students: 92, instructor: 5.06, course: 4.73)
- CS 5100/6100 — Foundations of Computer Science, Spring 2013 (students: 7, instructor: 5.54, course: 5.69). Received top teaching award from the College of Engineering.
- CS 6962 — Software Verification, Fall 2012 (students: 21, instructor: 5.40, course: 5.00)

FUNDING

National Science Foundation, "TWC:Small: Deker: Decomposing Commodity OS Kernels for Verification", CNS-1527526, PI Z. Rakamarić, co-PI A. Burtsev, Jul 2015–Jun 2018. [Award total: \$499,999; Utah share: \$499,999; My share: \$250,000]

National Science Foundation, "SHF:Small:Collaborative Research: Compositional Verification of Heterogeneous Software Protocol Stacks", CCF-1421678, PIs Z. Rakamarić and F. Howar (CMU), co-PI T. K. Azene (CMU), Jul 2014–Jun 2017. [Award total: \$499,954; Utah share: \$252,996; My share: \$252,996]

Lawrence Livermore National Laboratory (LLNL), Laboratory Directed Research and Development (LDRD), "PRUNER: Providing Reproducibility on Ubiquitously Non-deterministic Environments and Runs", PI D. H. Ahn (LLNL), co-PIs M. Schulz (LLNL), G. Gopalakrishnan, Z. Rakamarić, Jan 2014–Dec 2014. [Award total: $61,798; Utah share: $61,798; My share: $30,899]

National Science Foundation, "EAGER: Memory Models: Specification and Verification in a Concurrency Intermediate Verification Language (CIVL) Framework", CCF-1346756, PI Z. Rakamarić, co-PI G. Gopalakrishnan, Sep 2013–Aug 2015. [Award total: $299,998; Utah share: $299,998; My share: $149,999]

National Science Foundation and Semiconductor Research Corporation, "CCF: Localized, Layered Formal Hardware/Software Resilience Methods", CCF-1255776, PIs G. Gopalakrishnan and P. C. Diniz (USC), co-PI Z. Rakamarić, Apr 2013–Mar 2016. [Award total: $363,200; Utah share: $192,500; My share: $96,250]

NASA/Carnegie Mellon University, "Improving Coverage of Testing Complex Software Components", PI Z. Rakamarić, Oct 2012–Jan 2014. [Award total: $97,381; Utah share: $97,381; My share: $97,381]

Microsoft Research SEIF Award Gift, "Analysis of Heterogeneous Concurrent Programs", PI Z. Rakamarić. [Award total: $25,000; Utah share: $25,000; My share: $25,000]

PEER-REVIEWED JOURNAL PUBLICATIONS

A. Humphrey, Q. Meng, M. Berzins, D. C. B. de Oliveira, Z. Rakamarić, G. Gopalakrishnan, "Systematic Debugging Methods for Large Scale HPC Computational Frameworks", *Computing in Science and Engineering (CiSE)*, 16(3), IEEE, May 2014, pp 48–56.

D. Babić, B. Cook, A. J. Hu, Z. Rakamarić, "Proving Termination of Nonlinear Command Sequences", *Formal Aspects of Computing (FAC)*, 25(3), Springer, May 2013, pp 389–403. Invited paper.

PEER-REVIEWED CONFERENCE PUBLICATIONS

M. Carter, S. He, J. Whitaker, Z. Rakamarić, M. Emmi, "SMACK Software Verification Toolchain", *Proceedings of the 38th IEEE/ACM International Conference on Software Engineering (ICSE)*, 2016, to appear. Demonstrations Track. [Acceptance rate: $18/56 = 32\%$; Pages: 4]

K. Luckow, M. Dimjašević, D. Giannakopoulou, F. Howar, M. Isberner, T. Kahsai, Z. Rakamarić, V. Raman, "JDart: A Dynamic Symbolic Analysis Framework", *Proceedings of the 22nd International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2016, to appear. [Acceptance rate: unknown; Pages: 17]

P. Deligiannis, A. F. Donaldson, Z. Rakamarić, "Fast and Precise Symbolic Analysis of Concurrency Bugs in Device Drivers", *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, IEEE, 2015, pp 166–177. [Acceptance rate: $60/289 = 21\%$; Pages: 12]

A. Solovyev, C. Jacobsen, Z. Rakamarić, G. Gopalakrishnan, "Rigorous Estimation of Floating-Point Round-off Errors with Symbolic Taylor Expansions", *Proceedings of the 20th International Symposium on Formal Methods (FM)*, Lecture Notes in Computer Science, Springer, Vol. 9109, 2015, pp 532–550. [Acceptance rate: $32/124 = 26\%$; Pages: 19]

D. Giannakopoulou, F. Howar, M. Isberner, T. Lauderdale, Z. Rakamarić, V. Raman, "Taming Test Inputs for Separation Assurance", *Proceedings of the 29th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, ACM, 2014, pp 373–384. [Acceptance rate: $55/276 = 20\%$; Pages: 12]

Z. Rakamarić, M. Emmi, "SMACK: Decoupling Source Language Details from Verifier Implementations", *Proceedings of the 26th International Conference on Computer Aided Verification (CAV)*, Lecture Notes in Computer Science, Springer, Vol. 8559, 2014, pp 106–113. Short paper. [Acceptance rate: $11/54 = 20\%$; Pages: 7]

W. Chiang, G. Gopalakrishnan, Z. Rakamarić, A. Solovyev, "Efficient Search for Inputs Causing High Floating-point Errors", *Proceedings of the ACM SIGPLAN Symposium on Principles and Prac-*

tice of Parallel Programming (PPoPP), ACM, 2014, pp 43–52. [Acceptance rate: $28/184 = 15\%$; Pages: 10]

V. C. Sharma, A. Haran, Z. Rakamarić, G. Gopalakrishnan, "Towards Formal Approaches to System Resilience", *Proceedings of the 19th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2013, pp 41–50. [Acceptance rate: $33/71 = 47\%$; Pages: 10]

F. Howar, D. Giannakopoulou, Z. Rakamarić, "Hybrid Learning: Interface Generation through Static, Dynamic, and Symbolic Analysis", *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*, ACM, 2013, pp 268–279. [Acceptance rate: $32/124 = 26\%$; Pages: 12]

D. Babić, Z. Rakamarić, "Asynchronously Communicating Visibly Pushdown Systems", *Proceedings of the 2013 IFIP Joint International Conference on Formal Techniques for Distributed Systems (33rd FORTE/15th FMOODS)*, Lecture Notes in Computer Science, Springer, Vol. 7892, 2013, pp 225–241. [Acceptance rate: $20/49 = 41\%$; Pages: 17]

W. Chiang, G. Gopalakrishnan, G. Li, Z. Rakamarić, "Formal Analysis of GPU Programs with Atomics via Conflict-Directed Delay-Bounding", *Proceedings of the 5th NASA Formal Methods Symposium (NFM)*, Lecture Notes in Computer Science, Springer, Vol. 7871, 2013, pp 213–228. [Acceptance rate: $28/75 = 37\%$; Pages: 16]

D. Giannakopoulou, Z. Rakamarić, V. Raman, "Symbolic Learning of Component Interfaces", *Proceedings of the 19th International Static Analysis Symposium (SAS)*, Lecture Notes in Computer Science, Springer, Vol. 7460, 2012, pp 248–264. [Acceptance rate: $25/62 = 40\%$; Pages: 17]

M. Emmi, S. Qadeer, Z. Rakamarić, "Delay-Bounded Scheduling", *Proceedings of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, ACM, 2011, pp 411–422. [Acceptance rate: $49/209 = 24\%$; Pages: 12]

Peer-Reviewed Workshop Publications

M. Dimjašević, S. Atzeni, I. Ugrina, Z. Rakamarić, "Evaluation of Android Malware Detection Based on System Calls", *Proceedings of the 2nd ACM International Workshop on Security and Privacy Analytics (IWSPA)*, ACM, 2016, pp 1–8. [Acceptance rate: $6/20 = 30\%$; Pages: 8]

W. Chiang, G. Gopalakrishnan, Z. Rakamarić, "Practical Floating-point Divergence Detection", *Proceedings of the 28th International Workshop on Languages and Compilers for Parallel Computing (LCPC)*, Lecture Notes in Computer Science, Springer, Vol. 9519, 2015, pp 271–286. [Acceptance rate: $19/37 = 51\%$; Pages: 15]

S. Atzeni, G. Gopalakrishnan, Z. Rakamarić, D. H. Ahn, I. Laguna, M. Schulz, G. L. Lee, J. Protze, M. S. Müller, "Archer: Effectively Spotting Data Races in Large OpenMP Applications", *Informal Proceedings of the 8th International Workshop on Exploiting Concurrency Efficiently and Correctly (EC2)*, 2015. Position paper. [Acceptance rate: $3/3 = 100\%$; Pages: 6]

S. Lahiri, Z. Rakamarić, "Towards Automated Differential Program Verification for Approximate Computing", *Informal Proceedings of the Workshop on Approximate Computing Across the Stack (WAX)*, 2015. Position paper. [Acceptance rate: unknown; Pages: 3]

W. Chiang, G. Gopalakrishnan, Z. Rakamarić, "Unsafe Floating-point to Unsigned Integer Casting Check for GPU Programs", *Proceedings of the 8th International Workshop on Numerical Software Verification (NSV)*, Electronic Notes in Theoretical Computer Science, Elsevier, Vol. 317, 2015, pp 33–45. [Acceptance rate: unknown; Pages: 12]

J. Protze, S. Atzeni, D. H. Ahn, M. Schulz, G. Gopalakrishnan, M. S. Müller, I. Laguna, Z. Rakamarić, G. L. Lee, "Towards Providing Low-Overhead Data Race Detection for Large OpenMP Applications", *Proceedings of the LLVM Compiler Infrastructure in HPC Workshop (LLVM-HPC)*, IEEE, 2014, pp 40–47. [Acceptance rate: $5/6 = 83\%$; Pages: 8]

D. C. B. de Oliveira, Z. Rakamarić, G. Gopalakrishnan, A. Humphrey, Q. Meng, M. Berzins, "Systematic Debugging of Concurrent Systems Using Coalesced Stack Trace Graphs", *Proceedings of the 27th International Workshop on Languages and Compilers for Parallel Computing (LCPC)*, Lecture Notes in Computer Science, Springer, Vol. 8967, 2014, pp 317–331. [Acceptance rate: $25/39 = 64\%$; Pages: 15]

M. Dimjašević, D. Giannakopoulou, F. Howar, M. Isberner, Z. Rakamarić, V. Raman, "The Dart, the Psyco, and the Doop: Concolic Execution in Java PathFinder and its Applications", *Proceedings of the 2014 Java Pathfinder Workshop (JPF)*, *ACM SIGSOFT Software Engineering Notes*, 40(1), ACM, Jan 2015, pp 1–5. [Acceptance rate: $10/11 = 91\%$; Pages: 5]

D. H. Ahn, G. L. Lee, G. Gopalakrishnan, Z. Rakamarić, M. Schulz, I. Laguna, "Overcoming Extreme-Scale Reproducibility Challenges Through a Unified, Targeted, and Multilevel Toolset", *Proceedings of the 1st International Workshop on Software Engineering for High Performance Computing in Computational Science and Engineering (SE-HPCCSE)*, ACM, 2013, pp 41–44. [Acceptance rate: $7/12 = 58\%$; Pages: 4]

M. Dimjašević, Z. Rakamarić, "JPF-Doop: Combining Concolic and Random Testing for Java", *Java Pathfinder Workshop*, 2013. Extended abstract. [Acceptance rate: unknown; Pages: 4]

D. C. B. de Oliveira, Z. Rakamarić, G. Gopalakrishnan, A. Humphrey, Q. Meng, M. Berzins, "Practical Formal Correctness Checking of Million-core Problem Solving Environments for HPC", *Proceedings of the 5th International Workshop on Software Engineering for Computational Science and Engineering (SE-CSE)*, ACM, 2013, pp 75–83. [Acceptance rate: $10/15 = 67\%$; Pages: 9]

N. Ghafari, A. J. Hu, Z. Rakamarić, "Context-Bounded Translations for Concurrent Software: An Empirical Evaluation", *Proceedings of the 17th International SPIN Workshop on Model Checking Software (SPIN)*, Lecture Notes in Computer Science, Springer, Vol. 6349, 2010, pp 227–244. [Acceptance rate: $13/29 = 45\%$; Pages: 18]

BOOKS

D. Babić, Z. Rakamarić, J. Lorincz, "Guidebook for Graduate Studies Abroad" (in Croatian), 2nd edition, P.O.I.N.T., ISBN: 978-953-99805-3-3, Croatia, 2012.

INVITED TALKS

"Automated SMT-Based Verification for Reasoning About Approximations", Microsoft Research, Redmond, WA, USA, Oct 14, 2015

"SMACK: Decoupling Source Language Details from Verifier Implementations", New York University (NYU), New York, NY, USA, Sep 24, 2014

"SMACK: Decoupling Source Language Details from Verifier Implementations", Yale University, New Haven, CT, USA, Sep 23, 2014

"SMACK: Decoupling Source Language Details from Verifier Implementations", University of Texas at Austin, Austin, TX, USA, Sep 9, 2014

"Efficient Estimation of Floating-point Errors", University of Delaware, Newark, DE, USA, May 13, 2014

"Formal Analysis of GPU Programs with Atomics via Conflict-Directed Delay-Bounding", *Correct and Efficient Accelerator Programming*, Schloss Dagstuhl Seminar (by-invitation-only international seminar), Wadern, Germany, Apr 3, 2013

"Learning Symbolic Interfaces of Software Components", Brigham Young University, Provo, UT, USA, Mar 21, 2013

"Learning Symbolic Interfaces of Software Components", University of British Columbia, Vancouver, BC, Canada, Mar 13, 2013

"SMT at Utah", *Z3 Special Interest Group Meeting*, Microsoft Research, Redmond, WA, USA, Oct 22, 2012

October 2016

JOHN REGEHR

School of Computing
50 South Central Campus Drive, Rm 3190
University of Utah
Salt Lake City, UT 84112–9205

regehr@cs.utah.edu
`http://www.cs.utah.edu/~regehr`

Education

PhD, Computer Science, University of Virginia. Charlottesville, VA. Advisor: Prof. John A. Stankovic. Thesis title: "Using Hierarchical Scheduling to Support Soft Real-Time Applications on General-Purpose Operating Systems." May 2001.

Masters of Computer Science, University of Virginia. Charlottesville, VA. Advisor: Prof. Paul F. Reynolds. Project title: "An Isotach Implementation for Myrinet." May 1997.

BS, Computer Science, Kansas State University. Manhattan, KS. May 1995.

BS, Mathematics, Kansas State University. Manhattan, KS. May 1995.

Academic Positions

Professor, School of Computing, University of Utah, July 2015–present.

Associate Professor, School of Computing, University of Utah, July 2009–June 2015.

Assistant Professor, School of Computing, University of Utah, August 2003–June 2009.

Adjunct Assistant Professor, School of Computing, University of Utah. September 2002–May 2003.

Postdoctoral Fellow, School of Computing, University of Utah. Supervisor: Prof. Jay Lepreau. April 2001–July 2003.

Teaching since 2010

CS 6960, Advanced Compilers, Fall 2016. 18 students enrolled. 3 credit hours.

CS 5460/6460, Operating Systems, Spring 2015. 106 students enrolled. 4 credit hours.

CS/ECE 5785/6785, Advanced Embedded Software, Fall 2014. 38 students enrolled. 3 credit hours.

CS 5959, Writing Solid Code, Spring 2014. 21 students enrolled. 3 credit hours. I received the School of Computing Outstanding Teaching Award for this course.

CS 5962, Advanced Operating Systems, Spring 2014. 7 students enrolled. 3 credit hours.

CS 4400, Computer Systems, Fall 2013. 157 students enrolled. 4 credit hours.

CS 5460/6460, Operating Systems, Spring 2013. 55 students enrolled. 4 credit hours.

CS/ECE 5785/6785, Advanced Embedded Software, Fall 2012. 28 students enrolled. 3 credit hours.

CS 5460/6460, Operating Systems, Spring 2012. 77 students enrolled. 4 credit hours.

CS 7942, Seminar on System Support for Data Centers, Spring 2012, 3 students enrolled. 1–3 credit hours.

CS 5957, Android Projects, Fall 2011. 13 students enrolled. 3 credit hours.

CS 5460/6460, Operating Systems, Fall 2010. 56 students enrolled. 4 credit hours.

CS/ECE 5785/6785, Advanced Embedded Systems, Fall 2010. 29 students enrolled. 3 credit hours.

Conference and Journal Publications since 2010

Abstractions for Practical Virtual Machine Replay.
Anton Burtsev, David Johnson, Mike Hibler, Eric Eide, and John Regehr.
To appear in *Proceedings of the 12th International Conference on Virtual Execution Environments (VEE'16)*, Atlanta, GA, USA, April 2016.
`https://www.cs.utah.edu/~regehr/papers/vee16-xentt.pdf`

Alex Groce, Mohammad Amin Alipour, Chaoqiang Zhang, Yang Chen, and John Regehr.
Cause reduction: delta debugging, even without bugs.
In Software Testing, Verification and Reliability, Volume 26 Issue 1, January 2016.
`http://www.cs.utah.edu/~regehr/papers/mintest.pdf`

Understanding Integer Overflow in C/C++.
Will Dietz, Peng Li, John Regehr, and Vikram Adve.
In ACM Transactions on Software Engineering and Methodology (TOSEM), Volume 25, Issue 1, November 2015.
`http://www.cs.utah.edu/~regehr/papers/tosem15.pdf`

Deniable Backdoors Using Compiler Bugs.
Scotty Bauer, Pascal Cuoq, and John Regehr.
International Journal of PoC||GTFO 0x08, June 2015.
`https://www.alchemistowl.org/pocorgtfo/pocorgtfo08.pdf#page=7`

Nuno Lopes, David Menendez, Santosh Nagarakatte, and John Regehr.

Provably Correct Peephole Optimizations with Alive.
In *Proceedings of 36th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2015)*, Portland, OR, USA, June 2015.
`http://www.cs.utah.edu/~regehr/papers/pldi15.pdf`

Alex Groce, Amin Alipour, Chaoqiang Zhang, Yang Chen, and John Regehr.
Cause Reduction for Quick Testing.
In *Proceedings of the IEEE International Conference on Software Testing, Verification and Validation (ICST)*, Cleveland, Ohio, USA, March-April 2014.
This paper received the ICST 2014 Best Paper Award.
`http://www.cs.utah.edu/~regehr/papers/icst14.pdf`

Alex Groce, Chaoqiang Zhang, Mohammad Amin Alipour, Eric Eide, Yang Chen, John Regehr.
Help, help, I'm being suppressed! The significance of suppressors in software testing.
In *Proceedings of the 24th International Symposium on Software Reliability Engineering (ISSRE)*, Pasadena, CA, USA, November 2013.
`http://www.cs.utah.edu/~regehr/papers/issre13.pdf`

Yang Chen, Alex Groce, Chaoqiang Zhang, Weng-Keen Wong, Xiaoli Fern, Eric Eide, and John Regehr.
Taming Compiler Fuzzers.
In *Proceedings of 34th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2013)*, Seattle, WA, USA, June 2013.
`http://www.cs.utah.edu/~regehr/papers/pldi13.pdf`

Alex Groce, Chaoqiang Zhang, Eric Eide, Yang Chen, and John Regehr.
Swarm Testing.
In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA 2012)*, Minneapolis, MN, USA, July 2012.
`http://www.cs.utah.edu/~regehr/papers/swarm12.pdf`

John Regehr, Yang Chen, Pascal Cuoq, Eric Eide, Chucky Ellison, and Xuejun Yang.
Test-Case Reduction for C Compiler Bugs.
In *Proceedings of 33rd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2012)*, Beijing, China, June 2012.
`http://www.cs.utah.edu/~regehr/papers/pldi12-preprint.pdf`

Will Dietz, Peng Li, John Regehr, and Vikram Adve.
Understanding Integer Overflow in C/C++.
In *Proceedings of the 34th International Conference on Software Engineering (ICSE 2012)*. Zurich, Switzerland, June 2012.
This paper received the ACM SIGSOFT Distinguished Paper Award.
`http://www.cs.utah.edu/~regehr/papers/overflow12.pdf`

Pascal Cuoq, Benjamin Monate, Anne Pacalet, Virgile Prevosto, John Regehr, Boris Yakobowski, and Xuejun Yang.
Testing static analyzers with randomly generated programs.

Short paper in *Proceedings of the 4th NASA Formal Methods Symposium (NFM 2012).* Norfolk, Virginia, USA, April 2012.
`http://www.cs.utah.edu/~regehr/papers/nfm12.pdf`

Lu Zhao, Guodong Li, and John Regehr.
ARMor: Fully Verified Software Fault Isolation.
In *Proceedings of the International Conference on Embedded Software (EMSOFT)*, Taipei, Taiwan, October 2011.
`http://www.cs.utah.edu/~regehr/papers/emsoft11.pdf`

Xuejun Yang, Yang Chen, Eric Eide, and John Regehr.
Finding and Understanding Bugs in C Compilers.
In *Proceedings of 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2011)*, San Jose, CA, USA, June 2011.
`http://www.cs.utah.edu/~regehr/papers/pldi11-preprint.pdf`

Peng Li and John Regehr.
T-Check: Bug Finding for Sensor Networks.
In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN)*, SPOTS track, Stockholm, Sweden, April 2010.
`http://www.cs.utah.edu/~regehr/papers/ipsn553s-li.pdf`

Graduated Students

Xuejun Yang, PhD, May 2014

Yang Chen, PhD, December 2013

Jianjun Duan, PhD, July 2013

Anton Burtsev, PhD, December 2012

Lu Zhao, PhD, May 2012

Rohit Pagariya, MS, December 2010

Venkat Chakravarthy, MS, May 2009

Nathan Cooprider, PhD, August 2008

Usit Duongsaa, MS, May 2006

Robert Ricci

Research Assistant Professor
School of Computing, University of Utah

50 South Central Campus Drive, Room 3190, Salt Lake City, Utah 84112
ricci@cs.utah.edu *+1-801-581-8354* *www.flux.utah.edu/users/ricci*

Research Interests

My research interests are in the fields of **systems** and **networking**. Because experimentation is central to both systems and networking research, much of my energy has gone into creating **top-quality experimental environments**. Designing these environments raises a number of different research problems, and I have worked in a diverse set of areas including distributed systems, combinatorial optimization, security, networking, simulation, and embedded systems. Much of my research has been done in the context of the widely-used **Emulab** testbed and its successors—I have been a primary architect and implementor of Emulab since 2000.

Education

2010	**Ph.D.**, University of Utah Advised by Jay Lepreau until his passing in 2008, then by Sneha Kasera. Dissertation: *Enhancing Realism and Scalability in Network Testbeds*
2001	**Honors B.S.**, University of Utah Thesis: *Agile Protocols, an Application of Active Networking to Censor–Resistant Publishing Networks*

Academic Appointments

2010–present	**Research Assistant Professor**, University of Utah School of Computing Co-director of the Flux Research Group, which has more two dozen members, including faculty, research staff, postdocs, and students (Ph.D., Masters, and undergraduate).
2007, 2009	**Adjunct Professor**, Westminster College, Salt Lake City, Utah Taught undergraduate Computer Science classes as an adjunct at a small liberal-arts college.
2001–2010	**Research Staff**, University of Utah Member of the Flux Research Group, founded by Jay Lepreau.
2000–2001	**Undergraduate Research Assistant**, University of Utah I began my research career with the Flux Group while an undergraduate.

Testbeds

I have been one of the primary designers and implementors of the **Emulab** testbed since 2000, and am leading the development of some of it successors, such as **CloudLab**. These testbeds are central resources in the networking, operating systems, and distributed systems communities. Collectively, they have well over **10,000 users** from nearly every US state and dozens of countries throughout the world, spanning every inhabited continent. These users have run more than **half a million experiments** and hundreds of papers have been published based on research conducted on these testbeds. The software base that runs these testbeds is **open-source**, and more than fifty organizations worldwide, ranging from academic institutions to private companies, have built their own testbeds based on it. This software has played a critical role in subsequent testbeds with a variety of focuses: NSF's **CloudLab** (cloud computing), **GENI** (federation), **PhantomNet** (mobile networking), **PRObE** (scale) and **Apt** (adaptability); DARPA's **National Cyber Range** (security), and DHS's **DETERLAB** (security). (I have been directly involved in the development of all of these except DETERLAB.) The Emulab facility and codebase are key parts of the nationwide GENI infrastructure and several international federations in Europe, Brazil, Japan, and South Korea.

These testbeds (particularly CloudLab) have received significant attention in the **press**, including the Boston Globe, the Chronicle of Higher Education, local TV and radio stations, Slashdot, The Register, HPCWire, and numerous other publications.

Publications

Most Cited Works

Citations according to Google Scholar as of February 24, 2016

1,435 **White, et al., OSDI** *2002*
"An Integrated Experimental Environment for Distributed Systems and Networks"

236 **Ricci, et al., SIGCOMM CCR** *2003*
"A Solver For the Network Testbed Mapping Problem"

206 **Johnson, et al., INFOCOM** *2006*
"Mobile Emulab: A Robotic Wireless and Sensor Network Testbed"

193 **Hibler, et al., USENIX ATC** *2008*
"Large-scale Virtualization in the Emulab Network Testbed"

125 **Berman, et al., COMNETS** *2014*
"GENI: A Federated Testbed For Innovative Network Experiments"

Conference and Workshop Proceedings

1. "Introducing Configuration Management Capabilities into CloudLab Experiments". Dmitry Duplyakin and **Robert Ricci**. In *Proceedings of the International Workshop on Computer and Networking Experimental Research Using Testbeds (CNERT)*, April 2016.

Awarded best paper.

2. “OpenEdge: A Dynamic and Secure Open Service Edge Network”. Josh Kunz, Christopher Becker, Mohamed Jamshidy, Sneha Kasera, **Robert Ricci**, and Jacobus Van der Merwe. In *Proecceings of the Ninth IEEE/IFIP Network Operations and Management Symposium (NOMS)*, April 2016. To appear
3. “KnowNet: Towards a Knowledge Plane for Enterprise Network Management”. Ren Quinn, Josh Kunz, Aisha Syed, Joe Breen, Sneha Kasera, **Robert Ricci**, and Jacobus Van der Merwe. In *Proecceings of the Ninth IEEE/IFIP Network Operations and Management Symposium (NOMS)*, April 2016. To appear
4. ❖ “Realistic Packet Reordering for Network Emulation and Simulation”. Aisha Syed and **Robert Ricci**. In *Proceedings of the Eleventh ACM International Conference on Emerging Networking EXperiments and Technologies (CoNEXT)*, December 2015. Short paper.
5. “POTASSIUM: Penetration Testing as a Service”. Richard Li, Dallin Abendroth, Xing Lin, Yuankai Guo, Hyun wook Baek, Eric Eide, **Robert Ricci**, and Jacobus Van der Merwe. In *Proceedings of the Sixth ACM Symposium on Cloud Computing (SOCC)*, August 2015.
6. “Trust as the Foundation of Resource Exchange in GENI”. Marshall Brinn, Nicholas Bastin, Andrew Bavier, Mark Berman, Jeffrey Chase, and **Robert Ricci**. In *Proceedings of the 10th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom)*, June 2015.
7. “Metadata Considered Harmful ... to Deduplication”. Xing Lin, Fred Douglis, Jim Li, Xudong Li, **Robert Ricci**, Stephen Smaldone, and Grant Wallace. In *Proceedings of the 7th USENIX Workshop on Hot Topics in Storage and File Systems*, June 2015.
8. “Using Deduplicating Storage for Efficient Disk Image Deployment”. Xing Lin, Mike Hibler, Eric Eide, and **Robert Ricci**. In *Proceedings of the 10th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom)*, June 2015.
9. “SMORE: Software-Defined Networking Mobile Offloading Architecture”. Junguk Cho, Binh Nguyen, Arijit Banerjee, **Robert Ricci**, Jacobus Van der Merwe, and Kirk Webb. In *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications and Challenges*, August 2014.

10. "Secret Key Extraction using Bluetooth Wireless Signal Strength Measurements". Sriram Nandha Premnath, Prarthana Lakshmane Gowda, Sneha Kumar Kasera, Neal Patwari, and **Robert Ricci**. In *IEEE International Conference on Sensing, Communications and Networking (SECON)*, June 2014.

11. ❖ "Operational Experiences with Disk Imaging in a Multi-Tenant Datacenter". Kevin Atkinson, Gary Wong, and **Robert Ricci**. In *Proceedings of the Eleventh USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2014.

12. "Weir: A Streaming Language for Performance Analysis". Anton Burtsev, Nikhil Mishrikoti, Eric Eide, and **Robert Ricci**. In *Proceedings of the 7th Workshop on Programming Languages and Operating Systems (PLOS)*, November 2013.

13. ❖ "Fast and Flexible: Parallel Packet Processing with GPUs and Click". Weibin Sun and **Robert Ricci**. In *Proceedings of the ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, October 2013.

14. ❖ "How To Build a Better Testbed: Lessons From a Decade of Network Experiments on Emulab". Fabien Hermenier and **Robert Ricci**. In *Proceedings of the 8th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom)*, June 2012. **Awarded best paper**.

15. "Towards Fair Sharing of Block Storage in a Multi-tenant Cloud". Xing Lin, Yun Mao, Feifei Li, and **Robert Ricci**. In *Proceedings of the 4th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, June 2012.

16. "Designing a Federated Testbed as a Distributed System". **Robert Ricci**, Jonathon Duerig, Leigh Stoller, Gary Wong, Srikanth Chikkulapelly, and Woojin Seok. In *Proceedings of the 8th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom)*, June 2012.

17. "Harnessing GPU Computing for Storage Systems in the OS Kernel". Weibin Sun, **Robert Ricci**, and Matthew J. Curry. In *Proceedings of the Fifth International Systems and Storage Conference (SYSTOR)*, June 2012.

18. "Partitioning Trust in Network Testbeds". Gary Wong, **Robert Ricci**, Jonathon Duerig, Leigh Stoller, Srikanth Chikkulapelly, and Woojin Seok. In *Proceedings of the Software Testing and Internet Testbeds Mini-Track, HICSS 45*, January 2012.

19. “Emergency Service in Wi-Fi Networks Without Access Point Association”. Manav Seth, Sneha Kasera, and **Robert Ricci**. In *Proccedings of the First International Conference on Wireless Technologies for Humanitarian Relief (ACWR)*, December 2011.

20. “Trusted Disk Loading in the Emulab Network Testbed”. Cody Cutler, Mike Hibler, Eric Eide, and **Robert Ricci**. In *Proceedings of the Third Workshop on Cyber Security Experimentation and Test (CSET)*, August 2010.

21. ❖ “Modeling and Emulation of Internet Paths”. Pramod Sanaga, Jonathon Duerig, **Robert Ricci**, and Jay Lepreau. In *Proceedings of the Sixth USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2009.

22. “Securing the Frisbee Multicast Disk Loader”. **Robert Ricci** and Jonathon Duerig. In *Proceedings of the First Workshop on Cyber Security and Test (CSET)*, July 2008.

23. ❖ “Large-scale Virtualization in the Emulab Network Testbed”. Mike Hibler, **Robert Ricci**, Leigh Stoller, Jonathon Duerig, Shashi Guruprasad, Tim Stack, Kirk Webb, and Jay Lepreau. In *Proceedings of the USENIX Annual Technical Conference*, June 2008.

24. ❖ “The Flexlab Approach to Realistic Evaluation of Networked Systems”. **Robert Ricci**, Jonathon Duerig, Pramod Sanaga, Daniel Gebhardt, Mike Hibler, Kevin Atkinson, Junxing Zhang, Sneha Kasera, and Jay Lepreau. In *Proceedings of the Fourth USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2007.

25. “Flexlab: A Realistic, Controlled, and Friendly Environment for Evaluating Networked Systems”. Jonathon Duerig, **Robert Ricci**, Junxing Zhang, Daniel Gebhardt, Sneha Kasera, and Jay Lepreau. In *Proceedings of HotNets-V*. ACM SIGCOMM, June 2006.

26. “Leveraging Bloom Filters For Smart Search Within NUCA Caches”. **Robert Ricci**, Steve Barrus, Dan Gebhardt, and Rajeev Balasubramonian. In *Proceedings of the Sixth Workshop on Complexity-Effective Design (WCED)*, June 2006.

27. “Mobile Emulab: A Robotic Wireless and Sensor Network Testbed”. David Johnson, Tim Stack, Russ Fish, Daniel Flickinger, Leigh Stoller, **Robert Ricci**, and Jay Lepreau. In *Proceedings of IEEE INFOCOM*, April 2006.

28. “Integrated Network Experimentation using Simulation and Emulation”. Shashi Guruprasad, **Robert Ricci**, and Jay Lepreau. In *Proceedings of the First International Conference on Testbeds and*

Research Infrastructures for the Development of Networks and Communities (TridentCom), February 2005. ↗

29. "Implementing the Emulab-PlanetLab Portal: Experiences and Lessons Learned". Kirk Webb, Mike Hibler, **Robert Ricci**, Austin Clements, and Jay Lepreau. In *Proceedings of the First Workshop on Real, Large Distributed Systems (WORLDS)*. USENIX, December 2004. ↗

30. ❖ "Fast, Scalable Disk Imaging with Frisbee". Mike Hibler, Leigh Stoller, Jay Lepreau, **Robert Ricci**, and Chad Barb. In *Proceedings of the USENIX Annual Technical Conference*. USENIX, June 2003. ↗

31. ❖ "An Integrated Experimental Environment for Distributed Systems and Networks". Brian White, Jay Lepreau, Leigh Stoller, **Robert Ricci**, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. In *Proceedings of the USENIX Symposium on Operating System Design and Implementation (OSDI)*. USENIX, December 2002. ↗

32. "Active Protocols for Agile Censor-Resistant Networks". **Robert Ricci** and Jay Lepreau. In *Proceedings of HotOS-VIII*. USENIX, May 2001. ↗

Journal and Magazine Articles

1. "PhantomNet: Research Infrastructure for Mobile Networking, Cloud Computing and Software-Defined Networking". Arijit Banerjee, Junguk Cho, Eric Eide, Jonathon Duerig, Binh Nguyen, **Robert Ricci**, Jacobus Van der Merwe, Kirk Webb, and Gary Wong. *ACM GetMobile*, 19(2), April 2015. ↗

2. ❖ "Apt: A Platform for Repeatable Research in Computer Science". **Robert Ricci**, Gary Wong, Leigh Stoller, Kirk Webb, Jonathon Duerig, Keith Downie, and Mike Hibler. *ACM SIGOPS Operating Systems Review*, 49(1), January 2015. ↗

3. ❖ "Introducing CloudLab: Scientific Infrastructure for Advancing Cloud Architectures and Applications". **Robert Ricci**, Eric Eide, and The CloudLab Team. *USENIX ;login:*, 39(6), December 2014. ↗

4. "The InstaGENI Initiative: An Architecture for Distributed Systems and Advanced Programmable Networks". Nicholas Bastin, Andy Bavier, Jessica Blaine, Jim Chen, Narayan Krishnan, Joe Mambretti, Rick McGeer, **Robert Ricci**, and Nicki Watts. *Computer Networks*, 61(0):24–38, March 2014. ↗

5. ❖ "GENI: A Federated Testbed For Innovative Network Experiments". Mark Berman, Jeffrey S Chase, Lawrence Landweber, Akihiro Nakao, Max Ott, Dipankar Raychaudhuri, **Robert Ricci**, and Ivan Seskar. *Computer Networks*, 61(0):5–23, March 2014. ↗

6. "An Architecture For International Federation of Network Testbeds". **Robert Ricci**, Gary Wong, Leigh Stoller, and Jonathon Duerig. *IEICE Transactions*, E96-B(1), January 2013. Invited paper. ↗

7. "Getting Started with GENI: A User Tutorial". Jonathon Duerig, **Robert Ricci**, Leigh Stoller, Matt Strum, Gary Wong, Charles Carpenter, Zongming Fei, James Griffioen, Hussamuddin Nasir, Jeremy Reed, and Xiongqi Wu. *ACM SIGCOMM Computer Communication Review (CCR)*, 42(1):72–77, January 2012. Invited paper. ↗

8. "Lessons From Resource Allocators for Large-Scale Multiuser Testbeds". **Robert Ricci**, David Oppenheimer, Jay Lepreau, and Amin Vahdat. *ACM SIGOPS Operating Systems Review*, January 2006. ↗

9. ❖ "A Solver for the Network Testbed Mapping Problem". **Robert Ricci**, Chris Alfeld, and Jay Lepreau. *ACM SIGCOMM Computer Communications Review*, 33(2):65–81, April 2003. ↗

Books and Book Chapters

1. ❖ Rick McGeer, Mark Berman, Chip Elliott, and **Robert Ricci**, editors. *GENI: Prototype of the Next Internet*. Springer-Verlag, New York, 2016. In production for publication July 2016

2. "The InstaGENI Project". Rick McGeer and **Robert Ricci**. In Rick McGeer, Mark Berman, Chip Elliott, and **Robert Ricci**, editors, *GENI: Prototype of the Next Internet*, chapter 14. Springer-Verlag, New York, 2016. Invited chapter; in production for publication July 2016

3. "Emulab". **Robert Ricci**. In Rick McGeer, Mark Berman, Chip Elliott, and **Robert Ricci**, editors, *GENI: Prototype of the Next Internet*, chapter 2. Springer-Verlag, New York, 2016. Invited chapter; in production for publication July 2016

4. "The Need for Flexible Mid-scale Computing Infrastructure". **Robert Ricci**. In Rick McGeer, Mark Berman, Chip Elliott, and **Robert Ricci**, editors, *GENI: Prototype of the Next Internet*, chapter 6. Springer-Verlag, New York, 2016. Invited chapter; in production for publication July 2016

Tech Reports and Online Articles

1. ❖ **Robert Ricci** and Nick Feamster, editors. *Report of the NSF Workshop on Software Defined Infrastructures and Software Defined Exchanges*, Washington, DC, February 2016

2. "Rethinking Abstractions in Big Data: Why, Where, How, and What". Mary Hall, Robert M. Kirby, Feifei Li, Miriah Meyer, Valerio Pascucci, Jeff M. Phillips, **Robert Ricci**, Jacobus Van der Merwe, and Suresh

Venkatasubramanian. Technical Report UUCS-13-002, University of Utah, June 2013. arXiv:1306.3295.

3. "Augmenting Operating Systems With the GPU". Weibin Sun and **Robert Ricci**. Technical Report FTN-2011-02, University of Utah, 2011. arXiv:1305.3345.

4. "Optimizing IP Address Assignment on Network Topologies". Jonathon Duerig, **Robert Ricci**, John Byers, and Jay Lepreau. Technical Report FTN-2006-02, University of Utah, February 2006.

Teaching

Courses

Spring 2015 **CS 6963: Evaluating Networked Systems**, University of Utah
I taught this course, which I designed in Spring 2014, for a second time. It was taken by a mix of eight PhD, MS, and BS students. Materials for this class were used as part of a networking class at NYU Poly.

Spring 2014 **CS 6963: Evaluating Networked Systems**, University of Utah
Developed a new graduate-level course to acquaint students with the theory and practice of evaluating systems that have a network as a major component, with the goal of preparing students to conduct rigorous evaluations as part of their own research as well as looking with a critical eye at evaluations they find in the literature. Ten students (MS and PhD) took the initial offering of the course. All course materials are available online.

Spring 2009 **CMPT 355: Compilers**, Westminster College
Taught a four credit-hour class of 5 junior and senior Computer Science majors, covering compilers and related technologies. Took over the class mid-semester when the primary instructor took maternity leave.

Fall 2007 **CMPT 251: Computer Organization**, Westminster College
Taught a four credit-hour class of 13 second-year Computer Science majors, introducing a range of fundamental topics in operating systems and computer architecture. The course received excellent reviews.

Other

2010–2013 **Hands-on tutorials**, GENI Engineering Conferences
I have presented eight tutorials at GENI Engineering Conferences. These hands-on events covered the user of the GENI facility and various experimenter tools, and have typically lasted 2–3 hours with 20–60 attendees.

Fall 2013 **Guest lectures in CS 6480: Advanced Computer Networks**, Utah
Gave guest lectures on networked systems evaluation in general and the Emulab and GENI testbeds in particular.

Spring 2007 **Guest lectures in CS 6490: Network Security**, University of Utah
Gave guest lectures, with accompanying homework assignment, on systems and programming aspects of security, including buffer overflows and low-level network attacks such as ARP poisoning and DNS attacks.

August 2002 **Hands-on Tutorial**, SIGCOMM Conference
Part of a team of three that prepared and presented a well-received full-day tutorial at the premier networking conference. Taught attendees how to use the Emulab testbed for research and classwork, including a hands-on component.

Summer 1999 **Teaching Assistant for CP SC 2020: Computer Science II**, Utah
TA for an entry-level computer science class. Responsibilities included teaching two discussion sections per week, of about two dozen students each.

Spring 1999 **Teaching Assistant for CP SC 2010: Computer Science I**, Utah
TA for an entry-level computer science class. Responsibilities included teaching two discussion sections per week, of about two dozen students each.

Advising

Postdocs

2011 **Fabien Hermenier** *Faculty, Université Nice Sophia Antipolis*

Ph.D. Students

2015–present **Dmitry Duplyakin**
Student at University of Colorado Boulder; informally co-advising with Jed Brown of CU-Boulder

2014–present **Aisha Syed**
Co-advised with Kobus Van der Merwe

2013–present **Junguk Cho**
Co-advised with Kobus Van der Merwe

2013–present **Stephen “Ren” Quinn**
Co-advised with Kobus Van der Merwe

2009–2015 **Xing Lin** *NetApp Advanced Technology Group*
Thesis: “Using Similarity in Content and Access Patterns to Improve Space Efficiency and Performance in Storage Systems”

2009–2014 **Weibin Sun** *Google*
Thesis: “Harnessing GPU Computing in System Level Software”

M.S. Students

2015–present **Aniraj Kesavan**

2015–present **Anil Kumar**

2014–2015 **Anil Mallapur** *LinkedIn*

2012–2014 **Aisha Syed** *Ph.D. Student, Utah*
Thesis: "Realistic Traffic Shaping in Dummynet Link Emulator"

2010–2014 **Srikanth Manikarnike**
Project: "Enhancing Dummynet to Reproduce Real Link Characteristics"

2012–2013 **Nikhil Mishrikoti** *Cisco Systems*
Project: "Performance Analysis of Virtual Environments"

2011–2013 **Srikanth Raju** *Coverity*
Project: "Image Import and SSH Security in Emulab"

2011–2013 **Yathindra Dev Naik** *NetApp*
Project: "Xen-Cap: A Capability Framework for Xen"

2010–2011 **Srikanth Chikkulapelly** *Amazon AWS*
Thesis: "A Scalable and Flexible Node Configuration Service in an Advanced Network Testbed"

2009–2011 **Raghuveer Pullankandam** *Adobe Systems*
Thesis: "EmuStore: Large Scale Disk Image Storage and Deployment in the Emulab Network Testbed"

B.S./M.S. Students

2015–present **Keith Downie**

2009–2013 **Matt Strum** *Amazon Silk Browser Team*
Thesis: "FlowOps: Open Access Network Management and Operation". Co-advised with Kobus Van der Merwe.

2012–2013 **Jared Rose**
Project: "Anonymous File Transfer Network"

B.S. Students

2013–2015 **Keith Downie** *B.S./M.S. Student, Utah*

2009–2012 **Cody Cutler** *Ph.D. Student, MIT*
Thesis: "Trusted Disk Loading in the Emulab Network Testbed"

Interns

2015 **Dmitry Duplyakin**
PhD student at the University of Colorado, Boulder

2015 **Brenda Lamwaka**
International exchange student from Mbarara University of Science and Technology, Uganda

Professional Activities and Service

External Service

2015–2016	**Workshop Co-Chair**, NSF "Beyond the Internet" planning workshop Co-chair, with Nick Feamster of Princeton, of a workshop to inform future NSF planning on research programs looking "beyond the Internet."
2015–present	**Science Board**, PIK Journal Member of the scientific board for German journal *Praxis der Informationsverarbeitung und Kommunikation* (articles are in English).
2014	**Workshop Co-Chair**, NSFCloud Workshop on Experimental Support for Cloud Computing Co-organizer (with Kate Keahey of the University of Chicago/Argonne) of a workshop to bring together the community of potential users of the NSFCloud infrastructure.
2014	**Workshop Co-Chair**, Workshop on the Development of a Next-Generation Cyberinfrastructure Co-organizer (with Victor Hazlewood of the University of Tennessee Knoxville) of a workshop that brought together several large NSF-funded infrastructure communities (XSEDE, Grid, GENI, NSFCloud) and federal government attendees to talk about federation and collaboration between these communities.
2013–present	**Advisory Board**, Fed4FIRE On the advisory board of Fed4FIRE, a €10M European project consisting of 17 partner organizations from 8 countries.
2011–present	**Co-chair**, GENI Architecture Group Co-chair, with Marshall Brinn of BBN Technologies, of the group tasksed with defining the architecture of the NSF GENI project.
2008–present	**Technical lead**, GENI "Cluster C" As leader of the ProtoGENI project at the University of Utah, acting as head of a large collaborative effort involving over 20 projects from over 15 institutions. This position involves providing architectural vision for the collaboration, coordinating among groups, and planning and leading frequent meetings.
2014	**Workshop Organizer**, Workshop on the Development of a Next-Generation Cyberinfrastructure Co-organizer, with Victor Hazelwood of UTK, of an NSF-sponsored workshop on the development of next-generation cyberinfrastructures; it has a special emphasis on interoperation and federation.
2013	**Local arrangements chair**, GENI Engineering Conference Handled local arrangements for the sixteenth GENI Engineering Conference, held on the University of Utah campus.

2010	**Workshop organizing committee**, QUILT GENI Workshop Participated in the planning of a workshop to engage campus and regional networks in the NSF GENI project.
2009–2011	**Co-chair**, GENI Control Framework Working Group Co-chair, with Jeff Chase of Duke, of the GENI Control Framework Working group.
2009	**Local arrangements chair**, GENI Engineering Conference Handled local arrangements for the sixth GENI Engineering Conference, held in part on the University of Utah campus. The conference had approximately 200 attendees.
2009	**Network Research Strategic Planning Team**, Internet2 Helped to set network research priorities for the Internet2 national research and education network.
2007–2009	**Active participant**, GENI Control Framework Working Group Continuation of the work of the GENI Facility Architecture Working group under the guidance of the new GENI Project Office.
2006–2007	**Active participant**, GENI Facility Architecture Working Group Designed facility architecture for GENI, the NSF's effort to create a tested for next-generation Internet designs.
1999–2007	**Organizer and judge**, ACM High School Programming Contest Assisted with the annual High School Programming contest sponsored by the ACM and the University of Utah. I have helped with all aspects of the competition, from judging submissions and designing programming problems to acting as chief judge.

Departmental Service

2014–2015	Hiring subcommittee, Architecture
2013–present	Track committee, Networked Systems
2013–2014	Hiring subcommittee, Operating Systems
2010–2014, 2016	Graduate admissions committee
2013	Assisted with revision of the Graduate Handbook
2012–2013	Hiring subcommittee, Security
2011–2012	Auxiliary faculty review committee
2008–2011	Hiring subcommittee, Lepreau Professorship

Program Committees

2016	ACM SIGCOMM Workshop on Distributed Cloud Computing
2016	Workshop on Reproducibility in Parallel Computing, held in conjunction with EuroPar
2014	SIGCOMM Conference on emerging Networking EXperiments and Technologies (**CoNEXT**)

2014	SIGCOMM Workshop on Distributed Cloud Computing (**DCC**)
2014	Experimental Evaluation and Testbeds track of IEEE International Conference on Mobile Ad hoc and Sensor Systems (**MASS**)
2013	IEEE International Workshop on Future Internet Technology (**IWFIT**)
2012	Steering committee for a **shadow PC** for the USENIX Symposium on Networked Systems Design and Implementation (**NSDI**)
2012	International Conference on Computer Communication Networks (**ICCCN**), Network Architectures and Clean-Slate Designs track
2011	IEEE International Conference on Network Protocols (**ICNP**)
2010	ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (**VISA**)
2009	International Conference on Testbeds and Research Infrastructures (**TridentCom**)
2009	Workshop on Cyber Security Experimentation and Test (**CSET**)
2009	ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (**VISA**)
2008	Workshop on Cyber Security Experimentation and Test (**CSET**)

Funding

Current

NSF, $2.2M	"CI-EN: Revitalizing Emulab for Modern Networking and Systems Research", *2015–2018*, PI: Eric Eide, **co-PI**: Robert Ricci
NSF, $10M	"CloudLab: Flexible Scientific Infrastructure to Support Fundamental Advances in Cloud Architectures and Applications", *2014–2017*, **PI**: Robert Ricci, co-PIs: Aditya Akella (Wisconsin), KC Wang (Clemson), Chip Elliott (BBN), Mike Zink (UMass), Glenn Ricart (US Ignite)
NSF, $3.4M	"MRI: Development of Apt, a Testbed Instrument With Adaptable Profiles for Network and Computational Science", *2013–2016*, **PI**: Robert Ricci, co-PIs: Kobus Van der Merwe, Eric Eide, Julio Facelli, and Steve Corbató. Includes 30% University of Utah cost-sharing.
NSF, $2M	"CI-ADDO-NEW: PhantomNet: An End-to-End Mobile Network Testbed", *2013–2016*, PI: Kobus Van der Merwe, **co-PI**: Robert Ricci
NSF, $1.15M	"NeTS: Medium: KnowOps - Making Network Management and Operations Software Defined", *2013–2016*, PI: Kobus Van der Merwe, **co-PI**s: Robert Ricci, Sneha Kumar Kasera, and Suresh Venkatasubramanian
NSF, $1M	"TWC: Medium: Collaborative: TCloud: A Self-Defending, Self-Evolving and Self-Accounting Trustworthy Cloud Platform", *2013–2016*, Grant from the National Science Foundation, PI: Kobus Van der Merwe, **co-PI**s: Robert Ricci, Eric Eide and Fefei Li

Prior

GPO, $497K	"Experimenter Tools and Training for a More User-Friendly and Sustainable GENI", *2013–2015*, **PI**: Robert Ricci, Contract from GENI Project Office (BBN Technologies)
GPO, $199K	"Adopt-A-GENI: Bringing users into the GENI Community", *2013–2015*, PI: Kobus Van der Merwe, **co-PI**: Robert Ricci, Sub-contract via the University of Kentucky from from the GENI Project Office
NSF, $1M	"CC-NIE Integration: Science Slices Converting Network Research Innovation into Enhanced Capability for Computational Science and Engineering at the University of Utah", *2013–2015*, PI: Steve Corbató, **co-PI**s: Kobus Van der Merwe, Robert Ricci, Adam Bolton, and Thomas Cheatham
Corp., $150K	"Network Management and Operation for Open Access Networks", *2013–2014*, PI: Kobus van der Merwe, **co-PI**s: Robert Ricci and Sneha Kasera, Grant from Entrypoint LLC
GPO, $272K	"InstaGENI Meso-Scale Prototype", *2012–2014*, **PI**: Robert Ricci, Subcontract from HP via GENI Project Office
GPO, $254K	"Education and Support For GENI Experimenters", *2011–2014*, **PI**: Robert Ricci, Contract from GENI Project Office (BBN Technologies)
NSF, $50K	"The Sixteenth GENI Engineering Conference", *2013*, **PI**: Robert Ricci
NSF, $1.0M	"CI-ADDO-EN: Enhancing Emulab for Virtualization and Clouds", *2011–2013*, **PI**: Robert Ricci, co-PIs: Eric Eide and Mike Hibler
Corp., $25K	"Augmenting Operating Systems with the GPU", *2011–2012*, **PI**: Robert Ricci, Fellowship awarded to Weibin Sun by NVIDIA
NSF, $475K	"Collaborative Research: PRObE - The NSF Parallel Reconfigurable Observational Environment for Data Intensive Super-Computing and High End Computing", *2010–2015*, **PI**: Robert Ricci, Subcontract from New Mexico Consortium (via NSF)
GPO, $534K	"Experiment Workflow Tools and Services for GENI", *2010–2012*, **PI**: Robert Ricci, Contract from GENI Project Office (BBN Technologies)
GPO, $459K	"Integrating New Projects into the ProtoGENI Control Framework", *2010–2012*, **PI**: Robert Ricci, Contract from GENI Project Office (BBN Technologies)
GPO, $760K	"End-To-End ProtoGENI", *2008–2012*, **PI**: Robert Ricci, Contract from GENI Project Office (BBN Technologies)
NSF, $1.7M	"MRI: Evolutionary Development of an Advanced Distributed Testbed", *2007–2013*, PI: John Regehr, **co-PI**s: Robert Ricci and Steve Corbató
NSF, $1.0M	'NeTS-ProWin: Software Radio Testbeds: One Large, Many Small", *2005–2011*, PI: Sneha Kasera, **co-PI**: Robert Ricci
NSF, $1.2M	"NeTS-ProWin: An Open, Low Cost, High Quality Software Radio Platform and Testbed", *2004–2010*, PI: Sneha Kasera, **co-PI**, Robert Ricci

Talks

2016

March	❖ **Cloud Research With CloudLab**, Invited talk, Texas Tech, Lubbock, Texas Invited to give a talk at the Department of Computer Science and Cloud and Autonomic Computing Center at Texas Tech University.

2015

December	❖ **Infrastructure for Computer Systems Experimentation**, Invited talk, TU Darmstadt, Darmstadt, Germany Invited to give a talk giving a history of the Flux Group's development of research infrastructure over the last 15 years.
November	❖ **Building Community Around Testbeds**, Invited talk, NSF Workshop on Accessible Remote Testbeds, Washington, DC Invited to address a meeting of prospective testbed builders from the NSF ENG and CISE directorates
September	**Federated Monitoring**, GENI-Fed4FIRE Meeting, Washington, DC Invited talk at a meeting between the US GENI project and EU Fed4FIRE project.
September	**Federation Strategy**, GENI-Fed4FIRE Meeting, Washington, DC Invited talk at a meeting between the US GENI project and EU Fed4FIRE project.
September	**CloudLab Updates and Federation**, GENI-Fed4FIRE Meeting, Washington, DC Invited talk at a meeting between the US GENI project and EU Fed4FIRE project.
August	**Cloud Computing in HPC**, Invited talk, RMACC, Boulder, CO Invited to lead a session on cloud computing in HPC
July	❖ **CloudLab**, Plenary panel, XSEDE 15, St. Louis, MO Invited to participate as a panelist in a session discussing new and upcoming NSF-funded infrastructure
July	**Federation in CloudLab**, XSEDE 15, St. Louis, MO Invited talk about the internal and external federation aspects of CloudLab.
June	**Getting started with CloudLab and OpenStack**, Tutorial, GEC #23, Champaign, IL 2 hour hands-on tutorial presented at GEC #23 on the basics of creating a cloud in CloudLab. Approximately 30 attendees.
May	❖ **CloudLab Train-the-Trainers session**, Workshop, Salt Lake City, UT 2 day workshop for ACI-REF facilitators and University of Utah CHPC staff. Included a hands-on tutorial, and several presentations on the details of an intended use of CloudLab.

April	**CloudLab**, Cyber-physical Systems Week, Seattle, WA Remote presentation to a meeting of researchers and industrial practitioners involved in Cyber-physical Systems about how CloudLab can be used for CPS research.
March	❖ **Getting started with CloudLab and OpenStack**, Tutorial, GEC #22, Arlington, VA 1.5 hour hands-on tutorial presented at GEC #22 on the basics of creating a cloud in CloudLab. Approximately 60 attendees.
February	**CloudLab**, NITRD MAGIC meeting, Washington, DC Remote presentation to a meeting of the Middleware And Grid Interagency Coordination (MAGIC) group of the federal Networking and Information Technology Research and Development (NITRD) project.

2014

November	❖ **CloudLab**, NSFCloud Workshop, Arlington, VA Co-organized a community workshop on the NSFCloud facilities; gave a talk describing and demonstrating CloudLab.
November	❖ **CloudLab**, GENI-Fed4FIRE Meeting, Paris, France Invited talk at a meeting between the US GENI project and EU Fed4FIRE project.
November	**Cloud Research in the US**, GENI-Fed4FIRE Meeting, Paris, France Invited talk at a meeting between the US GENI project and EU Fed4FIRE project.
November	**Workshop on the Development of a Next-Generation Cyberinfrastructure**, GENI-Fed4FIRE Meeting, Paris, France Invited talk at a meeting between the US GENI project and EU Fed4FIRE project.
October	❖ **CloudLab**, GEC #21, Bloomington, IN
October	**Using GENI in "Evaluating Networked Systems**, GEC #21, Bloomington, IN
October	❖ **CloudLab**, Workshop, Washington, D.C. Invited talk at the Workshop for the Development of a Next-Generation Cyberinfrastructure
June	**InstaGENI Administration**, GEC #20, Davis, CA
June	**ProtoGENI Developer Topics**, GEC #20, Davis, CA
June	❖ **Apt: The Adaptable Profile-Driven Testbed**, GEC #20, Davis, CA
May	**SDN in Software**, GENI-Fed4FIRE Meeting, Cambridge, MA Invited talk at the second meeting between the US GENI project and EU Fed4FIRE project.
May	❖ **User Tool Lessons**, GENI-Fed4FIRE Meeting, Cambridge, MA Invited talk at the second meeting between the US GENI project and EU Fed4FIRE project.

April ❖ **Operational Experiences with Disk Imaging in a Multi-Tenant Datacenter**, Paper talk, Seattle, WA
Paper talk at the Symposium on Networked Systems Design and Implementation (NSDI)

March **PhantomNet: An End-to-End Mobile Wireless Testbed**, GEC #19, Atlanta, GA

2013

October **Getting Started with GENI: Part II**, Tutorial, GEC #18, Brooklyn, NY
2.5 hour hands-on tutorial presented at GEC #18 with Vic Thomas of the GENI Project Office

October **GENI Rack Operations Going Forward**, GEC #18, Brooklyn, NY

October **Flack Evolved: Jacks**, GEC #18, Brooklyn, NY

October ❖ **Overview of Federation**, GENI-Fed4FIRE Meeting, Leuven, Belgium
Invited talk at the first meeting between the US GENI project and EU Fed4FIRE project.

October **GENI Tools**, GENI-Fed4FIRE Meeting, Leuven, Belgium
Invited talk at the first meeting between the US GENI project and EU Fed4FIRE project.

October **Setting Testbed Policies**, GENI-Fed4FIRE Meeting, Leuven, Belgium
Invited talk at the first meeting between the US GENI project and EU Fed4FIRE project.

July **InstaGENI Overview**, GEC #17, Madison, WI

October **Experimentation and Instrumentation using InstaGENI Racks and GEMINI**, Tutorial, GEC #16, Salt Lake City, UT
2.5 hour hands-on tutorial presented at GEC #16 with Jim Griffioen of the University of Kentucky and Ezra Kissel of Indiana University

March **New Features in Flack**, GEC #16, Salt Lake City, UT

March **Speaks-For**, GEC #16, Salt Lake City, UT

March **InstaGENI Overview**, GEC #16, Salt Lake City, UT

2012

October **Introduction to GENI and the Experiment Lifecycle**, Tutorial, GEC #15, Houston, TX
2.5 hour hands-on tutorial presented at GEC #15

July **InstaGENI Tutorial**, Tutorial, GEC #14, Boston, MA
2 hour hands-on tutorial presented at GEC #14 with Gary Wong

July ❖ **InstaGENI Rack Update and Demo**, GEC #14, Boston, MA
Demonstration presented with Rick McGeer of HP Labs

March **ProtoGENI and Experimenters**, Workshop, Los Angeles, CA
Invited talk at the GENI Experimenters' workshop

March **ProtoGENI and ABAC**, GEC #13, Los Angeles, CA

March **❖ PRObE: Parallel Reconfigurable Observable Environment**, GEC #13, Los Angeles, CA

2011

July **Introduction to GENI using Flack and the Instrumentation Portal**, Tutorial, GEC #12, Kansas City, MO
2 hour hands-on tutorial presented at GEC #12 with Jim Griffioen of the University of Kentucky

November **Education and Support for GENI Experimenters**, GEC #12, Kansas City, MO

November **Tickets**, GEC #12, Kansas City, MO

October **❖ Managing Trust in Federated Testbeds**, Symposium, University of Tokyo, Tokyo, Japan
Invited talk at the Network Virtualization Symposium

October **❖ Emulab and ProtoGENI: Enabling Network Research and Education**, Invited talk, KDDI, Tokyo, Japan
Invited talk at the R&D organization of a Japanese telecommunications provider

July **Introduction to GENI using Flack and the Instrumentation Portal**, Tutorial, GEC #11, Denver, CO
2 hour hands-on tutorial presented at GEC #11 with Jim Griffioen of the University of Kentucky

July **ProtoGENI Control Framework Update**, GEC #11, Denver, CO

March **ProtoGENI Stitching**, GEC #10, San Juan, PR

March **ProtoGENI RSpec**, GEC #10, San Juan, PR

March **ProtoGENI Identity Management**, GEC #10, San Juan, PR

March **ProtoGENI Authorization**, GEC #10, San Juan, PR

March **ProtoGENI Control Framework Update**, GEC #10, San Juan, PR

2010

July **ProtoGENI Tutorial**, Tutorial, GEC #9, Washington, D.C.
3 hour hands-on tutorial presented at GEC #9 with Jim Griffioen of the University of Kentucky

November **ProtoGENI Status and Priorities**, GEC #9, Washington, D.C.

September **Evaluating Networked Systems**, Colloquium, Salt Lake City, UT
Invited talk in the University of Utah School of Computing Research Buffet

July **ProtoGENI and Emulab: Campus Connection Case Study**, Workshop, San Diego, CA
Invited talk at the QUILT GENI Workshop

July **ProtoGENI Tutorial**, Tutorial, GEC #8, San Diego, CA
3 hour hands-on tutorial presented at GEC #9 with Jim Griffioen of the University of Kentucky

July **Supporting ProtoGENI Users**, GEC #8, San Diego, CA

March **The ProtoGENI Vision for GENI Resource Representation**, Workshop, Durham, NC
Invited talk at the Workshop on Future of Resource Representations in GENI

March **Credentials in ProtoGENI**, GEC #7, Durham, NC

February **ProtoGENI and Emulab: Enabling Network Research and Education**, Meeting, Salt Lake City, UT
Invited talk at Internet2/ESNet Joint Techs Meeting

January **ProtoGENI and Emulab: Enabling Network Research and Education**, Meeting, Salt Lake City, UT
Invited talk at WestNet Meeting

2009

November ❖ **ProtoGENI Integrated Backbone Demonstration**, Talk and demonstration at GEC #6, Salt Lake City, UT

November **ProtoGENI Spirals 1 and 2**, GEC #6, Salt Lake City, UT

October ❖ **ProtoGENI and Emulab: Enabling Network Research and Education**, Meeting, San Antonio, TX
Invited talk at the Internet2 Fall Member Meeting

July **Federation in ProtoGENI**, Workshop, Seattle, WA
Invited talk at the Second GENI-FIRE Workshop

July **ProtoGENI Experimenter Tools**, GEC #5, Seattle, WA

July **ProtoGENI Backbone Plans and Status**, GEC #5, Seattle, WA

July **Cross-Aggregate Coordination**, GEC #5, Seattle, WA

June **Measurement and Experiment Specification**, Workshop, Madison, WI
Workshop talk at the GENI Measurement Workshop

June **The ProtoGENI RSpec**, Workshop, Chicago, IL
Talk at the GENI RSpec Workshop

April **ProtoGENI and the QUILT**, Meeting
Invited talk to the QUILT GENI Working Group

April **ProtoGENI Security Model**, GEC #4, Miami, FL

April **Vertical Integration in Emulab and ProtoGENI**, GEC #4, Miami, FL

2008

October ❖ **ProtoGENI**, GEC #3, Palo Alto, CA

June ❖ **Large-scale Virtualization in the Emulab Network Testbed**, Paper talk, Boston, MA
Paper talk at the USENIX Annual Technical Conference

March **Beyond Experiment Control: Experiment Workflow**, GEC #2, Arlington, VA

2007

October **RSpec: Resource Specification in GENI**, GEC #1, Minneapolis, MN

April ❖ **The Flexlab Approach To Realistic Evaluation of Networked Systems**, Paper talk, Cambridge, MA
Paper talk at the Symposium on Networked Systems Design and Implementation (NSDI)

2006

June **Leveraging Bloom Filters for Smart Search Within NUCA Caches**, Paper talk, Boston, MA
Paper talk at the Workshop on Complexity-Effective Design (WCED)

May **Running PlanetLab in Emulab**, Meeting, Palo Alto, CA
Invited talk at a PlanetLab meeting at HP Labs

2005

November **A Mapper for Managing Shared, Virtualized Computing and Network Resources**, San Francisco, CA
Invited talk at INFORMS, an Operations Research and Management Science conference

2004

May ❖ **Resource Mapping With `assign`**, Colloquium at Boston University, Boston, MA

2003

June ❖ **Fast, Scalable Disk Imaging with Frisbee**, Paper talk, San Antonio, TX
Paper talk at the USENIX Annual Technical Conference

August ❖ **How to Use the Emulab Public Network Testbeds**, Tutorial, Pittsburgh, PA
Full-day tutorial at SIGCOMM, presented with Jay Lepreau, Mac Newbold, and Chris Alfeld

Ellen Riloff

School of Computing
University of Utah
Salt Lake City, UT 84112

Phone: 801-581-7544
Email: riloff@cs.utah.edu
URL: www.cs.utah.edu/˜riloff

Current Research Areas: natural language processing, information extraction, sentiment analysis, semantic lexicon induction, biomedical NLP, bootstrapped learning methods for NLP.

Education

Ph.D.	University of Massachusetts at Amherst, Computer Science, 1994.
M.S.	University of Massachusetts at Amherst, Computer Science, 1989.
B.S.	Carnegie Mellon University, Applied Mathematics (Computer Science), with University Honors, 1987.

Professional Experience

2014-present	University of Utah, School of Computing, Professor.
2014-2015	University of California at Santa Cruz, Visiting Research Computer Scientist.
2000-2014	University of Utah, School of Computing, Associate Professor.
2007-2008	USC Information Sciences Institute, Visiting Scholar.
2000-2001	The Johns Hopkins University, Department of Computer Science, Visiting Associate Professor.
1994-2000	University of Utah, Department of Computer Science, Asst. Professor.

Selected Publications

1. Qadir, A., Riloff, E., and Walker, M. A., "Automatically Inferring Implicit Properties in Similes", to appear in *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, 2016.

2. Ding, H. and Riloff, E., "Acquiring Knowledge of Affective Events from Blogs using Label Propagation", *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, 2016.

3. Wagstaff, K., Riloff, E., Lanza, N., Mattmann, C., and Ramirez, P., "Creating a Mars Target Encyclopedia by Extracting Information from the Planetary Science Literature", *AAAI-16 Workshop on Knowledge Extraction from Text (KET)*, 2016.

4. Qadir, A., Riloff, E., and Walker, M. A., "Learning to Recognize Affective Polarity in Similes", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

5. Kim, Y., Riloff, E., and Hurdle, J., "A Study of Concept Extraction Across Different Types of Clinical Notes", *Proceedings of the American Medical Informatics Association (AMIA) 2015 Annual Symposium*, 2015.

6. Ding, H. and Riloff, E., "Extracting Information about Medication Use from Veterinary Discussions", *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT)*, Short Paper, 2015.

7. Qadir, A. and Riloff, E., "Learning Emotion Indicators from Tweets: Hashtags, Hashtag Patterns, and Phrases", *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Short Paper, 2014.

8. De Silva, L. and Riloff, E., "User Type Classification of Tweets with Implications for Event Recognition", *ACL 2014 Joint Workshop on Social Dynamics and Personal Attributes in Social Media*, 2014.

9. Goyal, A., Riloff, E., Daumé III, H., "A Computational Model for Plot Units", *Computational Intelligence*, 2013, Vol. 29, Issue 3, pp. 466-488.

10. Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., and Huang, R., "Sarcasm as Contrast between a Positive Sentiment and Negative Situation", *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

11. Huang, R. and Riloff, E., "Multi-faceted Event Recognition with Bootstrapped Dictionaries", *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 2013.

12. Gilbert, N. and Riloff, E., "Domain-Specific Coreference Resolution with Lexicalized Features", *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, Short Paper, 2013.

13. Huang, R. and Riloff, E., "Classifying Message Board Posts with an Extracted Lexicon of Patient Attributes", *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Short Paper, 2013.

14. Huang, R. and Riloff, E., "Modeling Textual Cohesion for Event Extraction", *Proceedings of the 26th Conference on Artificial Intelligence (AAAI)*, 2012.

15. Qadir, A. and Riloff, E., "Ensemble-based Semantic Lexicon Induction for Semantic Tagging", *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM)*, 2012, **Best Long Paper Award**.

16. Huang, R. and Riloff, E., "Bootstrapped Training of Event Extraction Classifiers", *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2012.

17. Wiebe, J. and Riloff, E., "Finding Mutual Benefit between Subjectivity Analysis and Information Extraction", *IEEE Transactions on Affective Computing*, 2011, Vol. 2, No. 4, pp. 175-191.

18. Huang, R. and Riloff, E., "Peeling Back the Layers: Detecting Event Role Fillers in Secondary Contexts", *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*, 2011.

19. Qadir, A. and Riloff, E., “Classifying Sentences as Speech Acts in Message Board Posts”, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

20. Kim, Y., Riloff, E., and Meystre, S. M., “Improving Classification of Medical Assertions in Clinical Notes”, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*, Short Paper, 2011.

21. Goyal, A., Riloff, E., Daumé III, H., “Automatically Producing Plot Unit Representations for Narrative Text”, *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2010.

22. Huang, R. and Riloff, E. “Inducing Domain-specific Semantic Class Taggers from (Almost) Nothing”, *Proceedings of The 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

23. Stoyanov, V., Cardie, C., Gilbert, N., Riloff, E., Buttler, D., and Hysom, D., “Coreference Resolution with Reconcile”, *Proceedings of The 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, Short Paper, 2010.

24. Patwardhan, S. and Riloff, E., “A Unified Model of Phrasal and Sentential Evidence for Information Extraction”, *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2009.

25. Hovy, E., and Kozareva, Z., and Riloff, E., “Toward Completeness in Concept Extraction and Classification”, *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2009.

Recent Research Funding

“Identifying Affective Events and Situations in Text”, National Science Foundation (NSF) award, PI, IIS-1450527, $149,991, 9/1/14-8/31/16.

“Open Source Indicators (OSI) Program”, subcontract to Virginia Polytechnic Institute and State University (IARPA award), PI, $120,000 (4/9/15-7/30/16), $250,000 (6/13/14-4/8/15).

“Open Source Indicators (OSI) Program”, subcontract to HRL Laboratories (IARPA award), PI, $657,789, 4/13/12-6/8/14.

“RI:Small:Acquiring Domain Knowledge from Text through Cooperative Bootstrapping”, National Science Foundation (NSF) award, PI, IIS-1018314, $383,845, 7/1/10-6/30/15.

“Text Mining Infrastructure for the Entire Biological Literature”, subcontract to USC/ISI (NSF award), PI, $119,917, 10/1/09-8/31/2013.

“RACR: Reader and Contextual Reasoner”, subcontract to IBM (DARPA award), PI, $472,112, 8/1/09-12/31/12.

"Information Extraction of Events and Beliefs from Text," subcontract to University of Pittsburgh (Department of Homeland Security (DHS) award), PI, ONR # N0014-07-1-0152, $498,200, 10/16/06-5/31/10.

Selected Invited Talks

Bootstrapped Learning of Affective Indicators in Social Media Text
- *Distinguished Speaker Seminar Series*, University of North Texas, 2015.

Sarcasm as Contrast between a Positive Sentiment and Negative Situation
- *Invited Speaker*, COGSCI 2014 Workshop: Can Cognitive Scientists Help Computers Recognize Irony?, 2014.

An NLP Voyage: Explorations with Information Extraction for Biocuration
- SciKnowMine 2013 Workshop: Bridging BioNLP and Biocuration, USC/ISI, 2013.

Finding Event Information: Multi-faceted Event Recognition and Discourse-Guided Extraction
- University of Illinios at Urbana-Champaign, 2013.

Automatically Generating Plot Unit Representations for Narrative Text
- University of California at Santa Cruz, 2013.

Adventures in Bootstrapping: Acquiring Lexical Knowledge for NLP
- Carnegie Mellon University, 2011 ; University of Texas at Austin, 2011.

Recent Awards

*SEM Best Long Paper Award, 2012: Qadir, A. and Riloff, E., "Ensemble-based Semantic Lexicon Induction for Semantic Tagging".

Honorable Mention: 2012 AAAI Classic Paper Award for AAAI 1993 paper: Riloff, E., "Automatically Constructing a Dictionary for Information Extraction Tasks".

Recent Major External Service Activities

Editorial Board, *Transactions of the Association for Computational Linguistics*, 2012-2014
Editorial Board, *Computational Linguistics*, January 2007-December 2009.
Program Co-Chair for the NAACL-HLT 2012 Conference, 2012.
Program Co-Chair for CoNLL-04 Conference, 2004.
Area Chair for Semantics, EMNLP Conference, 2011
Area Chair for Information Extraction, ACL Conference, 2009

Recent Courses Taught

Information Extraction from Text (CS 6961,6390)	2014,2016
NLP Seminar (CS 7935)	2016 (co-taught with Prof. Srikumar)
Natural Language Processing (CS 5340/6340)	2008-2013, 2015
Software Engineering Laboratory (CS 4500)	2011 (co-taught with Prof. de St. Germain)
Topics in Information Retrieval (CS 7961)	2010
Introduction to Computer Science II (CS 2420)	2009

Vivek Srikumar

School of Computing, University of Utah
50 S Central Campus Drive, Rm. 3126
Salt Lake City, Utah, 84112

Email: `svivek@cs.utah.edu`
Website: `http://svivek.com`
Phone: +1 (217) 766-2085

Experience

- *Assistant Professor*, School of Computing, University of Utah. 2014 – Present
- *Postdoctoral Scholar*, Stanford NLP group, Stanford University. 2013 – 2014
- *Postdoctoral Scholar*, Cognitive Computation Group, UIUC. 2013
- *Research Intern*, Microsoft Research. 2008 – 2008
- *Research Assistant*, UIUC. 2005 – 2013.

My research focuses on several areas in machine learning and its applications, primarily in the context of natural language processing. Some highlights include:

- **Machine learning**: (a) Learning discrete and continuous representations of unstructured data. (b) Learning to speed up structured predictors. (c) Developing machine learning frameworks for learning concepts with limited supervision.
- **Natural Language Understanding**: (a) Automatically answering reading comprehension questions by extracting entities, events and their relationships from the text. (b) Analyzing sentences for different linguistic phenomena, leading to a unified approach for predicting a semantic representation of text. (c) Applying machine learning techniques for recognizing textual entailment. (d) Detecting events and extracting their participants and attributes from news text.
- **Software for machine learning and natural language processing**: Designing programming interfaces for machine learning and NLP.

Education

Ph.D., Computer Science University of Illinois, Urbana-Champaign. 2013

Thesis: The Semantics of Role Labeling.

Doctoral Advisor: Prof. Dan Roth.

Certificate in Foundations of Teaching Center for Teaching Excellence, UIUC. 2011

Master of Science, Computer Science University of Illinois, Urbana-Champaign. 2007

Thesis: Masquerader detection through interactions.

Bachelor of Engineering, Computer Science Anna University, Chennai, India. 2005
First Class with Distinction.

Awards

- Selected as an outstanding reviewer at the annual conference of the North American Chapter of the ACL (NAACL), 2016.
- Best paper award: Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. Title: *Modeling Biological Processes for Reading Comprehension*
- Student travel scholarship, International Conference on Machine Learning (ICML), 2010.

Selected Publications

(Full list available at `http://svivek.com`.)

1. Tao Li and Vivek Srikumar. “Exploiting Sentence Similarities for Better Alignments”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2016.
2. Xingyuan Pan and Vivek Srikumar. “Expressiveness of Rectifier Networks”. In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2016.
3. Michael Tanana, Kevin A. Hallgren, Zac E. Imel, David C. Atkins, and Vivek Srikumar. “A Comparison of Natural Language Processing Methods for Automated Coding of Motivational Interviewing”. In: *Journal of Substance Abuse Treatment* (2016).
4. Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang Peter Clark, and Christopher D. Manning. “Modeling Biological Processes for Reading Comprehension”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014.
5. Vivek Srikumar and Christopher D Manning. “Learning Distributed Representations for Structured Output Prediction”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2014.
6. Gourab Kundu, Vivek Srikumar, and Dan Roth. “Margin-based Decomposed Amortized Inference”. In: *Annual meeting of the Association of Computational Linguistics (ACL)*. 2013.
7. Vivek Srikumar and Dan Roth. “Modeling Semantic Relations Expressed by Prepositions”. In: *Transactions of the Association for Computational Linguistics* vol. 1 (2013).
8. Ming-Wei Chang, Dan Goldwasser, Dan Roth, and Vivek Srikumar. “Discriminative Learning over Constrained Latent Representations”. In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2010.
9. Ming-Wei Chang, Vivek Srikumar, Dan Goldwasser, and Dan Roth. “Structured Output Learning with Indirect Supervision”. In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2010.
10. Ming-Wei Chang, Lev Ratinov, Dan Roth, and Vivek Srikumar. “Importance of Semantic Represenation: Dataless Classification”. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. 2008.

Tutorials

1. Kai-Wei Chang, Gourab Kundu, Dan Roth, and Vivek Srikumar. "Learning and Inference in Structured Prediction Models". In: *AAAI-16 Tutorial Forum*. Feb. 2016.
2. Dan Goldwasser, Vivek Srikumar, and Dan Roth. "Predicting Structures in NLP: Constrained Conditional Models and Integer Linear Programming in NLP". In: *NAACL HLT 2012 Tutorial Abstracts*. June 2012.
3. Vivek Srikumar. "Incorporating Machine Learning into your Application: Text Classification". In: *Data Sciences Summer Institute, UIUC*. 2012.
4. Vivek Srikumar. "An Introduction to Machine Learning and Natural Language Processing Tools". In: *Data Sciences Summer Institute, UIUC*. 2011.

Teaching

1. **CS 5350/6350: Machine Learning**, University of Utah. Fall 2016
   35 undergraduate students, 90 graduate students and streamed live for professional masters students.
2. **CS 7935: NLP Seminar**, University of Utah Spring 2016
   Co-taught with Ellen Riloff.
3. **CS 5350/6350: Machine Learning**, University of Utah. Fall 2015
   12 undergraduate students, 87 graduate students and streamed live for professional masters students.
4. **CS 7931: Machine Learning Seminar (Deep Learning)**, University of Utah. Fall 2015
   10 students. Co-taught with Dustin Webb.
5. **CS 5350/6350: Machine Learning**, University of Utah. Spring 2015
   Newly designed course with 66 graduate and 21 undergraduate students
6. **CS 6961: Structured Prediction**, University of Utah. Fall 2014
   Newly designed course with 20 graduate students and 1 undergraduate student

Students and Mentoring

Graduate Students Supervised (a) Xingyuan Pan, PhD, started Fall 2014. (b) Tao Li, MS thesis, expected Fall 2016, starting PhD in Fall 2016. (c) Jie Cao, PhD, started Fall 2015. (d) Nicolas Bertagnolli, MS, Spring 2016.

Undergraduate Research and Special Projects (a) Maks Cegielski-Johnson, Undergraduate honors thesis mentor, expected 2017. (b) Oliver Richardson, Undergraduate honors thesis mentor, expected 2017. (c) Tobin Yehle, Undergraduate honors thesis mentor, 2015-16. Tobin's research was supported by an award from the University's Undergraduate Research Opportunities Program.

Other Student Mentoring (a) Annie Cherkaev, 2015-16, former undergraduate, then non-matriculating student, and currently a graduate student at the University of Utah, (b) Kallie Bracken, 2015-16, former undergraduate student at the University of Utah, (c) William Guss, Summer 2015, then high school student at West High School, currently at University of California, Berkeley.

Service

External Service

- **Editorial Board**: Journal of Artificial Intelligence Research. July, 2016 – June, 2019.
- **Journal reviewer**: (a) Computational Linguistics (b) Language Resources and Evaluation (c) IEEE/ACM Transactions on Audio, Speech, and Language Processing (d) IEEE Transactions on Neural Networks and Learning Systems
- **Senior program committee member**: International Joint Conferences on Artificial Intelligence (IJCAI), 2015.
- **Program Committee Member**: (a) Twenty-Sixth, Twenty-Eighth and Twenty-Ninth, Thirtieth conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2012, 2014, 2015, 2016; (b) Annual meeting of the Association of Computational Linguistics (ACL), 2013, 2014, 2015, 2016; (c) Conference on Computational Natural Language Learning (CoNLL), 2014, 2015; (d) International Conference on Computational Linguistics (COLING), 2016; (e) Conference of the European Chapter of the ACL (EACL), 2012; (f) Conference on Empirical Methods in Natural Language Processing (EMNLP), 2012, 2014, 2015; (g) North American Chapter of the ACL (NAACL), 2010, 2012, 2013, 2015, 2016; (h) International Conference on Machine Learning (ICML), 2013,2014,2015; (i) International Joint Conferences on Artificial Intelligence (IJCAI), 2011, 2016; (j) Neural Information Processing Systems (NIPS), 2015, (k) The first AHA! Workshop on Information Discovery in Text, COLING 2014.

Workshop Organization

- Structured Prediction for Natural Language Processing, EMNLP 2016.
- Workshop on Declarative Learning Based Programming (DeLBP), AAAI 2016.

Proposal Panelist

- NSF (2016).

University of Utah Service

- School of Computing data czar: A newly created SoC position that focuses on unifying, managing and analyzing graduate student data, 2015-.
- School of Computing faculty hiring committee, 2015, 2016.
- School of Computing graduate recruitment visit coordinator, 2015, 2016.
- School of Computing admissions committee (NLP and machine learning), 2013, 2014, 2015.

Ryan Stutsman
Curriculum Vitæ

50 S. Central Campus Drive, School of Computing
Salt Lake City, UT, 84112 USA
Phone: 801.585.0913 Email: stutsman@cs.utah.edu Website: `http://www.cs.utah.edu/~stutsman/`

Research Interests

Large-scale software systems and database systems, low-latency in-memory database systems.

Education

Purdue University	West Lafayette, IN, USA	Computer Science	B.S. 2007
Stanford University	Stanford, CA, USA	Computer Science	M.S. 2010
Stanford University	Stanford, CA, USA	Computer Science	Ph.D. 2014
Microsoft Research	Redmond, WA, USA	Database Group	Postdoctoral Researcher 2013 - 2015

Professional Experience

Jul. 15 –	Assistant Professor, School of Computing, University of Utah.
Dec. 13 – Jul. 15	Postdoctoral Researcher, Database Group, Microsoft Research.
Jun. 11 – Sep. 11	Summer Intern, Memcache Group, Facebook.
Jun. 08 – Sep. 08	Summer Intern, Lawrence Livermore National Laboratory.
May 07 – Sep. 07	Summer Intern, Platforms Group, Google.

Awards

- 2013 Best Student Paper, USENIX ATC '13.
- 2007–2010 Department of Homeland Security Graduate Fellow
- 2007 Purdue University Dept. of Computer Science Outstanding Undergraduate Research Effort.
- 2007 National Science Foundation Graduate Research Fellowship Honorable Mention.
- 2007 Purdue University CERIAS Diamond Award for Outstanding Academic Achievement.
- 2007 Computing Research Assoc. Outstanding Undergraduate Award Honorable Mention.
- 2006 Outstanding Purdue College of Science Junior.
- 2006 Harry Morrison Scholarship.
- 2006 Verizon Academic Scholarship.
- 2005 Ruzika Undergraduate Summer Research. Award
- 2005 Purdue Summer Undergraduate Research Fellowship.

Grants

1. "CRII:CSR:Large-scale Systems Software Atop Scale-out In-memory Storage," sole PI, NSF, 05/2016 - 05/2018, $174,949.

Refereed Conference and Journal Publications

[1] Justin Levandoski, David Lomet, Sudipta Sengupta, Ryan Stutsman, and Rui Wang. Multi-version Range Concurrency Control in Deuteronomy. *Proceedings of the VLDB Endowment*, 8(13):2146–2157, September 2015.

[2] John Ousterhout, Arjun Gopalan, Ashish Gupta, Ankita Kejriwal, Collin Lee, Behnam Montazeri, Diego Ongaro, Seo Jin Park, Henry Qin, Mendel Rosenblum, Stephen Rumble, Ryan Stutsman, and Stephen Yang. The RAMCloud Storage System. *ACM Transactions on Computer Systems*, 33(3):7:1–7:55, August 2015.

[3] Ryan Stutsman, Collin Lee, and John Ousterhout. Experience with Rules-Based Programming for Distributed, Concurrent, Fault-Tolerant Code. In *USENIX ATC*, Santa Clara, CA, July 2015.
Presented at conference.

[4] Justin Levandoski, David Lomet, Sudipta Sengupta, Ryan Stutsman, and Rui Wang. High Performance Transactions in Deuteronomy. In *Conference on Innovative Data Systems Research (CIDR 2015)*, 2015.
Presented at conference.

[5] Asaf Cidon, Stephen M. Rumble, Ryan Stutsman, Sachin Katti, John Ousterhout, and Mendel Rosenblum. Copysets: Reducing the Frequency of Data Loss in Cloud Storage. In *Proceedings of the 2013 USENIX Conference on Annual Technical Conference*, USENIX ATC'13, pages 37–48, Berkeley, CA, USA, 2013. USENIX Association.
Best Student Paper Award.

[6] Diego Ongaro, Stephen M. Rumble, Ryan Stutsman, John Ousterhout, and Mendel Rosenblum. Fast Crash Recovery in RAMCloud. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 29–41, New York, NY, USA, 2011. ACM.
Presented at conference.

[7] John Ousterhout, Parag Agrawal, David Erickson, Christos Kozyrakis, Jacob Leverich, David Mazières, Subhasish Mitra, Aravind Narayanan, Diego Ongaro, Guru Parulkar, Mendel Rosenblum, Stephen M. Rumble, Eric Stratmann, and Ryan Stutsman. The Case for RAMCloud. *Communications of the ACM*, 54(7):121–130, July 2011.

[8] Arjun Roy, Stephen M. Rumble, Ryan Stutsman, Philip Levis, David Mazières, and Nickolai Zeldovich. Energy Management in Mobile Devices with the Cinder Operating System. In *Proceedings of the Sixth conference on Computer Systems*, EuroSys '11, pages 139–152, New York, NY, USA, 2011. ACM.
Presented at conference.

[9] Ryan Stutsman, Mikhail Atallah, Christian Grothoff, and Krista Grothoff. Lost in Just the Translation. In *Proceedings of the 2006 ACM Symposium on Applied Computing*, pages 338–345. ACM, April 2006.
Presented at conference.

Refereed Workshop Publications

[10] Mohammed Al-Mahfoudh, Ganesh Gopalakrishnan, and Ryan Stutsman. Toward Rigorous Design of Domain-Specific Distributed Systems. In *4th IEEE/ACM FME Workshop on Formal Methods in Software Engineering, FormaliSE 2016, Austin, Texas, May 15, 2016*.
To appear.

[11] Ryan Stutsman and John Ousterhout. Toward Common Patterns for Distributed, Concurrent, Fault-Tolerant Code. In *Proceedings of the 13th USENIX Conference on Hot Topics in Operating Systems*, HotOS'13, Berkeley, CA, USA, 2013. USENIX Association.
Presented at workshop.

[12] Stephen M. Rumble, Diego Ongaro, Ryan Stutsman, Mendel Rosenblum, and John K. Ousterhout. It's Time for Low Latency. In *Proceedings of the 13th USENIX Conference on Hot Topics in Operating Systems*, HotOS'11, pages 11–15, Berkeley, CA, USA, 2011. USENIX Association.

[13] Stephen M. Rumble, Ryan Stutsman, Philip Levis, David Mazières, and Nickolai Zeldovich. Apprehending Joule Thieves with Cinder. In *MobiHeld '09: Proceedings of the 1st ACM Workshop on Networking, Systems, and Applications for Mobile Handhelds*, pages 49–54, 2009.

[14] Jad Naous, Ryan Stutsman, David Mazières, Nick McKeown, and Nickolai Zeldovich. Delegating Network Security with More Information. In *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*, WREN '09, pages 19–26, 2009.

[15] Christian Grothoff, Krista Grothoff, Ludmila Alkhutova, Ryan Stutsman, and Mikhail Atallah. Translation-Based Steganography. In *Proceedings of Information Hiding Workshop*, IH 2005, pages 213–233. Springer-Verlag, 2005.

Other Publications

[16] Stephen M. Rumble, Ryan Stutsman, Philip Levis, David Mazières, and Nickolai Zeldovich. Apprehending Joule Thieves with Cinder. *SIGCOMM Computer Communication Review*, 40(1):106–111, 2010.

[17] John Ousterhout, Parag Agrawal, David Erickson, Christos Kozyrakis, Jacob Leverich, David Mazières, Subhasish Mitra, Aravind Narayanan, Guru Parulkar, Mendel Rosenblum, Stephen M. Rumble, Eric Stratmann, and Ryan Stutsman. The Case for RAMClouds: Scalable High-Performance Storage Entirely in DRAM. *SIGOPS Operating Systems Review*, 43(4):92–105, December 2009.

[18] Christian Grothoff, Krista Grothoff, Ryan Stutsman, Ludmila Alkhutova, and Mikhail J. Atallah. Translation-based Steganography. *Journal of Computer Security*, 17(3):269–303, 2009.

Service

- Department
  - School of Computing Ph.D. Admissions Committee 2014, 2015.
  - School of Computing Undergraduate Advisory Committee (UGSAC) Faculty Advisor 2015-.
- Academic Community
  - Program Committee Memberships: ACM SIGMETRICS '15, ACM SIGOPS OSR Jan. 2015, IMDM '14, IMDM '15, IMDM '16, IEEE ICDE '16, SIGMOD '16 Demo Committee, SIGMOD '17.

Open Source Software and Libraries

- RAMCloud large-scale, low-latency key-value store:
  `http://ramcloud.stanford.edu/`.
- Cinder from-scratch OS for mobile phones with first class energy management:
  `http://www.scs.stanford.edu/cinder/`.
- Rose source-to-source compiler for automated program transforms:
  `http://www.rosecompiler.org/`.
- Autotest release-driven distributed regression testing for the Linux kernel:
  `http://autotest.github.io/`.

Teaching

1/16–5/16 Instructor, CS5460/6460 Operating Systems. University of Utah. 130 students; combined undergraduate/graduates.

8/15–12/15 Instructor, CS6963 Distributed Systems. University of Utah. 19 gradudate students.

1/12–3/12 Teaching Assistant, CS244B Distributed Systems. Stanford University. 45 students.

1/08–3/08 Teaching Assistant, CS240 Advanced Operating Systems. Stanford University. 30 students.

1/07–5/07 Teaching Assistant, CS180 An Introduction to Computer Science. Purdue University. 30 students in section.

Research Advising

Doctoral Advising

Mohammed Al-Mahfoudh (Co-advised with Ganesh Gopalakrishnan), June 2015 - present.

Ph.D. Thesis Qualifying Exam Committee Member

Simone Atzeni, 2015.

Masters Thesis and Project Examining Committee Member

Charles Jabcobsen, 2016.

External Presentations

- Experience with Rules-Based Programming for Distributed, Concurrent, Fault-Tolerant Code. USENIX ATC'15.
- High Performance Transactions in Deuteronomy. CIDR 2015.
- Toward Common Patterns for Distributed, Concurrent, Fault-Tolerant Code. HotOS'13.
- Fast Crash Recovery in RAMCloud. SOSP '11.
- Energy Management in Mobile Devices with the Cinder Operating System. EuroSys '11.
- Lost in Just the Translation. ACM SAC '06.

Patents

- High Performance Transactions in Database Management Systems, 12/2014 (pending).

Media Exposure

11/11 Ars Technica, *Can DRAM replace hard drives and SSDs? RAMCloud creators say yes.*

10/11 HPC Wire, *RAMCloud: When Disks and Flash Memory are Just Too Slow.*

10/11 ZDNet, *RAMCloud puts everything in DRAM.*

March 15, 2016

Hari Sundar

Assistant Professor
School of Computing, University of Utah
50 S Central Campus Dr, Room 3190
Salt Lake City, UT 84112

Phone: 801-585-9913
email: hari@cs.utah.edu
WEB: http://www.cs.utah.edu/~hari

Research Focus

The central focus of my research is the development of computationally optimal *parallel, high-performance algorithms*, both discrete and continuous, that are efficient and scalable on state-of-the-art architectures. It is driven by applications in *biosciences* and *geophysics*, such as cardiovascular mechanics, medical image analysis, and seismic wave propagation. My research has resulted in the development of state-of-the-art distributed algorithms for *adaptive mesh refinement*, *geometric multigrid*, *fast Gauss transform* and *sorting*.

Education

2009 PhD, Bioengineering — University of Pennsylvania
Thesis: Spatio-Temporal Deformation Analysis of Cardiac MR Images.
Advisors: Christos Davatzikos & George Biros

2000 B.Engg., Control Systems — University of Delhi

Experience

2014- *Assistant Professor* — School of Computing, University of Utah
parallel and distributed algorithms, energy efficient algorithms, computational general relativity, seismic wave propagation, inverse problems, cardiac electrophysiology

2011-2014 *Research Associate* — ICES, University of Texas
multigrid methods, parallel and distributed algorithms, seismic wave propagation, distributed sorting, inverse problems

2008-2011 *Research Scientist* — Siemens Corporate Research
image registration, computer assisted surgery, image segmentation, gaze tracking, biomechanics, surgical simulation, augmented reality

2003-2008 *Research Assistant* — University of Pennsylvania
large-scale PDE constrained optimization, adaptive mesh refinement, parallel and distributed

computing, cardiac biomechanics, image registration

2002-2003 *Research Associate* Siemens Corporate Research
point cloud matching, image-based guidance for electrophysiological procedures

2000-2002 *Graduate Assistant* Rutgers University
Shape Matching, Graph Matching, Skeletal Shape Representaion

Teaching

Parallel Algorithms & High Performance Computing, Spring 2015,2016
School of Computing, University of Utah.

Big Data Computer Systems, Fall 2014,2015
School of Computing, University of Utah.

Distributed Linear Algebra, Seminar, Fall 2015,2016
School of Computing, University of Utah.

Distributed Algorithms, Seminar, Spring 2016
School of Computing, University of Utah.

Parallel Algorithms for Scientific Computing, Spring 2012, Spring 2013
University of Texas at Austin. Co-Instructed with George Biros.

Grants

1. Sponsor: *NSF*
   Role: **co-PI**
   EAGER: Application-driven Data Precision Selection Methods
   Award amount: $299,970
   Project Period: 8/1/2016 - 07/31/2018

2. Sponsor: *NSF*
   Role: **PI**
   Scalable Multigrid Algorithms for solving elliptic PDEs on power-efficient Clusters
   Grant# : 1464244
   Award amount: $175,000
   Project Period: 8/1/2015 - 07/31/2017

Gifts

Gift of two Jetson Tegra-K1 development boards from *NVIDIA* Cordporation, worth $400.

Selected Refereed Publications

Full list of publications available at Google Scholar

13. Hari Sundar, Omar Ghattas, *A Nested Partitioning Algorithm for Adaptive Meshes on Heterogeneous Clusters*, *ACM International Conference on Supercomputing, ICS'15, Newport Beach, CA*, 2015.

12. Hari Sundar, Georg Stadler, George Biros, *Comparison of Multigrid Algorithms for High-order Continuous Finite Element Discretizations*, to appear *Numerical Linear Algebra with Applications*, 2015.

11. Hari Sundar, Dhairya Malhotra, Karl Schultz, *Algorithms for High-throughput Disk-to-Disk Sorting*, Proceedings of Supercomputing (**SC13**), ACM/IEEE Computer Society, Denver.

10. Hari Sundar, Dhairya Malhotra, George Biros, *HykSort: a new variant of hypercube quicksort on distributed memory architectures*, Proceedings of ICS-13, ACM International Conference on Supercomputing, Eugene, OR.

9. David Rivest-Henault, Hari Sundar, Mohamed Cheriet, *Non-Rigid 2D/3D Registration of Coronary Artery Models with Live Fluoroscopy for Guidance of Cardiac Interventions*, IEEE Transactions on Medical Imaging, 31(8), 1557-1572.

8. Hari Sundar, George Biros, Carsten Burstedde, Johann Rudi, Omar Ghattas, Georg Stadler, *Parallel Geometric-Algebraic Multigrid on Unstructured Forests of Octrees*, Proceedings of Supercomputing (**SC12**), ACM/IEEE Computer Society, Salt Lake City.

7. Rahul Sampath, Hari Sundar, Shravan Veerapaneni, *Parallel Fast Gauss Transform*, Proceedings of Supercomputing (**SC10**), ACM/IEEE Computer Society, New Orleans. **Best Paper Finalist**.

6. Hari Sundar, Harold Litt, Dinggang Shen, *Estimating Consistent Myocardial Motion by 4D Image Warping*, Pattern Recognition, (42) 11, pp 2514-2526.

5. Hari Sundar, Rahul S. Sampath, George Biros,*Bottom-Up Construction and 2:1 Balance Refinement of Linear Octrees in Parallel*, SIAM Journal on Scientific Computing, 30(5) .

4. Hari Sundar, Christos Davatzikos, George Biros, *Biomechanically-Constrained 4D Estimation of Myocardial Motion*, Springer Lecture Notes in Computer Science, MICCAI, 5762(2): pp 257-65.

3. Rahul S. Sampath, Santi S. Adavani, Hari Sundar, Ilya Lashuk, George Biros, *Dendro: Parallel Algorithms for Multigrid and AMR Methods on 2:1 Balanced Octrees*, Proceedings of Supercomputing (**SC08**), ACM/IEEE Computer Society, Austin, Texas.

2. Hari Sundar, Dinggang Shen, George Biros, Chenyang Xu, Christos Davatzikos, *Robust estimation of Mutual Information using Spatially Adaptive Meshes*, Springer Lecture Notes in Computer Science, MICCAI, 4791(1): pp 950-58.

1. Hari Sundar, Rahul S. Sampath, Santi S. Adavani, Christos Davatzikos, George Biros, *Low-constant Parallel Algorithms for Finite Element Simulations using Linear Octrees*, Proceedings of Supercomputing (**SC07**), ACM/IEEE Computer Society, Reno, Nevada, **Best Student Paper Finalist**.

Honors and Awards

2008-	Multiple patents awarded and pending approval by the USPTO (filed by Siemens AG). Please see here for a complete list of patents.
2010	Best Paper finalist, ACM/IEEE SuperComputing 2010. (Best paper in Math Library Parallelization)
2007	Best Student Paper finalist, ACM/IEEE SuperComputing 2007. (Best paper in PDE Applications)
2003-2007	Siemens-Penn Fellowship for Ph.D. studies.

Professional Activities

- **Editorial Board** member for Journal of Computational Science (2014-present).
- **Program Committee** member for the following Conferences: ACM/IEEE Supercomputing (SC'16), IEEE Cluster 2016, 2017, ACM International Conference on Supercomputing (ICS'15), IEEE International Parallel & Distributed Processing Symposium (IPDPS'15, IPDPS'17)
- **Reviewer** for the following journals: IEEE Transactions on Medical Imaging, IEEE Transactions on Biomedical Engineering, IEEE Transactions on Image Processing, ACM Transactions on Mathematical Software, Medical Image Analysis, The Visual Computer, Signal, Image & Video Processing, .
- **Reviewer** for the following Conferences: Supercomputing, ICCV, CVPR, IPDPS, MICCAI, ISBI.
- Member on the Stampede User Advisory Committee at the Texas Advanced Computing Center (TACC), 2013-.

Last updated: October 14, 2016

James C. Sutherland

50 S. Central Campus Dr. 3290 MEB
Salt Lake City, UT 84112
James.Sutherland@utah.edu
www.che.utah.edu/∼sutherland

Academic Appointments & Affiliations

July, 2012 - Present	Associate Professor of Chemical Engineering, The University of Utah
October, 2006 - June, 2012	Assistant Professor of Chemical Engineering, The University of Utah
May, 2011 - present	Adjunct Assistant Professor, School of Computing, The University of Utah
Fall, 2010 - Present	Primary investigator, Center for Parallel Computing, The University of Utah
Fall, 2006 - Present	Primary investigator, Institute for Clean & Secure Energy, The University of Utah
August, 2004 – October, 2006	Post-doctoral research assistant – Thermal/Fluids Computational Engineering Sciences, Sandia National Laboratories (Albuquerque, NM).
August 1999 - July 2004	Ph.D. Student & Student Intern – Combustion Research Facility, Sandia National Laboratories (Livermore, CA) & The University of Utah.
1998-1999	Research Assistant - University of Utah Advanced Combustion Group.

Research Interests

- Multiscale modeling & simulation, including identification of controlling parameters from which one can construct high-fidelity, scale-bridging models.
- Extreme-scale computing, including design and implementation of software and algorithms suitable for petascale- and exascale-level computing.
- Turbulent reacting flows.

Education

May, 2004	Doctor of Philosophy in Chemical Engineering, The University of Utah GPA: 3.96 Dissertation: "Evaluation of Large-Eddy Simulation Mixing and Reaction Models Nonpremixed Combustion using Direct Numerical Simulation"
June, 1999	Bachelor of Science in Chemical Engineering, The University of Utah GPA: 3.88, Cum Laude Thesis: "A Study of the Chemistry of NO Formation and Reduction Using Methane, Carbon Monoxide, and Hydrogen as Reburning Fuels"

Peer Reviewed Publications

[1] Babak Goshayeshi and James C. Sutherland. A Comparison of Various Models in Predicting Ignition Delay in Single-Particle Coal Combustion. *Combustion and Flame*, Revision in progress.

[2] John Schmidt, Martin Berzins, Jeremy Thornock, Tony Saad, and James Sutherland. Large Scale Parallel Solution of Incompressible Flow Problems using Uintah and hypre. In *International Symposium on Cluster, Cloud and Grid Computing*, Delft, Netherlands, May 2013.

[3] Alessandro Parente and James C. Sutherland. Principal component analysis of turbulent combustion data: Data pre-processing and manifold sensitivity. *Combustion and Flame*, 160(2):340–350, February 2013.

[4] Patrick K. Notz, Roger P. Pawlowski, and James C. Sutherland. Graph-Based Software Design for Managing Complexity and Enabling Concurrency in Multiphysics PDE Software. *ACM Transactions on Mathematical Software*, 39(1):1–21, November 2012.

[5] Amir Biglari and James C. Sutherland. A filter-independent model identification technique for turbulent combustion modeling. *Combustion and Flame*, 159:1960–1970, January 2012.

[6] Martin Berzins, Qingyu Meng, John Schmidt, and James C. Sutherland. DAG-Based Software Frameworks for PDEs. In *17th International European Conference on Parallel and Distributed Computing*, Bordeaux, FR, September 2011. Springer.

[7] James C. Sutherland and Tony Saad. The Discrete Operator Approach to the Numerical Solution of Partial Differential Equations. In *20th AIAA Computational Fluid Dynamics Conference*, pages AIAA–2011–3377, Honolulu, Hawaii, June 2011.

[8] A. Parente, J. C. Sutherland, B. B. Dally, L. Tognotti, and P. J. Smith. Investigation of the MILD combustion regime via Principal Component Analysis. *Proc. Combust. Inst.*, 33(2):3333–3341, 2011.

[9] N. Punati, J. C. Sutherland, A. R. Kerstein, E. R. Hawkes, and J. H. Chen. An Evaluation of the One-Dimensional Turbulence Model: Comparison with Direct Numerical Simulations of CO/H2 Jets with Extinction and Reignition. *Proc. Combust. Inst.*, 33(1):1515–1522, 2011.

[10] J. C. Sutherland, N. Punati, and A. R. Kerstein. A Unified Approach to the Various Formulations of the One-Dimensional Turbulence Model. Technical Report ICSE091201, Institute for Clean and Secure Energy, The University of Utah, Salt Lake City, UT, 2010.

[11] A. Parente, J. C. Sutherland, B. B. Dally, L. Tognotti, and P. J. Smith. Investigation of the MILD combustion regime via Principal Component Analysis. In *Proceedings of the Australian Combustion Symposium*, pages 1–5, December 2009.

[12] A. Parente, J. C. Sutherland, P. J. Smith, and L. Tognotti. Identification of Low-Dimensional Manifolds in Turbulent Flames. In *Proc. Combust. Inst.*, volume 32, pages 1579–1586. The Combustion Institute, 2009.

[13] J. Sutherland and A. Parente. Combustion modeling using principal component analysis. *Proc. Combust. Inst.*, 32(1):1563–1570, 2009.

[14] S. P. Domino, G. J. Wagner, A. R. Black, A. Luketa-Hanlin, and J. C. Sutherland. Solution Verification for Turbulent Reacting CFD Codes. In *9th AIAA Non-Deterministic Methods Conference*, Hawaii, April 2007. AIAA.

[15] E. R. Hawkes, R. Sankaran, J. C. Sutherland, and J. H. Chen. Scalar Mixing in Direct Numerical Simulations of Temporally-Evolving Plane Jet Flames with Detailed CO/H2 Kinetics. In *Proc. Combust. Inst.*, volume 31, pages 1633–1640, 2007.

[16] J. C. Sutherland, P. J. Smith, and J. H. Chen. A Quantitative Method for A Priori Evaluation of Combustion Reaction Models. *Combust. Theory Modelling*, 11(2):287–303, 2007.

[17] J. C. Sutherland, P. J. Smith, and J. H. Chen. Quantification of Differential Diffusion in Nonpremixed Systems. *Combust. Theory Modelling*, 9(2):365–383, May 2005.

[18] E. R. Hawkes, R. Sankaran, J. C. Sutherland, and J. H. Chen. Direct Numerical Simulation of Turbulent Combustion - Fundamental Insights Towards Predictive Models. In *Journal of Physics: Conference Series*, volume 16, pages 65–79, 2005.

[19] J. C. Sutherland and C. A. Kennedy. Improved Boundary Conditions for Viscous, Reacting, Compressible Flows. *J. Comp. Phys.*, 191(2):502–524, 2003.

Book Chapters

[1] T. Echekki, A. R. Kerstein, and J. C. Sutherland. The One-Dimensional Turbulence (ODT) Model. In T. Echekki and E. Mastorakos, editors, *Turbulent Combustion Modeling: Advances, New Trends and Perspectives.*, chapter 11, pages 249–276. Springer, 2011.

Invited Talks

[1] James C. Sutherland. High Fidelity Models for Tractable Simulation of Turbulent Reacting Flows. Sandia National Laboratories, Livermore, CA, September 2013.

[2] James C. Sutherland. Scalable Multiphysics Software Design for Emerging HPC Architectures. Sandia National Laboratories, Livermore, CA, September 2013.

[3] James C Sutherland. Low-Dimensional Techniques for Modeling Turbulent Reacting Flow. Sandia National Laboratories, Albuquerque, NM, July 2012.

[4] James C. Sutherland. Software Design Paradigms for Massively Parallel Multiphysics Applications Acknowledgments. Sandia National Laboratories, Albuquerque, NM, July 2011.

[5] James C. Sutherland. Dimension Reduction in Combustion Modeling. DOE BES Combustion Contractor Meeting, Virginia, June 2011.

[6] James C Sutherland. Taming Complexity in Multiphysics Software Design Overview & Motivation. Sandia National Laboratories, Albuquerque, July 2009.

[7] James C. Sutherland. Combustion Modeling & Simulation : Challenges and Opportunities Challenges for Turbulent Combustion Modeling. In *23rd Annual ACERC Conference*, Provo, UT, 2009.

[8] James C. Sutherland and Alessandro Parente. Managing Thermochemical Complexity in CFD. In *Workshop on Fire Models & Validation*, Salt Lake City, UT, September 2007.

[9] James C. Sutherland. DNS & its Role in Validation of Mixing & Reaction Models. In *Workshop on Heat Transfer in Pool Fires*, Salt Lake City, UT, 2004.

WILLIAM B. THOMPSON

Present address:	School of Computing University of Utah 50 So. Central Campus Dr., room MEB-3190 Salt Lake City, UT 84112
phone:	(801) 581-8224
email:	thompson@cs.utah.edu
web:	http://www.cs.utah.edu/~thompson/

Education:

Ph.D., Computer Science, University of Southern California, Los Angeles, CA, January 1975.
M.S., Computer Science, University of Southern California, Los Angeles, CA, January 1972.
Sc.B., Physics, Brown University, Providence, RI, June 1970.

Professional experience:

Professor, Department of Computer Science, University of Utah, 1991 - .
Adjunct Professor, Department of Psychology, University of Utah, 2004 - .
Professor, Computer Science Department, University of Minnesota, 1990 - 1991.
Associate Professor, Department of Computer Science, University of Minnesota, 1982 - 1990.
Assistant Professor, Department of Computer Science, University of Minnesota, 1975 - 1982.

Principal areas of current research interest:

Computational models of perception, with an emphasis on spatial organization; Perception and computer graphics; Virtual environments; Computer vision;

Research accomplishments:

Prof. Thompson's current research lies at the intersection of computer graphics and visual perception, with the dual aims of making computer graphics more effective at conveying information and using computer graphics as an aid in investigating human perception. This is an intrinsically multi-disciplinary effort involving aspects of computer science, perceptual psychology, and computational vision. Prof. Thompson has also made contributions in the areas of visual motion perception and in the integration of vision and maps for navigation.

Recent professional activities:

Associate Editor, *ACM Transactions on Applied Perception*, 2003 – .

Member, Scientific Advisory Board, Max-Planck Institute for Biological Cybernetics, 2014-2019.

Chair, organizing commitee, CRA/CCC Workshop on Quantication, Communication, and Interpretation of Uncertainty in Simulation and Data Science, 2014.

Program Committee, ACM SIGGRAPH Symposium on Applied Perception 2014, 2013, 2012.

Program Committee, ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization, 2011, 2010, 2009, 2008, 2007, 2005, 2004.

Program Co-Chair, ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization, 2006.

Recent courses taught

CS 5650/6650 – Visual Perception from a Computer Graphics and Visualization Perspective

CS 6030 – Technical Communications in Computer Science

CS 7010 – Writing Research Proposals

CS 2000 – Introduction to Programming in C

Recent publications:

I.T. Ruginski, A.P. Boone, L.M. Padilla, L. Liu, N. Heydari, H.S. Kramer, M. Hegarty, W.B. Thompson, D.H. House, and S.H. Creem-Regehr, "Non-expert interpretations of hurricane forecast uncertainty visualizations," *Spatial Cognition & Computation*, 16(2), 2016.

S.H. Creem-Regehr, J.K. Stefanucci, W.B. Thompson, N. Nash, and M. McCardell, "Egocentric Distance Perception in the Oculus Rift (DK2)," *Proc. ACM Symposium on Applied Perception*, 2015.

M.N. Geuss, J.K. Stefanucci, S.H. Creem-Regehr, W.B. Thompson, and B.J. Mohler, "Effect of display technology on perceived scale of space," *Human Factors*, 57(7), 2015.

E. Jun, J.K. Stefanucci, S.H. Creem-Regehr, M. Geuss, and W.B. Thompson, "Big Foot: Using the size of a virtual foot to scale gap width," *ACM Transactions on Applied Perception*, 12(4). 2015.

G. Rauhoeft, M. Leyrer, W.B. Thompson, J.K. Stefanucci, R.L. Klatzky, and B.J. Mohler, "Evoking and Assessing Vastness in Virtual Environments," *Proc. ACM Symposium on Applied Perception*, 2015.

J.K. Stefanucci, S.H. Creem-Regehr, W.B. Thompson, D.A. Lessard, and M.N. Geuss , "Evaluating the accuracy of size perception on screen-based displays: Displayed objects appear smaller than real objects," *Journal of Experimental Psychology: Applied*, 21(3), 2015.

B.R. Kunz, S.H. Creem-Regehr, and W.B. Thompson, "Testing the Mechanisms Underlying Improved Distance Judgments in Virtual Environments," *Perception*, 44, 2015.

K.M. Rand, S.H. Creem-Regehr, and W.B. Thompson, "Spatial learning while navigating with severely degraded viewing: The role of attention and mobility monitoring," *Journal of Experimental Psychology: Human Perception and Performance*, 2015.

L.M. Padilla, G. Hansen, I.T. Ruginski, H.S. Kramer, W.B Thompson, and S.H Creem-Regehr, "The influence of different graphical displays on non-expert decision making under uncertainty," *Journal of Experimental Psychology: Applied*, 2015

S.H. Creem-Regehr, J.K. Stefanucci, and W.B. Thompson, "Perceiving Absolute Scale in Virtual Environments: How theory and application have mutually informed the role of body-based perception," in B. Ross (ed.), *The Psychology of Learning and Motivation*, vol. 63, 2015.

S.K. Satyavolu, S.H. Creem-Regehr, J.K. Stefanucci, and W.B. Thompson, “Pointing from a third person avatar location: does dynamic feedback help?” *Proc. ACM Symposium on Applied Perception*, 2014.

B.R. Kunz, S.H. Creem-Regehr, and W.B. Thompson, “Does perceptual-motor calibration generalize across two different forms of locomotion? Investigations of walking and wheelchairs,” *PLoS ONE*, 8(2), 2013.

M.N. Geuss, J.K. Stefanucci, S.H. Creem-Regehr, and W.B. Thompson, “Effect of viewing plane on perceived distances in real and virtual environments,” *Journal of Experimental Psychology: Human Perception and Performance*, 38(5), 2012.

M. Raj, S.H. Creem-Regehr, K.M. Rand, J.K. Stefanucci, and W.B. Thompson, “Kinect based 3D Object Manipulation on a desktop display,” *Proc. ACM Symposium on Applied Perception*, 2012.

J.K. Stefanucci, D.A. Lessard, M.N Geuss, S.H. Creem-Regehr, and W.B. Thompson, “Evaluating the accuracy of size perception in real and virtual environments,” *Proc. ACM Symposium on Applied Perception*, 2012.

K. Rand, M.R. Tarampi, S.H. Creem-Regehr, and W.B. Thompson, “The influence of ground contact and visible horizon on perception of distance and size under severely degraded vision,” *Seeing and Perceiving*, 25(5), 2012.

T. Ziemek, S.H. Creem-Regehr, W.B. Thompson, and R. Whitaker, “Evaluating the Effectiveness of Orientation Indicators with an Awareness of Individual Differences,” *ACM Transactions on Applied Perception*, 9(2), 2012.

K. Rand, M.R. Tarampi, S.H Creem-Regehr, and W.B. Thompson, “The Importance of a Visual Horizon for Distance Judgments under Severely Degraded Vision,” *Perception*, 40(2), 2011.

W.B. Thompson, R.W. Fleming, S.H. Creem-Regehr, and J.K. Stefanucci, *Visual Perception from a Computer Graphics Perspective*, CRC Press, 2011.

M. Geuss, J. Stefanucci, S.H. Creem-Regehr, and W.B. Thompson, “Can I pass?: Using affordances to measure perceived size in virtual environments,” *Proc. Symposium on Applied Perception in Graphics and Visualization*, 2010.

B.R. Kunz, S.H. Creem-Regehr, and W.B. Thompson, “Visual capture influences body-based indications of visual extent,” *Experimental Brain Research*, 207(3-4), 2010.

B.J Mohler, S.H Creem-Regehr, W.B. Thompson, and H.H. Bülthoff, “The Effect of Viewing a Self-Avatar on Distance Judgments in an HMD-Based Virtual Environment,” *Presence: Teleoperators and Virtual Environments*, 19(3), 2010.

M.R. Tarampi, S.H. Creem-Regehr, and W.B. Thompson, “Intact Spatial Updating with Severely Degraded Vision,” *Attention, Perception, & Psychophysics*, 72(1), 2010.

W.B. Thompson and P. Shirley, “Computer Graphics and Perception,” in *Encyclopedia of Perception*, E.B. Goldstein, ed., Sage, 2010.

W.B. Thompson, “Virtual Reality: Vision,” in *Encyclopedia of Perception*, E.B. Goldstein, ed., Sage, 2010.

M. Bratkova, P. Shirley, and W.B. Thompson, “Artistic Rendering of Mountainous Terrain,” *ACM Transactions on Graphics*, 28(4), 2009.

S.A. Kuhl, W.B. Thompson, and S.H. Creem-Regehr, “HMD calibration and its effects on distance judgments,” *ACM Transactions on Applied Perception*, 6(3), 2009.

B.R. Kunz, L. Wouters, D. Smith, W.B. Thompson, and and S.H. Creem-Regehr, "Revisiting the Effect of Quality of Graphics on Distance Judgments in Virtual Environments: A Comparison of Verbal Reports and Blind Walking, *Attention, Perception, & Psychophysics*, 71(6), 2009.

B.R. Kunz, and S.H. Creem-Regehr, and W.B. Thompson, "Evidence for Motor Simulation in Imagined Locomotion," *Journal of Experimental Psychology: Human Perception and Performance*, 35(5), 2009.

P. Willemsen, M.B. Colton, S.H. Creem-Regehr and W.B. Thompson, "The Effects of Head-Mounted Display Mechanical Properties and Field-of-View on Distance Judgments in Virtual Environments," *ACM Transactions on Applied Perception*, 6(2), 2009.

M. Bratkova, W.B. Thompson, and P. Shirley, "Automatic Views of Natural Scenes," *Proc. International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, May 2009.

P. Shirley and S. Marshcner, with M. Ashikhmin, M. Gleicher, N. Hoffman, G. Johnson, T. Munzner, E. Reinhard, K. Sung, W.B. Thompson, P. Willemsen, and B. Wyvill, *Fundamentals of Computer Graphics*, third edition, A K Peters, 2009 (chapter on Visual Perception).

Jacobus (Kobus) Van der Merwe

University of Utah
School of Computing
50 South Central Campus Drive
Salt Lake City, UT, 84112

Phone: (801) 581-3012
Email: `kobus@cs.utah.edu`
Homepage: `http://www.cs.utah.edu/~kobus`

Education

PhD in Computer Science, 1998
University of Cambridge, United Kingdom
Dissertation: Open Service Support for ATM

M.Eng in Electronic Engineering (cum laude), 1990
University of Pretoria, South Africa
Dissertation: A study of the effect of out-of-plane rotation on the performance of smart spatial filters

B.Eng in Electronic Engineering (cum laude), 1988
University of Pretoria, South Africa

Recognition and honors

Received the USENIX Test of Time award for "developing a logically centralized BGP routing controller, which was an important step towards the centralized routing controllers of Software-Defined Networks", 2015

Received AT&T Research Excellence award, 2001.

Received AT&T Science and Technology Medal, 2010, for work in Intelligent Route Control.

Recent Work Experience

August 2012 - Present *Associate Professor* University of Utah

October 2011 - July 2012 *Lead Member Technical Staff* AT&T Labs - Research

April 2001 - October 2011 *Principal Member Technical Staff* AT&T Labs - Research

January 1998 - March 2001 *Senior Member Technical Staff* AT&T Labs - Research

Teaching Experience

Fall 2016 CS 6480: Advanced Computer Networks

Spring 2016 CS7943: Networking Seminar

Spring 2016 CS 4480: Computer Networks

Fall 2015 CS 6480: Advanced Computer Networks

Spring 2015 CS7943: Networking Seminar

Spring 2015 CS 4480: Computer Networks

Spring 2015 CS 3011: Undergraduate Industry Forum

Fall 2014 CS 6480: Advanced Computer Networks

Spring 2014 CS 4480: Computer Networks

Fall 2013 CS 6480: Advanced Computer Networks

Spring 2013 CS 4480: Computer Networks

Fall 2012 CS 7934: Computer Systems Seminar

Recent Publications

Student authors below underlined. University of Utah students *underlined and italicized*.

Book Chapters

1. H. A. Alzoubi, M. Rabinovich, S. Lee, J. Van Der Merwe, and O. Spatscheck, "Anycast Request Routing for Content Delivery Networks". An invited chapter in Mukaddim Pathan and Ramesh Sitaraman (Eds.). "Advanced Content Delivery and Streaming in the Cloud". Wiley Publishers, 2014.
2. Brian Rexroad and Jacobus Van der Merwe, "Network Security - A Service Provider View", in "*Guide to Reliable Internet Services and Applications (Computer Communications and Networks)* ", Charles R Kalmanek, Sudip Misra, and Y. Richard Yang (ed.), 2012.

Journal Papers

1. Timothy Wood, K.K. Ramakrishnan, Prashant Shenoy, Jacobus Van der Merwe, Jinho Hwang, Guyue Liu, Lucas Chaufournier, "CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines", IEEE/ACM Transactions on Networking, August 2014.
2. Elliott Karpilovsky, Matthew Caesar, Jennifer Rexford, Aman Shaikh, Jacobus van der Merwe, "Practical Network-Wide Compression of IP Routing Tables", IEEE Transactions on Network and Service Management, November 2012
3. Timothy Wood, K.K. Ramakrishnan, Prashant Shenoy and Jacobus Van der Merwe, "Enterprise-Ready Virtual Cloud Pools: Vision, Opportunities and Challenges", The Computer Journal, June 2012; doi: 10.1093/comjnl/bxs060
4. Hussein Alzoubi, Seungjoon Lee, Michael Rabinovich, Oliver Spatscheck, and Jacobus Van Der Merwe, "A Practical Architecture for an Anycast CDN", ACM Transactions on Web, October 2011, Volume 5, Number 4

Peer Reviewed Conference and Workshop Papers

1. *Josh Kunz*, *Christopher Becker*, *Mohamed Mehdi Jamshidy*, Sneha Kumar Kasera, Robert Ricci, and Jacobus Van der Merwe, "OpenEdge: A Dynamic and Secure Open Service Edge Network", IEEE/IFIP Network Operations and Management Symposium (NOMS), April, 2016.

2. *Ren Quinn*, *Josh Kunz*, *Aisha Syed*, Joe Breen, Sneha Kumar Kasera, Robert Ricci, and Jacobus Van der Merwe, "KnowNet: Towards a Knowledge Plane for Enterprise Network Management", IEEE/IFIP Network Operations and Management Symposium (NOMS), April, 2016.

3. *Ryan Saunders*, *Junguk Cho*, *Arijit Banerjee*, *Frederico Rocha*, and Jacobus Van der Merwe, "P2P Offloading in Mobile Networks using SDN", Symposium on SDN Research (SOSR), March 2016.

4. *Arijit Banerjee*, Rajesh Mahindra, Karthik Sundaresan, Sneha Kumar Kasera, Jacobus Van der Merwe, and Sampath Rangarajan, "Scaling the LTE Control-Plane for Future Mobile Access", Proceedings of the Eleventh ACM International Conference on Emerging Networking EXperiments and Technologies (CoNEXT), December 2015.

5. *Binh Nguyen*, Zihui Ge, Jacobus Van der Merwe, He Yan, and Jennifer Yates, "ABSENCE: Usage-based Failure Detection in Mobile Networks", Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom), September 2015.

6. *Kaiqiang Wang*, *Minwei Shen*, *Junguk Cho*, *Arijit Banerjee*, Jacobus Van der Merwe, and Kirk Webb, "MobiScud: A Fast Moving Personal Cloud in the Mobile Network", 5th Workshop on All Things Cellular: Operations, Applications and Challenges, August 2015.

7. *Cai (Richard) Li*, *Dallin Abendroth*, *Xing Lin*, *Yuankai Guo*, *Hyun-wook Baek*, Eric Eide, Robert Ricci, and Jacobus Van der Merwe, "Potassium: Penetration Testing as a Service", ACM Symposium on Cloud Computing (SoCC), August 2015.

8. *Arijit Banerjee*, *Binh Nguyen*, Vijay Gopalakrishnan, Sneha Kumar Kasera, Seungjoon Lee, and Jacobus (Kobus) Van der Merwe, "Efficient, Adaptive and Scalable Device Activation for M2M Communications", IEEE International Conference on Sensing, Communication and Networking (SECON), June 2015.

9. *Junguk Cho*, *Binh Nguyen*, *Arijit Banerjee*, Robert Ricci, Jacobus Van der Merwe, and Kirk Webb,"SMORE: Software-Defined Networking Mobile Offloading Architecture", SIGCOMM Workshop on All Things Cellular: Operations, Applications and Challenges, August 2014.

10. *Binh Nguyen*, *Arijit Banerjee*, Vijay Gopalakrishnan, Sneha Kasera, Seungjoon Lee, Aman Shaikh, and Jacobus Van der Merwe, "Towards understanding TCP performance on LTE/EPC mobile networks", SIGCOMM Workshop on All Things Cellular: Operations, Applications and Challenges, August 2014.

11. *Hyun-wook Baek*, Abhinav Srivastava and Jacobus Van der Merwe, "CloudVMI: Virtual Machine Introspection as a Cloud Service", IEEE International Conference on Cloud Engineering (IC2E), March, 2014.

12. *Arijit Banerjee*, Xu Chen, Jeffrey Erman, Vijay Gopalakrishnan, Seungjoon Lee, and Jacobus Van der Merwe, "MOCA: A Lightweight Mobile Cloud Offloading Architecture", ACM Workshop on Mobility in the Evolving Internet Architecture (MobiArch), October, 2013.

13. Chia-Chi Lin, Virajith Jalaparti, Matthew Caesar, and Jacobus Van der Merwe, "DEFINED: Deterministic Execution for Interactive Control-Plane Debugging", USENIX Annual Technical Conference (ATC), June 2013.

14. Xu Chen, Jeffrey Erman, Seungjoon Lee, Jacobus Van der Merwe, “Mercado: Using market principles to drive alternative network service abstractions”, Capacity Sharing Workshop, ACM CoNEXT 2012, December, 2012

15. Virajith Jalaparti, Matthew Caesar, Seungjoon Lee, Jeffrey Pang, Jacobus van der Merwe, “SMOG: A Cloud Platform for Seamless Wide area Migration of Networked Games”, ACM/IEEE NetGames, November 2012.

16. Changbin Liu, Yun Mao, Xu Chen, Mary F. Fernandez, Boon Thau Loo and Jacobus E. Van der Merwe, "TROPIC: Transactional Resource Orchestration Platform In the Cloud", USENIX Annual Technical Conference (ATC), June 2012

Recent Funding

Co-PI, $499,999 (*50% share*), NSF, *CapNet: Secure Scientific Workloads with Capability Enabled Networks*, with Anton Burtsev, September 2015 - August 2018.

PI, $49,623 (*100% share*), NSF, *PhantomNet Users' Workshop*, September 2014 - August 31 2016.

Co-PI, $3,400,000 (including $1,000,000 University cost share) (*33% share*), NSF, *MRI: Development of Apt, a Testbed Instrument With Adaptable Profiles for Network and Computational Science*, with Robert Ricci (PI), Eric Eide and Julio Facelli, October 2013 - September 2017.

Co-PI, $999,978 (*20% share*), NSF, *CC-NIE Integration: Science Slices Converting Network Research Innovation into Enhanced Capability for Computational Science and Engineering at the University of Utah*, with Steven Corbato (PI), Adam Bolton, Thomas Cheatham and Robert Ricci, September 2013 - August 2016.

PI, $298,716 (*100% share*), NSF, *US Ignite EAGER: SeaCat: An SDN End-to-end Application Containment ArchitecTure to enable Secure Role Based Network Access in Healthcare*, September 2013 - August 2016.

PI, $1,140,747 (*25% share*), NSF, *NeTS: Medium: KnowOps - Making Network Management and Operations Software Defined*, with Sneha Kasera, Robert Ricci and Suresh Venkatasubramanian, September 2013 - August 2016. (Note: This was awarded by NSF as a “continuing grant” with $402,392, $364,187 and $374,167 for each year of the award.)

PI, $999,991 (*25% share*), NSF, *TWC: Medium: Collaborative: TCloud: A Self-Defending, Self-Evolving and Self-Accounting Trustworthy Cloud Platform*, with Eric Eide, Feifei Li and Robert Ricci, September 2013 - August 2016.

PI, $1,998,482 (*50% share*), NSF, *CI-ADDO-NEW: PhantomNet: An End-to-End Mobile Network Testbed*, with Robert Ricci, September 2013 - August 2016.

Last updated: October 25, 2016

Bei Wang

Assistant Professor

School of Computing, Scientific Computing and Imaging Institute
University of Utah
72 S Central Campus Drive, Salt Lake City, UT 84112
beiwang@sci.utah.edu
http://www.sci.utah.edu/~beiwang/

Education

2010	Ph.D. in Computer Science, Duke University
2010	Certificate in Computational Biology and Bioinformatics, Duke University
2003	B.S. in Computer Science and Mathematics, Minor in Psychology Summa Cum Laude, University of Bridgeport

Professional Experience

2016 – Present	Assistant Professor, School of Computing, Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT
2011 – 2016	Research Computer Scientist, Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT
2010 – 2011	Postdoctoral Fellow, Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT
2009 – 2010	Visiting Researcher, Institute of Science and Technology, Austria

Research[1]

Current Research Topics

- Data analysis and data visualization of large and complex data, in particular: stratification learning and structural inference of point cloud data, robust feature extraction from vector field data, visual analytics of high-dimensional data, large-scale network analysis and visualization, topology-inspired machine learning, multivariate and uncertainty analysis via category theory.

Selected Journal Publications / Book Chapters

(J1) Critical Point Cancellation in 3D Vector Fields: Robustness and Discussion. Primoz Skraba, Paul Rosen, Bei Wang, Guoning Chen, Harsh Bhatia and Valerio Pascucci. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 2016.

(J2) Interstitial and Interlayer Ion Diffusion Geometry Extraction in Graphitic Nanosphere Battery Materials. Attila Gyulassy, Aaron Knoll, Peer-Timo Bremer, Bei Wang, Kah Chun Lau, Michael Papka, Larry Curtiss, and Valerio Pascucci. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 22(1), pages 916 - 925, 2016.

(J3) Grassmannian Atlas: A General Framework for Exploring Linear Projections of High-Dimensional Data. Shusen Liu, Peer-Timo Bremer, Jayaraman J. Thiagarajan, Bei Wang, Brian Summa and Valerio Pascucci. Computer Graphics Forum (CGF), to appear, 2016.

(J4) Analyzing Simulation-Based PRA Data Through Traditional and Topological Clustering: A BWR Station Blackout Case Study. Dan Maljovec, Shusen Liu, Bei Wang, Valerio Pascucci, Peer-Timo Bremer, Diego Mandelli and Curtis Smith. *Reliability Engineering & System Safety (RESS)*, 145, pages 262-276, 2016.

(J5) Robustness-Based Simplification of 2D Steady and Unsteady Vector Fields. Primoz Skraba, Bei Wang, Guoning Chen and Paul Rosen. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 21(8), pages 930 - 944, 2015.

[1] Utah students underlined. Authors ordered alphabetically are marked with an **a**.

(J6) Local, Smooth, and Consistent Jacobi Set Simplification. Harsh Bhatia, Bei Wang, Gregory Norgard, Valerio Pascucci and Peer-Timo Bremer. *Computational Geometry: Theory and Applications (CGTA)*, 48(4), Pages 311-332, 2015.

(J7) ND2AV: N-Dimensional Data Analysis and Visualization – Analysis for the National Ignition Campaign. Peer-Timo Bremer, Dan Maljovec, Avishek Saha, Bei Wang, Jim Gaffney, Brian K. Spears and Valerio Pascucci. *Computing and Visualization in Science (CVS)*, 17(1), pages 1- 18, 2015.

(J8) Visual Exploration of High-Dimensional Data through Subspace Analysis and Dynamic Projections. Shusen Liu, Bei Wang, Jayaraman J. Thiagarajan, Peer-Timo Bremer and Valerio Pascucci. *Computer Graphics Forum (CGF)*, 34(3), pages 271-280, 2015.

(J9) Distortion-Guided Structure-Driven Interactive Exploration of High-Dimensional Data. Shusen Liu, Bei Wang, Peer-Timo Bremer and Valerio Pascucci. *Computer Graphics Forum (CGF)*, 33(3), pages 101-110, 2014.

(J10) Interpreting Feature Tracking Through the Lens of Robustness.[a] Primoz Skraba and Bei Wang. *Topological Methods in Data Analysis and Visualization III: Theory, Algorithms, and Applications*, pages 19-38, 2014.

(J11) Visualizing Robustness of Critical Points for 2D Time-Varying Vector Fields. Bei Wang, Paul Rosen, Primoz Skraba, Harsh Bhatia and Valerio Pascucci. *Computer Graphics Forum (CGF)*, 32(2), pages 221-230, 2013.

(J12) Adaptive Sampling with Topological Scores. Dan Maljovec, Bei Wang, Ana Kupresanin, Gardard Johannesson, Valerio Pascucci, Peer-Timo Bremer. *International Journal for Uncertainty Quantification (IJUQ)*, 3(2), pages 119-141, 2013.

(J13) Branching and Circular Features in High Dimensional Data. Bei Wang, Brian Summa, Valerio Pascucci and Mikael Vejdemo-Johansson. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 17(12), pages 1902-1911, 2011.

(J14) Computing Elevation Maxima by Searching the Gauss Sphere. Bei Wang, Herbert Edelsbrunner and Dmitriy Morozov. *ACM Journal of Experimental Algorithmics (JEA)*, 16, pages 1-13, 2011.

(J15) A Computational Screen for Site Selective A-to-I Editing Detects Novel Sites in Neuron Specific Hu Proteins. Mats Ensterö, Örjan Åkerborg, Daniel Lundin, Bei Wang, Terrence S Furey, Marie Öhman and Jens Lagergren. *BMC Bioinformatics*, 11(6), 2010.

Selected Conference Publications

(C1) Convergence between Categorical Representations of Reeb Space and Mapper.[a] Elizabeth Munch and Bei Wang. International Symposium on Computational Geometry (SOCG), 2016.

(C2) Kernel Partial Least Squares Regression for Relating Functional Brian Network Topology to Clinical Measures of Behavior. Eleanor Wong, Sourabh Palande, Bei Wang, Brandon Zielinski, Jeffrey Anderson and P. Thomas Fletcher. International Symposium on Biomedical Imaging (ISBI), 2016.

(C3) Exploring Persistent Local Homology in Topological Data Analysis.[a] Brittany T. Fasy and Bei Wang. Special session on Topological Methods in Data Science and Analysis, IEEE International Conference on Acoustics, Speech and Signal Process (ICASSP), 2016.

(C4) Critical Point Cancellation in 3D Vector Fields: Robustness and Discussion. Primoz Skraba, Paul Rosen, Bei Wang, Guoning Chen, Harsh Bhatia and Valerio Pascucci. *Proceedings IEEE Pacific Visualization (PacificVis)*, 2016. **Best Paper Award.**

(C5) Topology-Inspired Partition-Based Sensitivity Analysis and Visualization of Nuclear Simulations. Daniel Maljovec, Bei Wang, Paul Rosen, Andrea Alfonsi, Giovanni Pastore, Cristian Rabiti and Valerio Pascucci. *Proceedings IEEE Pacific Visualization (PacificVis)*, 2016.

(C6) Geometric Inference on Kernel Density Estimates.[a] Jeff M. Phillips, Bei Wang and Yan Zheng. *International Symposium on Computational Geometry (SOCG)*, 2015.

(C7) Visualizing High-Dimensional Data: Advances in the Past Decade. Shusen Liu, Dan Maljovec, Bei Wang, Peer-Timo Bremer and Valerio Pascucci. *Eurographics Conference on Visualization (EuroVis)*, State-of-the-Art Report (STAR), 2015.

(C8) Approximating Local Homology from Samples.[a] Primoz Skraba and Bei Wang. *Proceedings 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 174-192, 2014.

(C9) 2D Vector Field Simplification Based on Robustness. Primoz Skraba, Bei Wang, Guoning Chen and Paul Rosen. *IEEE Pacific Visualization (PacificVis)*, 2014. **Best Paper Award.**

(C10) Multivariate Volume Visualization through Dynamic Projections. Shusen Liu, Bei Wang, Jayaraman J. Thiagarajan, Peer-Timo Bremer and Valerio Pascucci. *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, 2014.

(C11) Adaptive Sampling Algorithms for Probabilistic Risk Assessment of Nuclear Simulations. Dan Maljovec, Bei Wang, Diego Mandelli, Peer-Timo Bremer and Valerio Pascucci. *International Topical Meeting on Probabilistic Safety Assessment and Analysis (PSA)*, 2013. **First runner-up for the Best Student Paper Award.**

(C12) Local Homology Transfer and Stratification Learning. Paul Bendich, Bei Wang and Sayan Mukherjee. *Proceedings 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1355-1370, 2012.

(C13) Topological Analysis and Visualization of Cyclical Behavior in Memory Reference Traces. A.N.M. Imroz Choudhury, Bei Wang, Paul Rosen and Valerio Pascucci. *IEEE Pacific Visualization (PacificVis)*, 2012.

(C14) Computing Elevation Maxima by Searching the Gauss Sphere.* Bei Wang, Herbert Edelsbrunner and Dmitriy Morozov. *Proceedings 13th International Symposium on Experimental Algorithms (SEA)*, 2009. *Lecture Notes in Computer Science (LNCS)*, 5526, pages 281-292, 2009.

Funding

- NSF IIS-1513616 (2015-2019).
  III: Medium: Collaborative Research: Topological Data Analysis for Large Network Visualization.
  Role: **Principal Investigator**. Total award amount: $761,067.
  The project investigates scalable, structure-preserving ways to sparsify networks making them suitable for visual exploration. The proposed research will help in understanding large, complex network data sources, which is highly relevant and important in application areas including brain connectomics, epidemiology, communication, law enforcement, public policy and marketing.

- NSF and National Radio Astronomy Observatory (NRAO) (2016-2017)
  Feature Extraction and Visualization of ALMA Data Cubes through Topological Data Analysis.
  Role: **Co-PI**. Total award amount: $185,133.
  The project focuses on designing effective visualizations for exploring the full range of complex features present in large Atacama Large Millimeter Array (ALMA) data cubes. The proposed research aims to provide for the first time, tools capable of simultaneously visualizing, comparing and analyzing the dozens to hundreds of data cubes and to improve the analysis pipeline for NRAO scientists.

- Idaho National Lab (INL) LDRD Project, INL contract No. 00158804 (2015).
  INL Reliability Analysis Using Topological Decomposition.
  Role: **Co-PI**. Total award amount: $100,076.
  The project focuses on developing topology-based sensitivity analysis tools for nuclear scientists.

Selected Invited Talks

- Topology, Geometry, and Data Analysis Conference at Ohio State University, 2016.
- Pacific Northwest National Laboratory, 2015.
- SAMSI workshop on Topological Data Analysis, 2014.
- Computer Science Department, Ohio State University, 2014.
- Computer Science Department Colloquium, University of Connecticut, 2013.
- Colloquium Series in School of Engineering, University of Bridgeport, 2013.
- IMA Workshop on Modern Applications of Homology and Cohomology, 2013.
- SIAM Conference on Applied Algebraic Geometry (AG), 2013.
- AMS-MAA Joint Mathematics Meeting (JMM), 2012.
- Theory Lunch, School of Computer Science, Carnegie Mellon University, 2012.
- Yaroslavl International Conference Discrete Geometry Dedicated to Centenary of A.D.Alexandrov, 2012.
- Summer school of the Delaunay Laboratory, Russia, 2012.
- ACM Symposium on Computational Geometry (SOCG) Workshop on Computational Topology, 2012.
- Fields Institute for Research in Mathematical Sciences, Thematic Program on Discrete Geometry and Applications, Workshop on Computational Topology, 2011.

Selected Awards

- Best Paper Award at IEEE Pacific Visualization (PacificVis), 2016.
- Best Paper Award at IEEE Pacific Visualization (PacificVis), 2014.
- First runner-up for the Best Student Paper Award at International Topical Meeting on Probabilistic Safety Assessment and Analysis (PSA), 2013.

Teaching

- Spring 2016: CS 1060 - Explorations in Computer Science (undergraduate level)
- Spring 2016: CS 4960 - Introduction to Computational Geometry (undergraduate level)
- Fall 2015: CS 6210 - Advanced Scientific Computing I (graduate level)

External Service

Program Committees

- IEEE Symposium on Large Data Analysis and Visualization (LDAV), 2016.
- EG/VGTC Conference on Visualization (EuroVis), Short Paper Track, 2016.
- Topology-Based Methods in Visualization (TopoInVis), 2015.

Journal/Conference Reviewing

- Discrete & Computational Geometry (DCG), Computational Geometry Theory and Applications (CGTA), International Journal of Computational Geometry & Applications (IJCGA), Journal of Computational Geometry (JoCG), IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), IEEE Transactions on Visualization and Computer Graphics (TVCG), Book chapter for AMS short course in Joint Math Meetings (JMM). ACM Symposium on Theory of Computing (STOC), ACM-SIAM Symposium on Discrete Algorithms (SODA), (ACM) Symposium on Computational Geometry (SOCG), European Symposium on Algorithms (ESA), SIAM Algorithm Engineering and Experiments (ALENEX), IEEE Conference on Visualization (VIS), Eurographics Conference on Visualization (EuroVis), Topology-Based Methods in Visualization (TopoInVis), IEEE Symposium on Large Data Analysis and Visualization (LDAV).

Other Synergistic Activities

- Founding member of Women in Visualization Mentoring Network, 2015.
- Member of Women in Computational Topology Network, 2015.
- Member and Speaker, Applied Algebraic Topology Research Network, 2014-2015.
- Workshop Organizer and Speaker: Topological Data Analysis and Visualization for Large-Scale and High-Dimensional Science Discovery. International Topical Meeting on Probabilistic Safety Assessment and Analysis (PSA), 2013.

Students

Student Supervising

- Sourabh Palande (PhD).

Student Research Project Mentoring

- Brian Summa (PhD), Harsh Bhatia (PhD), Yan Zheng (PhD), Hoa Nguyen (PhD), Wathsala Widanagamaachchi (PhD), Liang He (MS), Soumya S. Mishra (MS), Sam Leventhal (PhD).

PhD Committee

- Shusen Liu (PhD), Dan Maljovec (PhD).

BIOGRAPHICAL SKETCH

NAME Weiss, Jeffrey A.	POSITION TITLE Professor, Department of Bioengineering Adjunct Professor, Department of Orthopedics and School of Computing
eRA COMMONS USER NAME jaweiss	

EDUCATION/TRAINING *(Begin with baccalaureate or other initial professional education, such as nursing, and include postdoctoral training.)*

INSTITUTION AND LOCATION	DEGREE *(if applicable)*	YEAR(s)	FIELD OF STUDY
University of California San Diego, La Jolla, CA	B.S.	1989	Bioengineering
University of California San Diego, La Jolla, CA	M.S.	1990	Bioengineering
University of Utah, Salt Lake City, UT	Ph.D.	1994	Bioengineering
Lawrence Livermore National Laboratory, Livermore, CA	Post-Doc	1994-1996	Computational Biomechanics

A. Personal Statement

I have been active as a scientist in the mechanics of musculoskeletal and cardiovascular soft tissues for nearly 25 years. The structure and function of normal and healing collagenous connective tissues has been a major focus of my research program. I am an expert in experimental biomechanics, computational biomechanics and nonlinear finite element analysis. I have a strong background in materials testing and characterization, multiple imaging techniques including CT, MRI and multiphoton microscopy, and the histochemical, biochemical, and molecular biological methods that will be applied in the proposed research. My laboratory develops and maintains FEBio (Finite Elements for Biomechanics), an open-source finite element software suite for computational analyses in biomechanics and biophysics with over 6,000 registered users (www.febio.org).

Professor Yu and I began our current collaboration in 2014, shortly after Professor Yu joined the Department of Bioengineering at the University of Utah. The proposed research is based on our collaborative efforts to apply CHP targeting of collagen damage to the study of injury and disease in musculoskeletal soft tissues. We have collected convincing preliminary data over the last two years to support the project. The project brings together my expertise in the structure, function, injury and healing of musculoskeletal soft tissues with Professor Yu's expertise in bio-organic chemistry, polymer science and protein/peptide engineering.

$^{+}$Zitnay JL, $^{+}$Li Y, Qin Z, San B-H, Depalle B, Reese SP, Buehler MJ, *Yu SM, *Weiss JA: Molecular level detection and localization of mechanical damage in collagen enabled by collagen hybridizing peptides. *Nature Communications*, Revision 1 Submitted June 2016. + = co-first authors; * = co-corresponding authors.

Henninger HB, Valdez WR, Scott SA, Weiss JA: Elastin governs the mechanical response of medial collateral ligament under shear and transverse tensile loading. *Acta Biomaterialia* 24:304-312, 2015. PMID: 26162584.

Reese SP, Ellis BJ, Weiss JA: Micromechanical model of a surrogate for collagenous soft tissues: development, validation, and analysis of mesoscale size effects. *Biomechanics and Modeling in Mechanobiology* 12(6):1195-1204, 2013. PMCID: 3676693.

Maas SA, Ellis BJ, Ateshian GA, Weiss JA: FEBio: Finite elements for biomechanics. *Journal of Biomechanical Engineering* 134(1):011005, 2012. PMCID: 3705975.

B. Positions and Honors

Positions and Employment

10/1994 - 11/1996	Postdoctoral Scientist, Applied Mechanics Group, LLNL, Livermore, California
12/1994 - 10/1998	Research Assistant Professor, Department of Bioengineering, University of Utah
10/1997 - 10/1998	Institute Director, Orthopedic Biomechanics Institute, Salt lake City, Utah
10/1998 - 9/2000	Assistant Professor, Biomedical Engineering, University of Arizona
10/1998 - 9/2000	Assistant Professor, Mechanical Engineering, University of Arizona
9/2000 - 2003	Assistant Professor, Bioengineering, University of Utah
1/2001 - 5/2003	Adjunct Assistant Professor, Orthopedics, University of Utah
6/2003 - 6/2010	Adjunct Associate Professor, Orthopedics, University of Utah
3/2003 - 6/2010	Associate Professor, Bioengineering, University of Utah

7/2005 -	6/2010	Associate Chair, Department of Bioengineering, University of Utah
1/2004 -	present	Faculty Member, Scientific Computing and Imaging Institute, University of Utah
7/2010 -	present	Adjunct Professor, Orthopedics, University of Utah
7/2010 -	present	Professor, Bioengineering, University of Utah
7/2012 -	present	Adjunct Professor, School of Computing, University of Utah

Honors (selected)

ASME Van C. Mow Medal, 2013, "…for seminal contributions to research in biomechanics related to fundamental structure-function relationships in musculoskeletal soft tissue, subject-specific modeling of joint mechanics, image-based biomechanics, the mechanics of angiogenesis, and the development and distribution of the FEBio software suite".

2011 William H Harris Award from Orthopaedic Research Society, "in recognition of the outstanding quality and scientific achievement of the paper: The capsule's contribution to total hip construct stability". Journal of Orthopaedic Research, 29(11):1642-8, 2011.

2010 ASME Richard Skalak Award – co-author on "Best paper in Journal of Biomechanical Engineering".

Elected as Fellow, American Institute for Medical and Biological Engineering, December 2006.

Honored by Mentor Recognition Program (UCSD) for outstanding training of BME Undergraduates, Nov 2005.

Winner of Taylor & Francis prize for "outstanding innovation in computer methods in biomechanics & biomedical engineering", 6th Int. Symposium on Computer Methods in Biomechanics and Biomedical Engineering, Madrid, Spain, March 2004.

ASME Y-C. Fung Young Investigator Award Recipient - May 2002; NSF CAREER Award Recipient - Jan 2002; Whitaker Foundation Grant Recipient - 3/1995; NIH/NIAMS National Research Service Award - 12/1994.

Invited Speaker on over 50 occasions; 5 Keynote Lectures and 11 Plenary Lectures.

Activities and Affiliations (selected)

Program Chair, Summer Biomechanics, Biotransport and Bioengineering Conference (www.sb3c.com), 2015.

Co-Chair, Orthopaedics and Rehabilitation Engineering Track, Biomedical Engineering Society Annual Meeting, October 2014.

Conference Chair, 11th International Symposium on Computer Methods in Biomechanics and Biomedical Engineering, April 2013 (cmbbe13.sci.utah.edu/).

Member of Biomechanics Working Group (part of the Multiscale Modeling Consortium at NIBIB)

Associate Editor, ASME Journal of Biomechanical Engineering, 1/2004 to 6/2010

Associate Editor, Computer Methods in Biomechanics and Biomedical Engineering, 11/2002 to present

Editorial Board, Journal of Applied Biomechanics, 11/1996 to present

Regular ad-hoc reviewer for NIH study sections including MABS, SBSR (previously ORTH), MOSS-A, MIM.

Reviewer for NSF including CAREER Awards and BME/RAPD

ASME Bioengineering Solid Mechanics Committee, 1993 - present

C. Contributions to the Field

Ligament and tendon mechanics, injury, healing and structure-function relationships

My research career began in 1988 with the study of the structure, function, injury and healing of ligaments and tendons, and I continue this line of research to date. I demonstrated that injuries to ligament insertion sites healing much more slowly than injuries to the tissue substance due to osteoclast resorption at the mineralization front. As part of my research contributions in this area, research team determined the origins of the large Poisson's ratios in ligaments and tendons, and its implications for viscoelastic material behavior. More recently, we established the mechanical role of noncollagenous components of the extracellular matrix (decorin and other proteoglycans, elastin) in the multiaxial mechanical behavior of ligaments and tendons.

Weiss JA, Woo SL-Y, Ohland KJ, Horibe S, Newton PO: Evaluation of a new injury model to study medial collateral ligament healing: Primary repair vs nonoperative treatment. Journal of Orthopaedic Research, 9:516-528, 1991. PMID: 2045978.

Lujan TJ, Underwood CJ, Henninger HB, Thompson BM, Weiss JA: Effect of dermatan sulfate glycosaminoglycans on the quasi-static material properties of the human medial collateral ligament. Journal of Orthopaedic Research, 25(7):894-903, 2007. PMID: 17343278.

Reese SP, Maas SA, Weiss JA: Micromechanical models of helical superstructures in ligament and tendon fibers predict large Poisson's ratios. Journal of Biomechanics, 43(7):1394-1400, 2010. PMCID: 2881222.

Henninger HB, Valdez WR, Scott SA, Weiss JA: Elastin governs the mechanical response of medial collateral ligament under shear and transverse tensile loading. Acta Biomaterialia, 25:304–312, 2015. PMID: 26162584.

Computational Biomechanics

I have been involved in computational modeling applied to biomechanics since the early 1990s. My contributions have included the development and implementation of new anisotropic constitutive models for fibrous soft tissues, robust approaches for deformable image registration based on hyperelastic regularization, the adoption and promotion of a structured approach to verification and validation of models in our field, patient-specific modeling and validation of joint biomechanics, and a general framework for application of prestrain to computational models of biological materials. In collaboration with Professor Gerard Ateshian of Columbia University, I lead the development, support and distribution of FEBio (Finite Elements for Biomechanics and Biophysics (www.febio.org). My research in this area is highly cited, and the FEBio software suite is extremely popular among biomedical scientists, with over 6,000 registered users.

Weiss JA, Maker BN, Govindjee S: Finite element implementation of incompressible, transversely isotropic hyperelasticity. Computer methods in applied mechanics and engineering 135 (1), 107-128, 1996. http://mrl.sci.utah.edu/papers/fe.pdf

Maas SA, Ellis BJ, Ateshian GA, Weiss JA: FEBio: Finite elements for biomechanics. Journal of Biomechanical Engineering, 134(1), 2012. PMCID: PMC3705975.

Ateshian GA, Nims RJ, Maas S, Weiss JA: Computational modeling of chemical reactions and interstitial growth and remodeling involving charged solutes and solid-bound molecules. Biomechanics and Modeling in Mechanobiology, 13(5):1105-20, 2014. PMID: 24558059.

Maas SA, Erdemir A, Halloran JP, Weiss JA: A general framework for application of prestrain to computational models of biological materials. Journal of the Mechanical Behavior of Biomedical Materials, 61:499-510, 2016.

Subject-specific Modeling of Joint Biomechanics

My laboratory has developed and applied techniques to enable subject- and patient-specific modeling of soft tissue mechanics. Two major areas of study have been the ligaments of the knee and the articular cartilage of the hip. Among our many significant discoveries using these techniques, we determined that overload of the labrum in the hip is the likely cause of early osteoarthritis in patients with developmental dysplasia. This result was an important finding as previously it was thought that chronic stress overload of the articular surface led to joint degeneration in these patients.

Gardiner JC and Weiss JA: Subject-specific finite element models can predict strain in the human medial collateral ligament during valgus knee loading. Journal of Orthopaedic Research, 21:1098-1106, 2003. PMID: 14554224.

Anderson AE, Ellis BJ, Maas SA, Peters CL, Weiss JA: Validation of finite element predictions of cartilage contact pressure in the human hip joint. Journal of Biomechanical Engineering, 130(5):051008, 2010.

Henak CR, Anderson AE, Weiss JA: Subject-specific analysis of joint contact mechanics: application to the study of osteoarthritis and surgical planning. Journal of Biomechanical Engineering, 135(2):021003, 2013. PMID: 23404548.

Henak CR, Abraham CL, Anderson AE, Maas SA, Ellis BJ, Peters CL, *Weiss JA: Patient-specific analysis of cartilage and labrum mechanics in human hips with acetabular dysplasia. Osteoarthritis and Cartilage, 22(2):210-217, 2014. http://dx.doi.org/10.1016/j.joca.2013.11.003

Mechanics of Angiogenesis and Interaction with the Extracellular Matrix

In collaboration with Professor Jay Hoying at U. Louisville, we demonstrated that matrix strain induced by angiogenic microvessels, the effective stiffness of the matrix (influenced by both matrix density and mechanical boundary conditions), and the effects of matrix strain on collagen fibril alignment modulate the direction, rate and alignment of a growing vasculature. We have designed, implemented, and validated a coupled computational framework to study these processes, AngioFE, in which a discrete microvessel growth model interacts with a continuous finite element mesh through the application of local remodeling sprout stresses. The result of our in silico investigations demonstrate how mechanical boundary conditions, matrix density, and matrix heterogeneity affect neovascularization and matrix deformation and provides a platform for studying angiogenesis in complicated and multi-faceted mechanical environments that microvessels experience in vivo.

Krishnan L, Underwood CJ, Maas S, Ellis BJ, Kode TC, Hoying JB, Weiss JA: Effect of mechanical boundary conditions on orientation of angiogenic microvessels. Cardiovascular Research, 78(2):324-32, 2008. PMCID: 2840993.

Underwood CJ, Edgar LT, Hoying JB, Weiss JA. Cell-generated traction forces and the resulting matrix deformation modulate microvascular alignment and growth during angiogenesis. American Journal of Physiology: Heart and Circulation Physiology, 307(2):H152-64, 2014. PMID: 24816262

Utzinger U, Baggett B, Weiss JA, Hoying JB, Edgar LT: Large-scale time series microscopy of neovessel growth during angiogenesis. Angiogenesis, 18(3):219-232, 2015. PMID: 2579217.

Edgar LT, Maas SA, Guilkey JE, Weiss JA. A coupled model of neovessel growth and matrix mechanics describes and predicts angiogenesis in vitro. Biomechanics and Modeling in Mechanobiology, 14(4):767-782, 2015. PMID: 25429840.

Complete List of Published Work in MyBibliography:
http://www.ncbi.nlm.nih.gov/sites/myncbi/jeffrey.weiss.1/bibliography/40735009/public/?sort=date&direction=ascending

D. Research Support
Ongoing Research Support

1R01EB015133 (Weiss) 9/30/12 - 9/01/17
National Institutes of Health (NIBIB)
Multiscale Mechanics of Connective Tissues

The specific aims are: 1) Develop a FE-based algorithmic and software framework for analysis of nonlinear, multiscale models in biomechanics; 2) Construct idealized, multiscale physical surrogates with well-defined nano- and microstructure, consisting of extruded collagen fibers embedded within a collagen gel matrix; 3) Develop and validate parametric, multiscale FE models of the physical surrogates. Perform simulations using 1st- and 2nd-order homogenization algorithms with the software developed in Aim 1. Validate the computational models and investigate the issues of scale separation and size effects using the experimental results from Aim 2.

1R01AR069297 (Guldberg) 4/1/16 - 3/31/21
National Institutes of Health (NIAMS)
Mechanical Regulation of Vascular Growth and Remodeling
Co-Investigator(s): Jeffrey Weiss, Thomas Barker, Mark Allen

We will examine how the biomechanical environment affects bone micro-vascularity during bone healing. The first two Aims will obtain in vitro, and then in vivo data on the ability of fibrin-based hydrogels to control angiogenesis. In parallel, in silico studies, based on an already well-established computational framework for modeling microvascular growth, will be modified to reflect the needs of the current approach. The validated computational model will then allow examination of these mechanical interactions with vascular growth and remodeling in greater detail and more importantly establish a predictive framework based on this relationship that may ultimately guide post-traumatic rehabilitation programs or even the design of engineered vascularized scaffolds.

2R01GM083925 (Weiss and Ateshian) 9/1/12 - 8/31/16 (renewed for 9/1/16 – 8/31/20)
National Institutes of Health (NIGMS)
FEBio: Finite Elements for Biomechanics

The aims for the current period are: Aim 1: Implement chemical reactions between the constituents of a mixture that includes deformable porous solids under finite deformation, to model phenomena such as binding kinetics, nutrient-dependent interstitial growth mechanics, scaffold dissolution, and evolution of solid matrix properties with alterations in mass content. Aim 2: Develop and test porous shell elements that allow passive and active solute transport to model biological membranes. Aim 3: Redesign FEBio to accommodate a "plugin" framework to facilitate extension and integration with other applications. Aim 4: Extend parallelization of FEBio to include stiffness matrix and load vector assembly operations.

Ross T. Whitaker
937 E. 2nd Avenue, Salt Lake City, UT 84103
(W) 801/587-9549, (H) 801/524-0866
whitaker@cs.utah.edu — `http://www.cs.utah.edu/~whitaker`

EDUCATION

1/89 to 10/93 **The University of North Carolina** Chapel Hill, NC
Department of Computer Science: Ph.D. 1993, M.S. 1991.
- Course work emphasized computer vision, graphics, visualization, and parallel systems.
- Dissertation in nonuniform diffusion for image segmentation (advisor: S.M. Pizer).
- University of North Carolina Alumni Fellowship 1992-93.

9/82 to 6/86 **Princeton University** Princeton, NJ
Electrical Engineering and Computer Science/Engineering Physics, B.S. June 1986.
- G.P.A. 3.8/4.0.
- Summa cum laude, Phi Beta Kappa, Tau Beta Pi.

WORK EXPERIENCE

8/00–present **University of Utah** Salt Lake City, UT
8/07: Professor, School of Computing; Adjunct Professor, Bioengineering
8/03: Associate Professor, School of Computing; Adjunct Associate Professor, Bioengineering
8/00: Assistant Professor, School of Computing

3/96 to 7/00 **University of Tennessee** Knoxville, TN
Assistant Professor—Department of Electrical Engineering.

1/94 to 3/96 **European Computer-Industry Research Centre (ECRC)** Munich, Germany
Research Scientist—User Interaction and Visualization Group
- Developed new modeling methods for 3d segmentation and reconstruction.
- Built an object-oriented image processing platform.
- Researched and developed technologies for augmented reality.
- Led a small group of researchers and developed funded European research collaborations in excess of 2M DM.

HONORS AND AWARDS

- *Best Paper of Journal Award*, Medical Image Analysis 2010 (MICCAI 2009), “Manifold modeling for brain population analysis”.
- *Best Paper Award*, Int. Meshing Roundtable 2010, “Particle systems for adaptive, isotropic meshing of CAD models”.
- NSF CAREER Award (Signal Processing Systems Program, 2000)
- University of Tennessee 1997, College of Engineering/Allied Signal Award for Outstanding Research and Teaching.
- University of North Carolina, Alumni Fellowship, 1993.
- Princeton University 1986: Summa Cum Laude, Phi Beta Kappa, Tau Beta Pi.
- University of Utah, College of Engineering Dean’s list for outstanding teaching: 2011, 2013
- IEEE Fellow.
- AIMBE Fellow.

PUBLICATIONS (truncated by year)

Refereed Journals

1. A. Anderson, P. Atkins, P. Agrawal, S. Elhabian, R. Whitaker, J. Weiss, C. Peters, S. Aoki, "Which Radiographic Measurements Best Identify Anatomical Variation in Femoral Head Anatomy? Analysis Using 3D Computed Tomography and Statistical Shape Modeling", J. of Hip Preservation Surgery, 3(1), pp. 30–45, 2016.
2. I. Oguz, J. Cates, M. Datar, B. Paniagua, P.T. Fletcher, C. Vachet, M. Styner, R. Whitaker, "Entropy-based particle correspondence for shape populations,", Int. J. of Comp. Assisted Radiology and Surgery, 11(7), pp. 1221–1232, 2016.
3. "Evaluating Alignment of Shapes by Ensemble Visualization", M. Raj, M. Mirzargar, R.M. Kirby, R.T. Whitaker, J.S. Preston, *IEEE Computer Graphics and Applications*, 36(3), pp. 60–71, 2016.
4. S. Pujol et al., "The DTI Challenge: Toward Standardized Evaluation of Diffusion Tensor Imaging Tractography for Neurosurgery", *Journal of Neuroimaging*, To appear, 2015.
5. L. Liu, M. Mirzangar, R.M. Kirby, R. Whitaker, D. House, "Visualizing Time-Specific Hurricane Predictions, with Uncertainty, from Storm Path Ensembles" *Computer Graphics Forum*, 34(3), pp. 371–380, 2015.
6. Z. Fu, S. Yakovlev, R.M. Kirby, R. Whitaker, "Fast parallel solver for the levelset equations on unstructured meshes", *Concurrency and Computation: Practice and Experience*, 27(7), pp. 1639–1657, 2015.
7. J. Bronson, S. Sastry, J. Levine, R. Whitaker, "Adaptive and Unstructured Mesh Cleaving", *Procedia Engineering* 82, pp. 266–278, 2014.
8. S. Awate and R. Whitaker, "Multiatlas Segmentation as Nonparametric Regression", *IEEE Trans. on Medical Imaging*, 33(9), pp. 1803–1817, 2014.
9. J. Bronson, J. Levine, R. Whitaker, "Lattice cleaving: A multimaterial tetrahedral meshing algorithm with guarantees", *IEEE Trans. on Visualization and Comp. Graphics*, 20(2), pp. 223–237, 2014.
10. Z. Fu, T.J. Lewis, R.M. Kirby, R. Whitaker "Architecting the finite element method pipeline for the GPU" *J of Comp. and Applied Mathematics*, 257, pp. 195–211, 2014.
11. X. Hao, K. Zygmunt, R. Whitaker, P. Fletcher "Improved Segmentation of White Matter Tracts with Adaptive Riemannian Metrics" *Medical Image Analysis*, 18(1), pp. 161–175, 2014.
12. L. Luo et al. "Targeted intraceptor nanoparticle therapy reduces angiogenesis and fibrosis in primate and murine macular degeneration", *ACS Nano*, 7(4), pp. 3264–3275, 2013.
13. P. Tóth, J. Lighty, A. Palos, R. Whitaker, E. Eddings, "A novel framework for the quantitative analysis of high resolution transmission electron micrographs of soot II. Robust multiscale nanostructure quantification Combustion and Flame", *J. of Combustion and Flame*, to appear.
14. P. Tóth, J. Lighty, A. Palos, R. Whitaker, E. Eddings, "A novel framework for the quantitative analysis of high resolution transmission electron micrographs of soot I. Robust multiscale nanostructure quantification Combustion and Flame", *J. of Combustion and Flame*, to appear.
15. Z. Fu, R.M. Kirby, R. Whitaker, "A Fast Iterative Method for Solving the Eikonal Equation on Tetrahedral Domains", *SIAM J. Scientific Computing* 35(5), C473-C494, 2013.
16. M. Harris, M. Datar, R. Whitaker, E. Jurrus, C. Peters, A. Anderson, "Statistical shape modeling of cam femoroacetabular impingement" *J of Orthopaedic Research*, 31(10), pp. 1620–1626, 2013.
17. R. Whitaker, M. Mirzargar, R.M. Kirby "Contour Boxplots: A Method for Characterizing Uncertainty in Feature Sets from Simulation Ensembles" *IEEE Trans. on Visualization and Computer Graphics*, 19(12), pp. 2713–2722, 2013.
18. K.B. Jones, M. Datar, S. Ravichandran, H. Jin, E. Jurrus, R.T. Whitaker, M.R. Capecchi, "Toward an understanding of the short bone phenotype associated with multiple osteochondromas", *J. of Orthopedic Research*, 31(4), pages 651–657, 2013

19. S. Gerber, R. Whitaker, “Regularization Free Princpal Curve Estimation”, *J. of Machine Learning Research*, 14, 1285–1302, 2013.
20. S. Gerber, O. Ruebel, P.-T. Bremer, V. Pascucci, R. Whitaker, Morse-Smale Regression, *J. of Computational and Graphical Statistics*, Spring, 2012.

Conference Proceedings—Full Paper Review

1. D. Perry, R. Whitaker, “Augmented Leverage Score Sampling with Bounds”, Proc. European Conf. on Machine Learning and Knowledge Discovery in Databases, pp. 543–558, 2016.
2. J. Lewis, S. Sastry, R.M. Kirby, R. Whitaker, “A GPU-Based MIS Aggregation Strategy: Algorithms, Comparisons, and Applications Within AMG”, Proc. IEEE 22nd Int. Conf. on High Performance Computing, pp. 214–223, 2015.
3. G. Veni, S. Elhabian, R. Whitaker, “A Bayesian formulation of graph-cut surface estimation with global shape priors”, *IEEE 12th Int. Sym. on Biomedical Imaging (ISBI)*, pp. 368–371, 2015.
4. J. S. Preston, S. Joshi, R. Whitaker, “Multiscale MRF optimization for robust registration of 2D biological data”, *IEEE 12th Int. Sym. on Biomedical Imaging (ISBI)*, pp. 302–305, 2015.
5. M. Datar, I. Lyu, S. Kim, J. Cates, M. Styner, R. Whitaker, “Geodesic Distances to Landmarks for Dense Correspondence on Ensembles of Complex Shapes” *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 19–26, 2013.
6. J. Preston, C. Rottman, A. Cheryauka, L. Anderton, R. Whitaker, S. Joshi, “Multi-layer deformation estimation for fluoroscopic imaging”, *Proc. Information Processing in Medical Imaging (IPMI)*, pp. 123–134, 2013.
7. G. Veni, Z. Fu, S. Awate, R. Whitaker, “Bayesian segmentation of atrium wall using globally-optimal graph cuts on 3D meshes” *Proc. Information Processing in Medical Imaging (IPMI)*, pp. 656–667, 2013.
8. G. Veni, Z. Fu, S. Awate, R. Whitaker, “Proper ordered meshing of complex shapes and optimal graph cuts applied to atrial-wall segmentation from DE-MRI”, IEEE 10th International Symposium on Biomedical Imaging (ISBI), pp. 1296–1299, 2013.
9. S.P. Awate, P. Zhu, R.T. Whitaker, “How Many Templates Does It Take for a Good Segmentation?: Error Analysis in Multiatlas Segmentation as a Function of Database Size”, *Proc. Int. Workshop Multimodal Brain Image Analysis (MBIA, at MICCAI), Lecture Notes in Computer Science (LNCS)*, Vol. 2, pp. 103–114, 2012.
10. M. Datar, P. Muralidharan, A. Kumar, S. Gouttard, J. Piven, G. Gerig, R. Whitaker, P. T. Fletcher, “Mixed-Effects Shape Models for Estimating Longitudinal Changes in Anatomy”, *Proc. 2nd Int. MICCAI Workshop on Spatiotemporal Image Analysis for Longitudinal and Time-Series Image Data (STIA’12)*, 2012.
11. D.J. Swenson, J.A. Levine, J.D. Tate, R.T. Whitaker, R.S. MacLeod, “Impacts of Boundary Conforming Meshes on Electrical Cardiac Simulation”, *Proc 21st Int. Meshing Roundtable*, pp. 585-602, 2012.
12. Bronson, J., Levine, J. Whitaker R., “Lattice Cleaving: Conforming Tetrahedral Meshes of Multimaterial Domains with Bounded Quality”. *Proc. of the 21st Int. Meshing Roundtable*, pp. 191–209, 2012.
13. B. Paniagua, L. Bompard, J. Cates, R. Whitaker, M. Datar, C. Vachet, M. Styner, “Combined SPHARM-PDM and entropy-based particle systems shape analysis framework”, *Soc. of Photo-Optical Instrumentation Engineers (SPIE) Conf. Series,* 8317, pp. 20, 2012.

PROFESSIONAL SERVICE (truncated)

Conference Organization

- General Cochair, *IEEE Visualization*, 2010 Salt Lake City, 2011 Providence.
- Organizing Committee (Workshops Chair), *IEEE Int. Symposium on Biomedical Imaging*, 2009, Boston.
- Cochair (Cofounder), *Microscopic Image Analysis with Applications in Biology*, 2006, Copenhagen.
- Cochair, *SIAM Imaging Science*, 2005, Salt Lake City.

Committees/Other

- Member Computing Community Consortium Council, 2013–2016.

Editorial Boards

- Associate Editor, *IEEE Transactions on Medical Imaging*, 2015–present.
- Associate Editor, *IEEE Transactions on Visualization and Computer Graphics*, 2006–2011.
- Guest Editor, *Medical Image Analysis*, Special Issue on "Microscopy Image Analysis", Jan, 2009.

WORKSHOPS AND PROFESSIONAL COURSES

- "MeshMed", MICCAI 2011, (www.imm.dtu.dk/MeshMed)
- "Workshop on Computational Diffusion MRI", MICCAI, 2008.
- "Image Processing for Volume Graphics", SIGGRAPH, 2002.
- "Beyond Blobs", SIGGRAPH, 2002.
- "PDEs for Graphics and Image Processing", SIGGRAPH, 2002.
- "Image Processing for Volume Graphics", IEEE Visualization, 2002.
- "Image Processing for Volume Graphics", SIGGRAPH 2001.
- "Image Processing for Volume Graphics", IEEE Visualization, 2001.
- "Multiscale Geometric Image Analysis—Diffusion and Cores", Visualization in Biomedical Computing (VBC), 1994.

Jason Wiese

Assistant Professor
School of Computing
University of Utah
50 S. Central Campus Drive #3190
Salt Lake City, UT 84112

Mobile: +1 919 995 0334
Email: wiese@cs.utah.edu
http://www.cs.utah.edu/~wiese

Research Interests

My research seeks to achieve a unified, expressive and actionable representation of users that can empower future systems. I design and build systems to address the challenges of dealing with unified personal data. To inform the design of my systems and to demonstrate their utility, I develop approaches for interpreting personal data, create applications that leverage personal data, and conduct user studies to understand the perspectives of users and application developers. Rooted in my immediate background in computer science, I employ knowledge and methods from multiple domains in my research, including: machine learning, user-centered design, real-world data collection, and user study design.

Education

Ph.D., Human-Computer Interaction, September 2015
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA
Advised By: Prof. Jason Hong and Prof. John Zimmerman

B.S., Computer Science, Cum Laude, June 2008
University of California at San Diego, La Jolla, CA
Revelle College Provost's Honor List
Minor: Cognitive Science

Honors and Awards

2014	Yahoo Fellow
	Ubicomp Student Travel Grant
2012	Stu Card Fellowship Recipient
	Microsoft Research Student Travel Grant
2011	Facebook Ph.D. Fellowship Award Finalist
	Carnegie Mellon Usable Privacy and Security Fellowship
	Yahoo! Key Scientific Challenges Award Winner
	Facebook Ph.D. Fellowship Award Finalist

Refereed Conference Publications

P14 Das, S., **Wiese, J.**, and Hong, J.. 2016. Epistenet: facilitating programmatic access & processing of semantically related mobile personal data. In Proceedings of the 18th

International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16).

P13 Kratz, S. and **Wiese, J.**. 2016. GestureSeg: developing a gesture segmentation system using gesture execution phase labeling by crowd workers. In Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '16).

P12 Gerritsen, D., Tasse, D., Olsen, J., Vlahovic, T., Gulotta, R., Odom, W., **Wiese, J.**, and Zimmerman, J.. 2016. Mailing Archived Emails as Postcards: Probing the Value of Virtual Collections. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16).

P11 Laput, G., Lasecki, W., **Wiese, J.**, Xiao, R., Bigham, J., Harrison, C. 2015. Zensors: Adaptive, Rapidly Deployable, Human-Intelligent Sensor Feeds. In Proceedings of the 33nd Annual SIGCHI Conference on Human Factors in Computing Systems, 2015.

P10 **Wiese, J.**, Min, J.K., Hong, J. and Zimmerman, J. 2015. "You Never Call, You Never Write": Call and SMS Logs Do Not Always Indicate Tie Strength. In Proceedings of the 2015 conference on Computer supported cooperative work (CSCW '15).

P9 **Wiese, J.**, Hong, J. and Zimmerman, J. 2014. Challenges and opportunities in data mining contact lists for inferring relationships. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14). ACM, New York, NY, USA, 643-647.

P8 Min, J.K., Doryab, A., **Wiese, J.**, Amini, S., Zimmerman, J., Hong, J. 2014. Toss 'n' turn: smartphone as sleep and sleep quality detector. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 477-486.

P7 **Wiese, J.**, Saponas, T.S., Brush, A.J.Phoneprioception: enabling mobile phones to infer where they are kept. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13). ACM, New York, NY, USA, 2157-2166.

P6 Oney, S., Harrison, C., Ogan, A., **Wiese, J.** ZoomBoard: a diminutive qwerty soft keyboard using iterative zooming for ultra-small devices. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13). ACM, New York, NY, USA, 2799-2802.

P5 Min, J.K., **Wiese, J.**, Hong, J., Zimmerman, J. 2013. Mining smartphone data to classify life-facets of social relationships. In Proceedings of the 2013 conference on Computer supported cooperative work (CSCW '13). ACM, New York, NY, USA, 285-294.

P4 Sleeper, M., Balebako, R., Das, S., McConahy, A., **Wiese, J.**, Cranor, L . 2013. The post that wasn't: exploring self-censorship on facebook. In Proceedings of the 2013 conference on Computer supported cooperative work (CSCW '13). ACM, New York, NY, USA, 793-802.

P3 **Wiese, J.**, Kelley, P., Cranor, L., Dabbish, L., Hong, J., Zimmerman, J. 2011. Are you close with me? are you nearby?: investigating social groups, closeness, and willingness to share. In Proceedings of the 13th international conference on Ubiquitous computing (UbiComp '11). ACM, New York, NY, USA, 197-206.

P2 **Wiese, J.**, Biehl, J., Turner, T., van Melle, B., Girgensohn, A . 2011. Beyond 'yesterday's tomorrow': towards the design of awareness technologies for the contemporary worker. In Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11). ACM, New York, NY, USA, 455-464.

P1 Lindqvist, J., Cranshaw, J., **Wiese, J.**, Hong, J. and Zimmerman, J. 2011. I'm the mayor of my house: examining why people use foursquare - a social-driven location sharing application. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11). ACM, New York, NY, USA, 2409-2418.

Invited Talks

2015 Enabling and Ecosystem of Personal Behavioral Data, Invited Talk
Bosch Research, May.

Enabling and Ecosystem of Personal Behavioral Data, Tech Talk
Google, May.

Enabling and Ecosystem of Personal Behavioral Data, Invited Talk
University of North Carolina at Charlotte, April.

Enabling and Ecosystem of Personal Behavioral Data, Colloquium
University of Utah, March.

Enabling and Ecosystem of Personal Behavioral Data, Invited Talk
FX Palo Alto Laboratory, February.

Enabling and Ecosystem of Personal Behavioral Data, Colloquium
University of Virginia, February.

Enabling and Ecosystem of Personal Behavioral Data, Colloquium
Worcester Polytechnic Institute, February.

2014 Context Awareness, Guest Lecture, Designing Human-Centered Systems,
Carnegie Mellon University, February.

2013 Uncovering New Dimensions of Context-Awareness, Invited Talk
FX Palo Alto Laboratory, June.

2012 Mobile Social Systems, Guest Lecture, The Social Web: Content, Communities and Context
Carnegie Mellon University, December.

Understanding Social Relationships Within Interactive Systems, Invited Talk,
DUB Seminar Series, University of Washington, August.

Professional Experience

2016 to Present **University of Utah**
Salt Lake City, UT. Assistant Professor.

2015 to 2016 **FX Palo Alto Laboratory**
Palo Alto, CA. Research Scientist.
Research on unauthenticated personalization, gestural interfaces, social cues in video.

2012 **Microsoft Research**
Redmond, WA. Research Intern.
Worked with A.J. Brush and Scott Saponas to develop and evaluate Phoneprioception.

2011 **Yahoo! Labs**
Santa Clara, CA. Research Intern.

Developed an experimental system for social location sharing.

2010 **FX Palo Alto Laboratory**
Palo Alto, CA. Research Intern.
Developed and deployed an mobile social awareness system.

2007 to 2008 **Qualcomm, Inc.**
San Diego, CA. Human Factors Engineering Intern.
Worked on a variety of projects in the Advanced Technology group.

Teaching

2016 *Instructor, CS5540 Human Computer Interaction*
Introduction to user-centered design, including user research, design, prototyping, testing, and communicating process. 49 students.

2011 *Instructor, Structures of Software User Interfaces Mobile Lab*
Prepared and delivered weekly lectures, created and graded assignments, and held weekly office hours.

2010 *Teaching Assistant, Human-Computer Interaction Methods*
Advised project groups, held weekly office hours, created and graded assignments and exams

Selected Press

2015 Wired (2015). “Human Smarts Plus AI Could Unlock Computer Vision.” April 29.
PCWorld (2015). “Zensors app lets you crowdsource live camera monitoring” April 24.
Engadget (2015). “Scientists turn old smartphones into all-seeing eyes” April 22.
Gizmodo (2015). “One Old Android Phone Could Make All Your Dumb Things Smart” April 21.

2014 World Economic Forum Blog (2014) “Top 10 Emerging Technologies for 2014.” September 1.

2013 Wired (2013). "Researchers Figure Out How You Can Type on a Smartwatch." May 1.
Slashdot (2013). “Carnegie Mellon Offers Wee QWERTY Texting Tech For Impossibly Tiny Devices.” May 1.
Gizmodo (2013). "How Typing on a Smart Watch Might Actually Make Sense." April 29.
MIT Technology Review (2013). “A QWERTY Keyboard for Your Wrist.” April 27.

2010 MIT Technology Review (2010). “Someone’s Watching You.” October 28.

Curriculum Vitae

Cem Yuksel

Address

Warnock Engineering Building
72 South Campus Central Dr., Room 2686
University of Utah
Salt Lake City, Utah 84112

Email: `cem@cemyuksel.com`
Web: `www.cemyuksel.com`
Phone: (801) 581-4439

Education

2006 - 2010	PhD in Computer Science, Texas A&M University
2004 - 2006	Visualization Sciences, Texas A&M University
2003 - 2004	Visual Arts & Communication Design, Sabanci University, Turkey
2000 - 2003	MS in Computer Engineering, Bogazici University, Turkey
1995 - 2000	BS in Physics, Bogazici University, Turkey

Work Experience

2012 - present Assistant Professor School of Computing, University of Utah
Working on computer graphics research projects.

2009 - present Founder of Cyber Radiance LLC.
Working on Hair Farm, a leading hair plugin for 3ds Max.

2010 - 2012 Postdoctoral Fellow in Computer Science, Cornell University
Worked with Dr. Doug James on graphics research projects.

2006 - 2010 Research Assistant in Computer Science, Texas A&M University
Worked on several different research projects in graphics.

2004 - 2006 Research Assistant in Visualization Sciences, Texas A&M University
Worked on hair rendering with global illumination.

2002 - 2004 Teaching & Research Assistant in Sabanci University, Turkey
Taught graduate and undergraduate level courses, worked on computer vision based research projects.

1998 - 2003 Teaching & Research Assistant in Bogazici University, Turkey
Taught several graduate & undergraduate level classes, worked on global illumination and rendering algorithms.

1999 - 2003 Research Assistant in Bogazici University Pattern Analysis & Machine Vision Lab, Turkey
Worked on several graphics & vision research projects.

Publications

Peer Reviewed Journal Papers:

1. Cem Yuksel, "Sample Elimination for Generating Poisson Disk Sample Sets," *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2015)*, 37, 2, 2015.

2. Cem Yuksel, Jonathan M. Kaldor, Doug L. James, Steve Marschner, "Stitch Meshes for Modeling Knitted Clothing with Yarn-level Detail," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012)*, 31, 3, 2012.

3. Cem Yuksel, Scott Schaefer, John Keyser, "Parameterization and Applications of Catmull-Rom Curves," *Computer Aided Design*, 43, 7, 2011.
4. Cem Yuksel, John Keyser, Donald H. House, "Mesh colors," *ACM Transactions on Graphics*, 29, 2, 2010.
5. Cem Yuksel, Scott Schaefer, John Keyser, "Hair Meshes," *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2009)*, 28, 5, 2009.
6. Cem Yuksel, John Keyser, "Fast Real-time Caustics from Height Fields," *The Visual Computer (Proceedings of CGI 2009)*, 25, 5-7, 2009.
7. Arno Zinke, Cem Yuksel, Andreas Weber, John Keyser, "Dual Scattering Approximation for Fast Multiple Scattering in Hair," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008)*, 27, 3, 2008.
8. Cem Yuksel, John Keyser, "Deep Opacity Maps," *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2008)*, 27, 2, 2008.
9. Cem Yuksel, Donald H. House, John Keyser, "Wave Particles," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, 26, 3, 2007.
10. Zeki Melek, David Mayerich, Cem Yuksel, John Keyser, "Visualization of Fibrous and Thread-like Data," *IEEE Transactions on Visualization and Computer Graphics*, 12, 5, 2006.

Peer Reviewed Conference Papers (excluding papers listed above):

1. Kui Wu, Cem Yuksel, "Real-time Hair Mesh Simulation," *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D 2016)*, 2016.
2. Cem Yuksel, Scott Schaefer, John Keyser, "On the Parameterization of Catmull-Rom Curves," *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling*, 2009.
3. Mayank Singh, Cem Yuksel, Donald House, "Fast Occlusion Sweeping," *Advances in Visual Computing (Proceedings of ISVC 2009)*, 2009.
4. Cem Yuksel, "Gradient Space Projection," *Computer Graphics International 2008*, 2008.
5. Cem Yuksel, Ergun Akleman, John Keyser, "Practical Global Illumination for Hair Rendering," *Proceedings of Pacific Graphics 2007*, 2007.
6. Bei Xu, Cem Yuksel, Ali Abur, Ergun Akleman, "3D Visualization of Power System State Estimation," *Electrotechnical Conference, 2006. MELECON 2006. IEEE Mediterranean*, 2006.
7. Ergun Akleman, Cem Yuksel, "On a Family of Symmetric, Connected and High Genus Sculptures," *Bridges London: Mathematics, Music, Art, Architecture, Culture*, 2006.

Other Publications (limited peer review from abstract):

1. Ian Mallett, Cem Yuksel, Amit Prakash, "Adaptive Deferred Shading," *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2016.
2. Cem Yuksel, "Hardware Accelerated Mesh Colors," *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2016.
3. Cem Yuksel, Sarah Tariq, "Advanced techniques in real-time hair rendering and simulation," *ACM SIGGRAPH 2010 Courses*, 2010.
4. Cem Yuksel, Donald H. House, John Keyser, "Implementing Wave Particles for Real-time Water Waves with Object Interaction," *ACM SIGGRAPH 2007 Sketches*, 2007.
5. Cem Yuksel, Donald H. House, John Keyser, "Implementing Wave Particles for Real-time Water Waves with Object Interaction," *ACM SIGGRAPH 2007 Research Posters*, 2007.
6. Cem Yuksel, Ergun Akleman, "Rendering hair with global illumination," *ACM SIGGRAPH 2006 Research Posters*, 2006.
7. Cem Yuksel, Ergun Akleman, "Rendering hair-like objects with indirect illumination," *ACM SIGGRAPH 2005 Sketches*, 2005.

Videos (refereed):

1. Cem Yuksel, "Dual Scattering for Real-Time Multiple Scattering in Hair," *ACM SIGGRAPH 2008 Computer Animation Festival*, 2008.
2. Cem Yuksel, "Wave Particles," *ACM SIGGRAPH 2007 Computer Animation Festival*, 2007.

Patents:

1. Cem Yuksel, "Hair Meshes," *US and International Patent*, 2011.

Thesis:

1. Cem Yuksel, "Real-time Water Waves with Wave Particles," *PhD. Thesis, Texas A&M University*, 2010.
2. Cem Yuksel, "Gradient Space Projection," *M.S. Thesis, Bogazici University*, 2003.

Research Grants

1. NSF ESD: Computer Aided Design for 3D Fabrication Using Knitted Structures, NSF Grant #1538593. PI: Cem Yuksel, $150,000 over 2 years, 2015-2017.
2. NSF CSR: III: CGV: Medium: Architectures for Energy Efficient Ray Tracing. NSF Grant #1409129. PI: Erik Brunvand, Co-PIs: Cem Yuksel, Alan Davis, $899,991 over 4 years, 2014-2018.

Service

Conference Organization

1. General co-Chair, Interactive 3D Graphics and Games (I3D 2016) with Chris Wyman

Program Committee

1. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2015)
2. International Conference on Computer Animation and Social Agents (CASA 2015)
3. Computer-Aided Design and Computer Graphics (CAD/Graphics 2015)
4. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D 2015)
5. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2014)
6. Pacific Conference on Computer Graphics and Applications (Pacific Graphics 2014)
7. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2013)
8. International Conference on Computer Animation and Social Agents (CASA 2013)

Journal and Conference Reviews

1. ACM SIGGRAPH Conference
2. ACM SIGGRAPH Asia Conference
3. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)
4. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)
5. ACM Transactions on Graphics Journal
6. ACM Journal on Computing and Cultural Heritage
7. Eurographics Computer Graphics Forum Journal
8. Springer The Visual Computer Journal
9. IEEE Transactions on Visualization and Computer Graphics Journal
10. Elsevier Graphical Models Journal
11. Hindawi International Journal of Computer Games Technology Journal

12. International Journal of Humanities and Arts Computing
13. MDPI Fibers Journal
14. International Conference on Computer Animation and Social Agents (CASA)
15. Pacific Conference on Computer Graphics and Applications (Pacific Graphics)

University of Utah Service

1. Director of SoC BS/MS program, 2014-present.
2. SoC Faculty Search Committees, 2013-2014, 2014-2015.
3. SoC Graduate Admissions Committee, 2011-2012, 2012-2013, 2013-2014.
4. SoC Graduate Visit, 2013, 2014, 2015, 2016.

Other University Service

1. Texas A&M University Graphics 2008 Contest committee.
2. Texas A&M University Games 2007 Contest committee.

Awards

- 2010 CIFellows Award for Post-doctoral Research in Computer Science
  Sponsored by the CRA and NSF, given to 60 recipients Nationwide.
- 2010 Graduate Research Excellence Award
  Texas A&M University Department of Computer Science and Engineering, given to one graduate student each year.
- 1995 Exceptional Success in Nationwide University Entrance Examination
  The Scientific and Technical Research Council of Turkey, a scholarship given to new undergraduate students studying natural sciences, who ranked in the top 0.0001 bracket in the nationwide university entrance examination.

Teaching at the University of Utah

Fall 2015	CS 6620 - Ray Tracing for Graphics
Spring 2015	CS 4150 - Algorithms
Fall 2014	CS 6620 - Ray Tracing for Graphics
Spring 2014	CS 4150 - Algorithms
Spring 2014	CS 7933 - Graphics Seminar
Fall 2013	CS 6620 - Advanced Computer Graphics II (Ray Tracing for Graphics)
Spring 2013	CS 6958 - Special Topics in Physically Based Graphics
Spring 2013	CS 7933 - Graphics Seminar
Fall 2012	CS 6620 - Advanced Computer Graphics II (Ray Tracing for Graphics)

Joseph L. Zachary

October 25, 2016

50 S Central Campus Drive, RM 3190
School of Computing
Salt Lake City, UT 84112

(801) 581-7079 (voice)
(801) 581-5843 (fax)
zachary@cs.utah.edu
http://www.cs.utah.edu/∼zachary

Education

Ph.D., Computer Science, Massachusetts Institute of Technology, September 1987.
S.M., Computer Science, Massachusetts Institute of Technology, June 1983.
S.B., Computer Science and Engineering, Massachusetts Institute of Technology, June 1979.

Professional Employment

July 1999 to present, Professor (Lecturing) of Computer Science, University of Utah, Salt Lake City, UT.
July 1997 to June 2000, Associate Chair for Academics, University of Utah, Salt Lake City, UT.
July 1993 to June 1999, Assoc. Professor (Lecturing) of Computer Science, University of Utah, SLC, UT.
August 1987 to June 1993, Assistant Professor of Computer Science, University of Utah, SLC, UT.
September 1980 to July 1987, Graduate Research and Teaching Assistant, MIT, Cambridge, MA.
August 1979 to August 1980, Software Engineer, Digital Equipment Corporation, Merrimack, NH.

Honors

2015 College of Engineering Outstanding Service Award, University of Utah.
2010 School of Computing Outstanding Teaching Award, University of Utah.
1999 IEEE Computer Society Computer Science & Engineering Undergraduate Teaching Award.
1998 College of Engineering Outstanding Service Award, University of Utah.
1997 University of Utah Distinguished Teaching Award.
1996 Department of Energy Undergraduate Computational Science Education Award.
1995 University of Utah Presidential Teaching Scholar.
1990 College of Engineering Outstanding Teaching Award, University of Utah.
1988–2016. Recognized by Dean of Engineering for outstanding teaching evaluations in 40 courses taught.

Professional Activities

Charter member of the DOE's working group on Undergraduate Computational Engineering and Science.
Referee for numerous journals, conferences, publishers.
Panelist for NSF.

Publications

Books

J. Zachary. *Introduction to Scientific Programming: Computational Problem Solving using* Mathematica *and C*. TELOS/Springer-Verlag, 1998. 433 pages.

J. Zachary. *Introduction to Scientific Programming: Computational Problem Solving using Maple and C*. TELOS/Springer-Verlag, 1996. 371 pages.

Computer-Based Educational Material

J. Zachary and P. Jensen. HTML/JavaScript courseware. Interactive course material for a 100% online course on HTML and JavaScript programming, 2000.

Journals

Fraughton, T., Sansone, C., Butner, J., Zachary, J. (2011). Interest and performance when learning online: Providing utility value information can be important for both novice and experienced students. International Journal of Cyber Behavior, Psychology and Learning, 1(2), 1-15.

Sansone, C., Fraughton, T., Zachary, J., Butner, J., and Heiner, C. (2011). Self-regulation of Motivation when Learning Online: The Importance of Who, Why and How. Eductional Technology Research and Development, 59 (2), 199-212.

Refereed Conferences

Fraughton, T., Sansone, C., Butner, J., & Zachary, J. (January, 2012). Interest And Performance When Learning Online: Providing Utility Value Information Can Be Important For Both Novice And Experienced Students. Presented at the annual meeting of the Society for Personality and Social Psychology, San Diego, CA.

Sansone, C., Fraughton, T., Butner, J., Zachary, J. & Sinclair, S. (2011, September). Self-regulating Learning: The Relationships of Utility Value, Competence Value and Lesson Value to Interest and Learning. In K.A. Renninger (chair), Competence, Value, Achievement, and Interest: How Are They Related in Academic Motivation? Symposium at the biannual meeting of the European Association for Research in Learning and Instruction, Exeter, UK.

Sansone, C., Butner, J., Zachary, J., Fraughton, T., & Ripley, S. (2011, September). Regulating the interest experience over time: The role of utility value, on-task, and off-task behaviors. In B. Spinath (chair), What Explains the Development of Interest and Intrinsic Motivation for Learning? Symposium presented at the biannual meeting of the European Association for Research in Learning and Instruction, Exeter, UK.

Sansone, C., Butner, J., Fraughton, T.B., & Zachary, J.L. (2011, April). Self-regulatory Trade-offs When Learning Online: Interested Engagement Can Hurt AND Help. In P. O'Keefe & I. Plante (chairs), Developments in Interest Theory and Research. Symposium to be presented at the annual meeting of the American Educational Research Association, New Orleans, LA.

Fraughton, T., Sansone, C., Butner, J. & Zachary, J. (2011, January). Fully engaged: Creating an interesting experience for those with low efficacy. Presented at the annual meeting of the Society for Personality and Social Psychology, San Antonio, TX.

Sansone, C., Fraughton, T.B., Zachary, J.L., Heiner, C., & Butner, J. (2010, May). Interest, engagement and learning over time: Making it personal. In K. Ann Renninger (chair), Interest, Engagement, and Learning: Implications for STEM. Symposium at the annual meeting of the American Educational Research Association, Denver, CO.

Sansone, C., Zachary, J.L., Fraughton, T.B., Heiner, C., & Butner, J. (2010, May). Initial orientations, interest and online learning: What students do is as important as why. In K. Ann Renninger (chair), Studying Motivation and Learning Online: Prospects and Challenges. Symposium at the annual meeting of the American Educational Research Association, Denver, CO.

Fraughton, T.B., Sansone, C., Thoman, D.B., Butner, J., Zachary, J., & Thompson, W. (2010, January). Differences in task engagement as a function of self-control: Why those higher in self control might be better at regulating potential trade-offs between interest and performance. Presented at the annual meeting of the Society for Personality and Social Psychology, Las Vegas, NV.

Recent Research Support

"Increasing Student Motivation Without Compromising Student Performance in Online Classes," $499,370, National Science Foundation. 9/15/08–8/31/13. (Co-PI with Carol Sansone and William Thompson.)

Teaching

The term, number, title, text, and enrollment of recent courses taught at the University of Utah follow.

F16	CS 1040	Creating Interactive Web 0Content	Zachary	33
	CS 3100	Models of Computation	Garland	117
	CS 4150	Algorithms	das Gupta	95
S16	CS 1040	Creating Interactive Web Content	Zachary	41
	CS 3500	Software Practice		119
	CS 4150	Algoithms	das Gupta	180
F15	CS 1040	Creating Interactive Web Content	Zachary	42
	CS 1410	Introduction to Object-Oriented Programming	Eck	201
	CS 3100	Models of Computation	Garland	152
S15	CS 1040	Creating Interactive Web Content	Zachary	47
	CS 1410	Introduction to Object-Oriented Programming	Eck	189
	CS 3500	Software Practice		69
F14	CS 1040	Creating Interactive Web Content	Zachary	32
	CS 1410	Introduction to Object-Oriented Programming	Eck	179
	CS 3100	Models of Computation	Garland	93
S14		On leave		
F13		On leave		

S13	CS 1040	Creating Interactive Web Content	Zachary	71
	CS 4150	Algorithms	Dasgupta	118
	CS 4540	Web Software Architecture	Hall	71
F12	CS 1040	Creating Interactive Web Content	Zachary	65
	CS 1410	Introduction to Object-Oriented Programming		163
	CS 3500	Software Practice		196
	CS 5040	Teaching Introductory Computer Science		28
S12	CS 1040	Creating Interactive Web Content	Zachary	84
	CS 4150	Algorithms	Dasgupta	116
	CS 4540	Web Software Architecture	Hall	71
F11	CS 1040	Creating Interactive Web Content	Zachary	58
	CS 1410	Introduction to Object-Oriented Programming		147
	CS 3500/5010	Software Practice		208
	CS 5040	Teaching Introductory Computer Science		18
S11	CS 1010	Introduction to Unix	Zachary	190
	CS 1040	Creating Interactive Web Content	Zachary	72
	CS 4150	Algorithms	Dasgupta	122
	CS 4540	Web Software Architecture	Hall	66

Chairman of Dissertation and Thesis Committees

Brian Rague, (Ph.D., 2010, University of Utah).

Cecily Heiner (Ph.D., 2010, University of Utah), “Towards a Virtual Teaching Assistant to Answer Questions asked by Students in Introductory Computer Science.”

Peter Jensen (Ph.D., March 2008, University of Utah).

David Price (M.S., 2007, University of Utah). (Co-chair with Ellen Riloff.)

Elizabeth Odekirk (M.S., 2001, University of Utah), “Providing Hints to Novice Programmers by Examining Their Program’s Text and Output.”

Joseph Turner (M.S., Nov. 2000, University of Utah), “Javiva: A Compiler for Visualizing and Validating Java Classes.”

Tom Tolman (M.S., July 1994, University of Utah), “Improving the Fortran Electronic Classroom.”

Wen-Yan Kuo (Ph.D., Sept. 1992, University of Utah), “A Processing-in-Memory Computer for the Parallel Evaluation of Functional Programs.”

Laurie Hannon (M.S., July 1992, University of Utah), “A Hierarchical, Polymorphic Type System for Prolog.”

Arthur Lee (Ph.D., June 1992, University of Utah), “The Persistent Object System MetaStore: Persistence via Metaprogramming.”

Appendix E: Policy 6-400: Code of Student Rights and Responsibilities (“Student Code”)

Policy 6-400: Code of Student Rights and Responsibilities ("Student Code")

1. **Section I: General Provisions and Definitions**

   A. General Provisions

      1. The Code of Student Rights and Responsibilities has seven parts: General Provisions and Definitions, Student Bill of Rights, Student Behavior, Student Academic Performance, Student Academic Conduct, Student Professional and Ethical Conduct, and Student Records.

      2. The mission of the University of Utah is to educate the individual and to discover, refine and disseminate knowledge. The University supports the intellectual, personal, social and ethical development of members of the University community. These goals can best be achieved in an open and supportive environment that encourages reasoned discourse, honesty, and respect for the rights of all individuals. Students at the University of Utah are encouraged to exercise personal responsibility and self-discipline and engage in the rigors of discovery and scholarship.

      3. Students at the University of Utah are members of an academic community committed to basic and broadly shared ethical principles and concepts of civility. Integrity, autonomy, justice, respect and responsibility represent the basis for the rights and responsibilities that follow. Participation in the University of Utah community obligates each member to follow a code of civilized behavior.

      4. The purposes of the Code of Student Rights and Responsibilities are to set forth the specific authority and responsibility of the University to maintain social discipline, to establish guidelines that facilitate a just and civil campus community, and to outline the educational process for determining student and student organization responsibility for alleged violations of University regulations. University policies have been designed to protect individuals and the campus community and create an environment conducive to achieving the academic mission of the institution. The University encourages informal resolution of problems, and students are urged to discuss their concerns with the involved faculty member, department chair, dean of the college or dean of students. Informal resolution of problems by

mutual consent of all parties is highly desired and is appropriate at any time.

5. In cases where a more formal resolution of problems is needed, distinct administrative procedures and time lines have been established for proceedings under the Standards of Behavior (Section III), the Standards of Academic Performance (Section IV), the Standards of Academic Conduct (Section V) and the Standards of Professional Conduct (Section VI). Certain conduct by students may fall within more than one section of the Student Code. When this is the case, an appropriate University administrator shall determine which section of the code is the appropriate section under which to proceed. In special circumstances, the appropriate University administrator may extend time lines in the interest of fairness to parties or to avoid injury to one of the parties or to a member of the University community.

6. The University, the Committees and all participants shall take reasonable steps to protect the rights and, to the extent appropriate, the confidentiality of all parties involved in any proceedings under the Student Code.

7. At the sole discretion of the University, proceedings under the Student Code may be postponed when acts or conduct involving possible violations of the Standards of Behavior, the Standards of Academic Conduct or the Standards of Professional Conduct are also the subject of ongoing criminal or civil enforcement proceedings brought by federal, state, or local authorities and when postponing the proceedings will serve the best interests of the University or will better facilitate the administration of justice by such authorities. The vice president for student affairs, or designee, shall make the decision regarding proceedings under the Standards of Behavior. The senior vice president for academic affairs or the senior vice president for health sciences, or their designees, shall make the decision regarding proceedings under the Standards of Academic Conduct and the Standards of Professional Conduct.

8. The dean of students, or the senior vice president for academic affairs, or the senior vice president for health sciences, or their designees, may place a hold on the student's records and/or registration pending the resolution of proceedings under the Student Code.

B. Definitions

1. As used in the Student Code:

1. “Academic action” means the recording of a final grade (including credit/no credit and pass/fail) in a course, on a comprehensive or qualifying examination, on a culminating project, or on a dissertation or thesis. It also includes a decision by the appropriate department or college committee to place a student on academic probation, or to suspend or dismiss a student from an academic program because the student failed to meet the relevant academic standards of the discipline or program. The term “academic action” does not include the decision by a department or program to refuse admission of a student into an academic program. Academic action also does not include academic sanctions imposed for academic misconduct or for professional misconduct.

2. “Academic misconduct” includes, but is not limited to, cheating, misrepresenting one's work, inappropriately collaborating, plagiarism, and fabrication or falsification of information, as defined further below. It also includes facilitating academic misconduct by intentionally helping or attempting to help another to commit an act of academic misconduct.

    a. “Cheating” involves the unauthorized possession or use of information, materials, notes, study aids, or other devices in any academic exercise, or the unauthorized communication with another person during such an exercise. Common examples of cheating include, but are not limited to, copying from another student's examination, submitting work for an in-class exam that has been prepared in advance, violating rules governing the administration of exams, having another person take an exam, altering one's work after the work has been returned and before resubmitting it, or violating any rules relating to academic conduct of a course or program.

    b. Misrepresenting one's work includes, but is not limited to, representing material prepared by another as one's own work, or submitting the same work in more than one course without prior permission of both faculty members.

    c. “Plagiarism” means the intentional unacknowledged use or incorporation of any other person's work in, or as a basis for, one's own work offered for academic consideration or credit or for public presentation. Plagiarism includes, but is not limited to, representing as one's own, without attribution, any other individual's words, phrasing, ideas, sequence of ideas, information or any other mode or content of expression.

   d. “Fabrication” or “falsification” includes reporting experiments or measurements or statistical analyses never performed; manipulating or altering data or other manifestations of research to achieve a desired result; falsifying or misrepresenting background information, credentials or other academically relevant information; or selective reporting, including the deliberate suppression of conflicting or unwanted data. It does not include honest error or honest differences in interpretations or judgments of data and/or results.

3. “Academic sanction” means a sanction imposed on a student for engaging in academic or professional misconduct. It may include, but is not limited to, requiring a student to retake an exam(s) or rewrite a paper(s), a grade reduction, a failing grade, probation, suspension or dismissal from a program or the University, or revocation of a student's degree or certificate. It may also include community service, a written reprimand, and/or a written statement of misconduct that can be put into an appropriate record maintained for purposes of the profession or discipline for which the student is preparing.

4. “Arbitrary and capricious” means that there was no principled basis for the academic action or sanction.

5. “Behavioral misconduct” includes acts of misconduct as further defined in Section III A.

6. “Behavioral sanction” means a sanction imposed on a student for engaging in behavioral misconduct. It may include, but is not limited to, a written reprimand, the imposition of a fine or payment of restitution, community service, probation, or suspension or dismissal from the University.

7. “Business day” is every day that the University is open for business, excluding weekends and University-recognized holidays. The official calendar is maintained by the University registrar's office.

8. “Department” means an academic unit, program, department, division, college or school, whichever is the appropriate academic unit of organization.

9. “Disciplinary records” are all records relating to the imposition of an academic sanction or a behavioral sanction.

10. “Faculty” or “faculty member” refers to an individual who teaches or conducts research at or under the auspices of the University and includes students with teaching responsibilities and other instructional personnel. It also refers to the chair of a faculty committee that has assessed an academic action.

11. “Notice” or “Notification” refers to the date of delivery if notification is delivered personally or ten (10) business days after the time of postmark if the notification is mailed by U.S. mail. In the case of grades, notification refers to the date the grades are available on the World Wide Web.

12. “Professional misconduct” means the violation of professional or ethical standards for the profession or discipline for which a student is preparing as adopted or recognized as authoritative by the relevant academic program. The term also includes specific misconduct that demonstrates the student's unfitness for such profession or discipline.

13. “Program” refers to any set of courses that may be a degree, major, minor, certificate, or related course of study.

14. “Sexual harassment” is defined in Policy and Procedures 5-107.

15. “Staff” or “Staff member” refers to a person other than a faculty member who receives compensation for work or services from funds controlled by the University, regardless of the source of funds, the duties of the position, or the amount of compensation paid.

16. “Student” refers to a person who is currently, or was at the time of the offense, matriculated and/or registered in any class or program of instruction or training offered by the University at any level, whether or not for credit.

17. “University” means the University of Utah and all of its undergraduate, graduate and professional schools, divisions and programs.

18. “University activities” are teaching, research, service, administrative functions, ceremonies, or programs conducted under the auspices of the University.

19. “University premises” means the University campus and any other property, building or facility, that is owned, operated or controlled by the University.

2. Section II: Student Bill of Rights

1. Students have certain rights as members of the University community in addition to those constitutional and statutory rights and privileges inherent from the State of Utah and the United States of America. Nothing in this document shall be construed so as to limit or abridge students' constitutional rights. Students have the responsibility not to deny these rights to other members of the University community. Students have the additional legal rights and privileges described below and they will not be subject to discipline for the exercise of such rights and privileges.

   A. Learning Environment. Students have a right to support and assistance from the University in maintaining a climate conducive to thinking and learning. University teaching should reflect consideration for the dignity of students and their rights as persons. Students are entitled to academic freedom and autonomy in their intellectual pursuits and development. Students have a right to be treated with courtesy and respect.

   B. Rights in the Classroom. Students have a right to reasonable notice of the general content of the course, what will be required of them, and the criteria upon which their performance will be evaluated. Students have a right to have their performance evaluated promptly, conscientiously, without prejudice or favoritism, and consistently with the criteria stated at the beginning of the course.

   C. Role in Governance of the University. Students have a right to participate in the formulation and application of University policy affecting academic and student affairs through clearly defined means, including membership on appropriate committees and administrative bodies. Students have a right to perform student evaluations of faculty members, to examine and publish the numerical results of those evaluations, and to have those evaluations considered in the retention, promotion, tenure and post-tenure reviews of faculty members.

   D. Due Process. Students have a right to due process in any proceeding involving the possibility of substantial sanctions. This includes a right to be heard, a right to decision and review by impartial persons or bodies, and a right to adequate notice.

E. Freedom from Discrimination and Sexual Harassment. Students have a right to be free from illegal discrimination and sexual harassment. University policy prohibits discrimination, harassment or prejudicial treatment of a student because of his/her race, color, religion, national origin, sex, sexual orientation, gender identity/expression, age, or status as an individual with a disability, or as a protected veteran.

F. Freedom of Expression. Students have a right to examine and communicate ideas by any lawful means. Students will not be subject to academic or behavioral sanctions because of their constitutionally protected exercise of freedom of association, assembly, expression and the press.

G. Privacy and Confidentiality. Students have a right to privacy and confidentiality subject to reasonable University rules and regulations. Matters shared in confidence (including, but not limited to, information about a student's views, beliefs and political associations) must not be revealed by faculty members or University administrators except to persons entitled to such information by law or University policies. Students have a right to be free from unreasonable search and seizures.

H. Student Records. Students have a right to protection against unauthorized disclosures of confidential information contained in their educational records. Students have a right to examine and challenge information contained in their educational records. For detailed information regarding confidentiality of educational records, and student access to records, students should refer to Part VII, Student Records.

I. I. Student Government and Student Organizations. Students have a right to participate in elections for the Associated Students of the University of Utah. Students have a right to form student organizations for any lawful purpose.

3. **Section III: Student Behavior**

    . Standards of Behavior

    0. In order to promote personal development, to protect the University community, and to maintain order and stability on campus, students who engage in any of the following acts of behavioral misconduct may be subject to behavioral sanctions:

        1. Acts of dishonesty, including but not limited to the following:

    a. Furnishing false or misleading information to any University official.

    b. Forgery, alteration or misuse of any University document, record, fund or identification.

2. Intentional disruption or obstruction of teaching, research, administration, disciplinary proceedings or other University activities.

3. Physical or verbal assault, sexual harassment[1], hazing, threats, intimidation, coercion or any other behavior which threatens or endangers the health or safety of any member of the University community or any other person while on University premises, at University activities, or on premises over which the University has supervisory responsibility pursuant to state statute or local ordinance.

4. Attempted or actual theft, damage or misuse of University property or resources.

5. Sale or distribution of information representing the work product of a faculty member to a commercial entity for financial gain without the express written permission of the faculty member responsible for the course. (“Work product” means original works of authorship that have been fixed in a tangible medium and any works based upon and derived from the original work of authorship.)

6. Unauthorized or improper use of any University property, equipment, facilities, or resources, including unauthorized entry into any University room, building or premises.

7. Possession or use on University premises or at University activities of any firearm or other dangerous weapon, incendiary device, explosive or chemical, unless such possession or use has been authorized by the University.

8. Use, possession or distribution of any narcotic or other controlled substance on University premises, at University activities, or on premises over which the University has supervisory responsibility pursuant to state statute or local ordinance, except as permitted by law and University regulations.

9. Use, possession or distribution of alcoholic beverages of any type on University premises except as permitted by law and University regulations.

10. Violation of published University policies, rules or regulations.

11. Violation of federal, state or local civil or criminal laws on University premises, while participating in University activities, or on premises over which the University has supervisory responsibility pursuant to state statute or local ordinance.

A. Initial Oral or Written Complaint

0. Any person directly aggrieved by an alleged violation of the Standards of Behavior or any faculty member, student, or staff member may submit an oral(2) or written complaint to the dean of students, or designee, within forty-five (45) business days of the date of discovery of the alleged violation.

1. A complaint that is frivolous, that fails to state facts that constitute a violation of the Standards of Behavior, or that is not timely, may be dismissed by the dean of students, or designee, after an initial review. A person who knowingly and intentionally files a false complaint may be referred to the appropriate committee or office within the University for possible disciplinary action as described in Policy 5-111 (staff), Policy 6-316 (faculty) or this code (students).

B. Initial Inquiry and Informal Resolution

0. After an oral or written complaint has been submitted, the dean of students, or designee, shall, within ten (10) business days, give written notice to the student against whom the complaint was lodged (the responding student) of the allegations of the complaint and the procedures under the Student Code to resolve the issue.

1. Within twenty (20) business days of receipt of the complaint, the dean of students, or designee, shall begin an initial inquiry to determine whether there is a reasonable basis for believing that the responding student violated the Standards of Behavior. The dean of students, or designee, shall interview the complaining party, the responding student and any other persons believed to have pertinent factual knowledge of the allegations. The dean of students, or designee, may also review any other relevant evidence, including documentary material.

2. At the conclusion of the initial inquiry the dean of students, or designee, shall determine whether there is a reasonable basis for believing that the responding student violated the Standards of Behavior. The dean of students, or designee, shall notify the

student and the complaining party in writing of his or her decision.

3. If the dean of students, or designee, determines that there is a reasonable basis for believing that the responding student violated the Standards of Behavior, he/she shall determine whether efforts at informal resolution are appropriate and, if so, shall take whatever steps are useful to that end. Efforts to informally resolve the dispute shall occur within ten (10) business days of the conclusion of the initial inquiry or within thirty days (30) business days of receipt of the initial complaint (whichever is later). If an informal resolution is reached and the responding student complies with the terms and conditions, if any, of the resolution, no further action against the responding student will be taken and the matter will be closed.

C. Formal Written Complaint and Referral to Student Behavior Committee

0. If informal resolution is inappropriate, or if efforts at informal resolution are not successful within the allowed time period, the dean of students, or designee, shall determine whether the initial written complaint (if any) is sufficiently detailed to submit to the Student Behavior Committee. If the initial complaint was oral, or was not sufficiently detailed, the dean of students shall instruct the complaining party to prepare and submit, within five (5) business days of this instruction, a detailed formal written complaint of the circumstances giving rise to the complaint.

1. If a complaining party elects not to pursue a matter before the Student Behavior Committee, the dean of students, or designee, or another University official, may submit a formal written complaint against the responding student and pursue the matter before the Student Behavior Committee.

2. The dean of students, or designee, shall provide the responding student with a copy of the formal written complaint.

3. Within five (5) business days of receiving the formal written complaint, the responding student may submit a written response to the complaint to the dean of students, or designee.

4. The dean of students, or designee, shall refer the formal written complaint and any written response from the responding student to the Student Behavior Committee.

D. Proceedings Before the Student Behavior Committee

0. Makeup of the Committee. The Student Behavior Committee shall be composed of seven (7) members. Two members shall be faculty appointed by the president of the University upon

nomination by the Personnel and Elections Committee of the Academic Senate. Two members shall be staff appointed by the president upon nomination by the vice president for student affairs. Three members shall be students appointed by the president upon nomination by the vice president for student affairs, in consultation with the president of ASUU. At least one of the students shall be a graduate student. The president shall appoint three alternates to the Committee: one student, one faculty member, and one staff member. Student members shall serve staggered two-year terms. Faculty and staff members shall serve staggered three-year terms. The Committee chair shall be appointed by the president. The Committee shall establish internal procedures consistent with the Student Code.

1. Conflict of Interest. Upon written request of one of the parties or Committee members, the dean of students may excuse any member of the Committee if the dean determines that the member has a conflict of interest. The dean shall notify the appropriate alternate member (i.e., student, faculty member, or staff member) to replace the excused member.
2. Proceedings Before the Committee. When a timely complaint and response are filed, the Committee chair shall schedule a hearing date if:
    a. The documents raise material issues of disputed fact;
    b. The Committee chair determines that a hearing is necessary or otherwise desirable to aid in the resolution of the issues; or
    c. The possible sanctions against the responding student may include dismissal from the University, suspension from the University for longer than ten (10) business days, or revocation of the student's degree or certificate.
        1. If the Committee chair determines that no circumstances exist that require a hearing, as provided above, the chair shall notify the complaining party and the student in writing of this determination and convene a closed meeting of the Committee to consider the documentation submitted by the complaining party and the student. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the vice president for student affairs, or designee, within ten (10) business days after the Committee meeting.

3. Notice of Hearings Before Committee. If the Committee chair determines that a hearing is required, the chair shall schedule a hearing date and notify the parties[3] in writing of the date of the hearing, the names of the Committee members, and the procedures outlined below at least fifteen (15) business days prior to the hearing.

4. Hearing Procedures. Hearings shall be conducted according to the following procedures:

    a. Hearings shall be conducted within a reasonable time after the Committee's receipt of the complaint.

    b. At least five (5) business days prior to the date of the hearing, the parties shall make available to each other and to the Committee a list of their witnesses and a list of the documents to be offered at the hearing. In exceptional circumstances, the Committee may allow a party to call witnesses not listed or submit additional documents at the hearing.

    c. The parties have a right to be accompanied by any person as advisor, including legal counsel, who will be permitted to attend, but not directly participate in, the proceedings.

    d. Hearings shall be closed to the public.

    e. The hearing, except for Committee deliberations and voting, shall be recorded and a copy made available to any party upon request. Committee deliberations and voting shall take place in closed sessions.

    f. The Committee must have a quorum present to hold a hearing. A quorum consists of five (5) members, including at least one (1) student. If there is more than one hearing in a matter, or if the hearing continues over more than one session, the same five members must be present for all sessions. All findings and recommendations of the Committee shall require a majority vote of the Committee members present at the hearing.

    g. At the hearing, the parties shall have the right to present questions to witnesses through the Committee chair, to present evidence and to call witnesses in their own behalf, in accordance with the Committee's internal procedures.

h. The Committee shall not be bound by strict rules of legal evidence or procedure and may consider any evidence it deems relevant.

i. University legal counsel shall serve as a resource to the Committee and may be present at the hearing to provide guidance on substantive law and procedural matters.

j. If a majority of the Committee members find, by a preponderance of the evidence that the responding student violated the Standards of Behavior, the Committee may recommend any behavioral sanction it deems appropriate given the entire circumstances of the case, including but not limited to a written reprimand, the imposition of a fine or payment of restitution, community service, probation, suspension, or dismissal from the University.

k. The Committee shall make its findings and recommendations based only on evidence and testimony presented by the parties at the hearing. Committee members shall not conduct their own investigations, rely on prior knowledge of the facts or develop their own evidence.

l. If the complaining party or the responding student fails to attend the hearing without good cause, the Committee may proceed with the hearing and take testimony and evidence and report its findings and recommendations to the vice president for student affairs, or designee, on the basis of such testimony and evidence.

m. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the vice president for student affairs, or designee, within ten (10) business days after the conclusion of the hearing.

E. Review and Decision by the Vice President for Student Affairs or Designee

0. The vice president for student affairs, or designee, shall consider the documentation submitted to the Committee and the findings and recommendations of the Committee in making a decision. Based upon such review, and without conducting further hearings, the vice president, or designee, shall, within ten (10) business days, take one of the following actions:

a. Accept the Committee's findings and recommendations;

    b. Return the report to the Committee chair, requesting that the Committee reconvene to reconsider or clarify specific matters, materials, and issues, and forward to the vice president, or designee, a second report of its findings and recommendations relating to the specific matters referred by the vice president, or designee, for further consideration; or

    c. Reject all or parts of the Committee's findings and recommendations, stating reasons and actions to be taken therefore. The vice president may impose a greater or lesser sanction than recommended by the Committee.

1. Written notification of the vice president's, or designee's, decision shall be communicated to the parties concerned within ten (10) business days of receipt of the recommendation.

2. The vice president's, or designee's, decision is final unless appealed to the president within ten (10) business days of receipt of the decision.

F. Appeal to President

0. Within ten (10) business days of receipt of the vice president's or designee's decision, any party may appeal the decision by filing a written notice of appeal with the president and delivering a copy to the other party. The other party may file a response to the appeal with the president within five (5) business days of receipt of the appeal. In the case of an appeal:

    1. The president shall consider the appeal and the response and may solicit whatever counsel and advice the president deems appropriate to arrive at a final decision. The president may also convene an ad hoc committee composed of students and faculty members from outside the Student Behavior Committee to determine if there were substantial defects that denied basic fairness and due process. After receiving the appeal, the president shall, within ten (10) business days, or twenty (20) business days if an ad hoc committee is formed, take one of the following actions:

        a. Accept the decision of the vice president for student affairs or his/her designee;

        b. Return the report to the vice president, or his/her designee, requesting that he/she clarify specific matters, materials, and issues, and forward to the president a second report of his/her decision relating to the specific matters referred by the president for further explanation; or

c. Reject all or parts of the vice president's, or designee's, decision, stating reasons and actions for either imposing a greater or lesser sanction than determined by the vice president.

2. Written notification of the president's decision and the basis for that decision shall be communicated to the parties concerned within ten (10) business days after receipt of the appeal, or within twenty (20) business days after receipt of the appeal if an ad hoc committee is formed.

3. The decision of the president is final.

G. Suspension or Dismissal from the University for Behavioral Misconduct

0. The sanctions of suspension or dismissal from the University for behavioral misconduct may be imposed: (1) if agreed upon in informal resolution between the responding student and the dean of students or designee; (2) if recommended by the Student Behavior Committee to the vice president for student affairs or designee; (3) by the vice president for student affairs or designee notwithstanding the recommendation of the Committee; or (4) by the president notwithstanding the decision of the vice president for student affairs. A student who has been suspended or dismissed from the University shall be denied all privileges accorded to a student.

1. Suspension

a. Suspension from the University shall be for a minimum time of one semester following the semester the student is found responsible for the behavioral misconduct.

b. The office of the dean of students shall notify the student in writing of the suspension, conditions for reinstatement, and of the obligation of the student to petition for reinstatement. Notice of the suspension shall also be provided to the student's department chair.

c. Petitions for reinstatement shall be submitted to the office of the dean of students and shall explain how the conditions for reinstatement have been met.

        d. The office of the dean of students shall consider the petition and shall issue a decision regarding the student's reinstatement within fifteen (15) business days of receipt of the petition.

        e. The office of the dean of students may grant conditional reinstatement contingent upon the student meeting written requirements specified by the office of the dean of students or by the chair of the Student Behavior Committee in the original sanction to the extent that such conditions pertain to the original offense in the original sanction.

    2. Dismissal

        a. Dismissal from the University is final. A student dismissed from the University for behavioral misconduct may not petition for reinstatement.

        b. Permanent records of dismissal shall be kept in the office of the dean of students. Notice of the dismissal shall be provided to the student's department chair.

        c. The dismissed student's transcript will reflect his/her dismissal. [See Procedure 6-400-Sec.VII#1]

        d. Dismissal should be reserved for only the most egregious of offenses.

H. Administrative Suspension to Protect the University Population

    0. The vice president for student affairs (or designee) or the senior vice president for academic affairs (or designee) or the senior vice president for health sciences (or designee) may suspend a student from the University prior to an initial inquiry and hearing before the Student Behavior Committee if such action appears necessary to protect the health or well-being of any member of the University community, any member of the public, or to prevent serious disruption of the academic process. Prior to, contemporaneous with, or immediately after the suspension, the vice president shall give the student written notice of the suspension specifying the alleged misconduct and setting forth briefly the relevant facts and supporting evidence. The vice president shall then provide the student with an opportunity to meet with him/her to present the student's views and object to the suspension. This meeting shall take place prior to the suspension taking effect or as soon as possible thereafter. The

vice president shall thereafter immediately refer the complaint to the appropriate University administrator for proceedings under the code, and the suspension will be in effect pending a final determination of the matter. The vice president shall notify other University administrators of the suspension as appropriate.

I. Other University Proceedings

   0. If the filing of a complaint or an appeal concerning behavioral misconduct under the Student Code raises issues of academic misconduct or professional misconduct, the dean of students, or designee, shall immediately notify the involved faculty member, dean or cognizant senior vice president and these individuals shall determine the appropriate procedure(s) for processing the complaint or the appeal.

J. Retention of Records of Proceedings

   0. Records of proceedings under the Student Code shall be confidential to the extent permitted by law. Records of behavioral misconduct shall be kept in the office of the dean of students, and a copy may be retained in other academic departments as appropriate.

4. Section IV: Student Academic Performance

. Standards of Academic Performance

   0. In order to ensure that the highest standards of academic performance are promoted and supported at the University, students must:

      0. Meet the academic requirements of a course; and

      1. Meet the academic requirements of the relevant discipline or program.

         0. Faculty members are qualified as professionals to observe and judge all aspects of a student's academic performance, including demonstrated knowledge, technical and interpersonal skills, attitudes and professional character, and ability to master the required curriculum. An academic action, as defined in Section I.B., may be overturned on appeal only if the academic action was arbitrary or capricious.

A. Appeals Process

   0. A student who believes that an academic action taken in connection with Subsection A above is arbitrary or capricious

should, within twenty (20) business days of notification of the academic action, discuss the academic action with the involved faculty member[4] and attempt to resolve the disagreement. If the faculty member does not respond within ten (10) business days, if the student and faculty member are unable to resolve the disagreement, or if the faculty member fails to take the agreed upon action within ten (10) business days, the student may appeal the academic action in accordance with the following procedures. It is understood that all appeals and proceedings regarding academic actions will initiate with the faculty and administrators in the college or program offering the course in question. If the course is cross-listed, appeals and proceedings shall take place with the faculty and administrators offering the section for which the student is registered.

0. Appeal to Chair of the Department or Dean's Designee[5]. Within forty (40) business days of notification of the academic action, the student shall appeal the academic action in writing to, and consult with, the chair of the relevant department regarding such academic action. Within fifteen (15) business days of consulting with the student, the chair shall notify the student and faculty member, in writing, of his/her determination of whether the academic action was arbitrary or capricious and of the basis for that decision. If the chair determines that the academic action was arbitrary or capricious, the chair shall take appropriate action to implement his/her decision unless the faculty member appeals the decision. If the chair fails to respond in fifteen (15) business days, the student may appeal to the Academic Appeals Committee.

1. Appeal to Academic Appeals Committee. If either party disagrees with the chair's decision, that party may appeal to the college's Academic Appeals Committee within fifteen (15) business days of notification of the chair's decision in accordance with the procedures set forth in Subsection C, below.

B. Proceedings Before the Academic Appeals Committee

0. Written Appeal. The appeal to the Academic Appeals Committee shall set forth in writing the reasons for the appeal, shall be addressed to the Committee, and shall be delivered to the chair of the Committee, with a copy to the other party.

1. Response to Appeal. The faculty member whose decision is being appealed, or the student in the case of a faculty member's appeal, may deliver a response to the appeal to the chair of the

Academic Appeals Committee, with a copy to the other party, no later than five (5) business days after receipt of the complaint and recommendations.

2. Makeup of the Committee. The dean of each college shall ensure that an Academic Appeals Committee is constituted according to college procedures, subject to the following parameters. Two faculty members shall come from the college. The Personnel and Elections Committee of the Academic Senate shall appoint one faculty member from outside the college. The faculty members shall be appointed to the Committee for staggered three-year terms. The dean shall appoint two undergraduate student members and two graduate student members who are either from the relevant Student Advisory Committee or listed as a major within the college. Undergraduate student and graduate student members will be appointed for staggered two-year terms(6). No more than one faculty member and two Committee members in total may come from the same department in a multi-department college. The members of the Committee who shall hear the case are the three faculty members and the two students from the appealing student's peer group (i.e., undergraduates or graduates). The dean shall designate one of the faculty members to serve as chair of the Committee. The Committee shall establish internal procedures consistent with the Student Code.

3. Conflicts of Interest. Upon written request of one of the parties or Committee members, the dean may excuse any member of the Committee if the dean determines that the member has a conflict of interest. The dean shall select an appropriate replacement for the excused member (i.e., student or faculty member).

4. Proceedings Before the Committees. When an appeal and response are filed in a timely manner, the Committee chair shall schedule a hearing date if:

    . The documents raise material issues of disputed fact;

    a. The Committee chair determines that a hearing is necessary or otherwise desirable to aid in the resolution of the issues; or

    b. The academic action included dismissal from a program.

        0. If the Committee chair determines that no circumstances exist that require a hearing, as provided above, the chair shall within a reasonable time notify the student and the faculty member (the parties) in writing of this determination and

convene a closed meeting of the Committee to consider the documentation submitted by the parties. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the dean of the college, or designee, within ten (10) business days after the Committee meeting.

5. Notice of Hearings Before Committees. If the Committee chair determines that a hearing is required, the chair shall schedule a hearing date and notify the parties in writing of the date of the hearing, the names of the Committee members, and the procedures outlined below at least fifteen (15) business days prior to the hearing.

6. Hearing Procedures. Hearings shall be conducted according to the following procedures:

   . Hearings shall be conducted within a reasonable time after the Committee's receipt of the written appeal and written response to the appeal.

   a. At least five (5) business days prior to the date of the hearing, the parties shall make available to each other and to the Committee a list of their witnesses and a list of the documents to be offered at the hearing. In exceptional circumstances, the Committee may allow a party to call witnesses not listed or submit additional documents at the hearing.

   b. The parties have a right to be accompanied by any person as advisor, including legal counsel, who will be permitted to attend, but not directly participate in, the proceedings.

   c. Hearings shall be closed to the public.

   d. All hearings, except Committee deliberations and voting, shall be recorded and a copy made available to any party upon request. Committee deliberations and voting shall take place in closed sessions.

   e. The Committee must have a quorum present to hold a hearing. A quorum consists of three (3) members, including at least one (1) student and the faculty member from outside the college. If there is more than one hearing in a matter, or if the hearing continues over more than one session, the same three members must be present for all sessions. All findings and recommendations of the

Committee shall require a majority vote of the Committee members present at the hearing.

f. At the hearing, the parties shall have the right to present questions to witnesses through the Committee chair, to present evidence and to call witnesses in their own behalf, in accordance with the Committee's established internal procedures.

g. The Committee shall not be bound by strict rules of legal evidence or procedure and may consider any evidence it deems relevant.

h. University legal counsel shall serve as a resource to the Committee and may be present at the hearing to provide guidance on substantive law and procedural matters.

i. To overturn the original academic action, the Committee must find that the academic action was arbitrary or capricious.

j. The Committee shall make its findings and recommendations based only on evidence and testimony presented by the parties at the hearing. Committee members shall not conduct their own investigations, rely on prior knowledge of the facts or develop their own evidence.

k. If either party to the appeal fails to attend the hearing without good cause, the Committee may proceed with the hearing and take testimony and evidence and report its findings and recommendations to the dean of the college, or designee, on the basis of such testimony and evidence.

l. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the dean of the college, or designee, within ten (10) business days after the conclusion of the hearing.

C. Review and Decision by the Dean or Designee

0. The dean of the college, or designee, shall consider the documentation submitted to the Committee and the findings and recommendations of the Committee in making a decision. Based upon such review, and without conducting further hearings, the dean of the college, or designee, shall, within ten (10) business days, take one of the following actions:

. Accept the Committee's findings and recommendations;

a. Return the report to the Committee chair, requesting that the Committee reconvene to reconsider or clarify specific matters, materials, and issues, and forward to the dean of the college, or designee, a second report of its findings and recommendations relating to the specific matters referred by the dean of the college, or designee, for further consideration; or

b. Reject all or parts of the Committee's findings and recommendations, stating reasons and actions to be taken therefore.

1. Written notification of the dean's, or designee's, decision shall be communicated to the parties, to the chair of the Academic Appeals Committee and to the cognizant vice president within ten (10) business days after receipt of the recommendation.

2. The dean's, or designee's, decision is final unless appealed to the cognizant vice president within ten (10) business days after receipt of the decision.

D. Appeal to Cognizant Senior Vice President

0. Within ten (10) business days of receipt of the dean's, or designee's, decision, any party may appeal the decision by filing a written notice of appeal with the senior vice president for academic affairs or the senior vice president for health sciences, as appropriate, and delivering a copy to the other party. The other party may file a response to the appeal with the vice president within five (5) business days of receipt of the notice of appeal. In the case of an appeal:

0. The vice president shall consider the appeal and response to the appeal, and may solicit whatever counsel and advice the vice president deems appropriate to arrive at a final decision. The vice president may also convene an ad hoc committee composed of students and faculty members from outside the college or department to determine if there were substantial defects that denied basic fairness and due process. After receiving the appeal, the vice president shall within ten (10) business days, or within twenty (20) business days if an ad hoc committee is formed, take one of the following actions:

. Accept the decision of the dean of the college or his/her designee;

a. Return the report to the dean of the college, or his/her designee, requesting that he/she clarify specific matters, materials, and issues, and forward to the vice president a second report of his/her decision relating to the specific matters referred by the vice president for further explanation; or

b. Reject all or parts of the dean's, or designee's, decision, stating reasons and actions to be taken therefore.

1. Written notification of the vice president's decision and the basis for that decision shall be communicated to the parties, to the chair of the Academic Appeals Committee and to the dean within ten (10) business days after receipt of the appeal, or within twenty (20) business days after receipt of the appeal if an ad hoc committee is formed.

2. The decision of the vice president is final. At the conclusion of the appeals process, the chair of the department or dean of the college considering the academic appeal shall take appropriate action to implement the final decision.

E. Copies of Documents to Department Chair

0. During the appeals process and at the time they are submitted, the following documents should be copied to the chair of the department considering the academic appeal: the first written appeal, all subsequent appeals, all responsive documents, and all written recommendations or decisions made at each level of the appeal.

F. Programs That Do Not Report to Academic Deans

0. In cases where a program does not report directly to an academic dean, the program director will serve as department chair, and the cognizant associate vice president will serve as dean for purposes of these proceedings. Any ambiguity concerning appeal procedures for courses offered in a program (e.g., determination of the relevant Academic Appeals Committee) shall be resolved by the program director, in consultation with the cognizant associate vice president, and in a manner that preserves the spirit and intent of this policy.

5. Section V: Student Academic Conduct

. Standards of Academic Conduct

0. In order to ensure that the highest standards of academic conduct are promoted and supported at the University, students

must adhere to generally accepted standards of academic honesty, including but not limited to refraining from cheating, plagiarizing, research misconduct[7] misrepresenting one's work, and/or inappropriately collaborating.

A. Academic Misconduct

0. A student who engages in academic misconduct as defined in Part I.B. may be subject to academic sanctions including but not limited to a grade reduction, failing grade, probation, suspension or dismissal from the program or the University, or revocation of the student's degree or certificate. Sanctions may also include community service, a written reprimand, and/or a written statement of misconduct that can be put into an appropriate record maintained for purposes of the profession or discipline for which the student is preparing.

    0. Any person who observes or discovers academic misconduct by a student should file a written complaint with the faculty member responsible for the pertinent academic activity within thirty (30) business days of the date of discovery of the alleged violation.

    1. A faculty member who discovers or receives a complaint of misconduct relating to an academic activity for which the faculty member is responsible shall take action under this code and impose an appropriate sanction for the misconduct.

    2. Upon receipt of a complaint or discovery of academic misconduct, the faculty member shall make reasonable efforts to discuss the alleged academic misconduct with the accused student no later than twenty (20) business days after receipt of the complaint, and give the student an opportunity to respond. Within ten (10) business days thereafter, the faculty member shall give the student written notice of the academic sanction, if any, to be taken and the student's right to appeal the academic sanction to the Academic Appeals Committee for the college offering the course. Such sanctions may include requiring the student to rewrite a paper(s) or retake an exam(s), a grade reduction, a failing grade for the exercise, or a failing grade for the course[8]. In no event shall the academic sanction imposed by the faculty member be more severe than a failing grade for the course.

    3. If the faculty member imposes the sanction of a failing grade for the course, the faculty member shall, within ten (10) business days of imposing the sanction, notify in

writing, the chair[9] of the student's home department[10] and the senior vice president for academic affairs or senior vice president for health sciences, as appropriate, of the academic misconduct and the circumstances which the faculty member believes support the imposition of a failing grade. If the sanction imposed by the faculty member is less than a failing grade for the course, the faculty member should report the misconduct to the dean or chair of the student's home department or college.[11] Each college shall develop a policy specifying the dean and/or the chair as the appropriate person to receive notice of sanctions less than a failing grade for the course.

4. A student who believes that the academic sanction given by the faculty member is arbitrary or capricious should discuss the academic sanction with the faculty member and attempt to resolve the disagreement. If the student and faculty member are unable to resolve the disagreement, the student may appeal the academic sanction to the Academic Appeals Committee for the college offering the course within fifteen (15) business days of receiving written notice of the academic sanction.

5. If the faculty member, chair or vice president believes that the student's academic misconduct warrants an academic sanction of probation, suspension or dismissal from a program, suspension or dismissal from the University, or revocation of a student's degree or certificate, he/she may, within thirty (30) business days of receiving notice of the misconduct, prepare a complaint with recommendations, refer the matter to the chair or dean's designee of the student's home department or college,[12] and notify the student of the complaint and recommendation. The chair and/or dean's designee of the home department/college may undertake an investigation of the allegations and recommendations set forth in the complaint. Within ten (10) business days of receipt of the complaint, the chair and/or dean's designee shall forward the complaint and recommendation to the Academic Appeals Committee of the home college for proceedings in accordance with Section C, below, and so notify the student in writing. The chair and/or dean may accompany the complaint with his/her own recommendation supporting or opposing the sanction sought in the complaint. The person initiating the original complaint continues as the complainant in the case unless that person and the chair/dean's designee both

agree that the latter shall become the complainant. If the student has appealed the academic sanction imposed by the faculty member, the time periods set forth in this paragraph may be extended until ten (10) business days after the resolution of the student's appeal.

6. If a department chair, the dean, the senior vice president for academic affairs and/or the senior vice president for health sciences, become aware of multiple acts of academic misconduct by a student, they or their designees may, within thirty (30) business days after receiving notice of the last act of misconduct,(13) prepare a complaint with recommendations for probation, suspension or dismissal from a program, suspension or dismissal from the University, or revocation of a degree or certificate, and refer the matter to the Academic Appeals Committee of the student's home college(14) for proceedings in accordance with Section C, below, and so notify the student in writing.

B. Proceedings Before the Academic Appeals Committee

0. Written Complaint and Recommendations or Appeal. The written complaint and recommendations or the written appeal shall be delivered to the chair of the Committee, with a copy to the other party.

1. Response to Complaint and Recommendations or Appeal. The person responding to the complaint and recommendations or the appeal may deliver his/her response to the chair of the Academic Appeals Committee, with a copy to the other party, no later than five (5) business days after receipt of the complaint and recommendations.

2. Makeup of the Committee. The dean of each college shall ensure that an Academic Appeals Committee is constituted according to college procedures, subject to the following parameters. Two faculty members shall come from the college. The Personnel and Elections Committee of the Academic Senate shall appoint one faculty member from outside the college. The faculty members shall be appointed to the Committee for staggered three-year terms. The dean shall appoint two undergraduate student members and two graduate student members who are either from the relevant Student Advisory Committee or listed as a major within the college. Undergraduate student and graduate student members will be appointed for staggered two-year terms.(15) No more than one faculty member and two Committee members in total may come from the same department in a multi-

department college. The members of the Committee who shall hear the case are the three faculty members and the two students from the peer group of the student accused of academic misconduct (i.e., undergraduates or graduates). The dean shall designate one of the faculty members to serve as chair of the Committee. The Committee shall establish internal procedures consistent with the Student Code.

3. Conflicts of Interest. Upon written request of one of the parties or Committee members, the dean may excuse any member of the Committee if the dean determines that the member has a conflict of interest. The dean shall select an appropriate replacement for the excused member (i.e., student or faculty member).

4. Scheduling Hearings Before the Committees. When a complaint and recommendations or an appeal, together with a response, are filed in a timely manner, the Committee chair shall schedule a hearing date if:

   . The documents raise material issues of disputed fact;

   a. The Committee chair determines that a hearing is necessary or otherwise desirable to aid in the resolution of the issues; or

   b. The possible sanctions against the student may include dismissal from the University, dismissal from a program, suspension from either for longer than ten (10) business days, or revocation of the student's degree or certificate.

      0. If the Committee chair determines that no circumstances exist that require a hearing, as provided above, the chair shall notify the student and the faculty member (the parties) in writing of this determination and convene a closed meeting of the Committee to consider the documentation submitted by the parties. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the dean of the college, or designee, within ten (10) business days after the Committee meeting.

5. Notice of Hearings Before Committees. If the Committee chair determines that a hearing is required, the chair shall schedule a hearing date and notify the parties in writing of the date of the hearing, the names of the Committee members, and the procedures outlined below at least fifteen (15) business days prior to the hearing.

6. Hearing Procedures. Hearings shall be conducted according to the following procedures:

   . Hearings shall be conducted within a reasonable time after the Committee's receipt of the written complaint and recommendations or the written appeal, and the response.

   a. At least five (5) business days prior to the date of the hearing, the parties shall make available to each other and to the Committee a list of their witnesses and a list of the documents to be offered at the hearing. In exceptional circumstances, the Committee may allow a party to call witnesses not listed or submit additional documents at the hearing.

   b. The parties have a right to be accompanied by any person as advisor, including legal counsel, who will be permitted to attend, but not directly participate in, the proceedings.

   c. Hearings shall be closed to the public.

   d. All hearings, except Committee deliberations and voting, shall be recorded and a copy made available to any party upon request. Committee deliberations and voting shall take place in closed sessions.

   e. The Committee must have a quorum present to hold a hearing. A quorum consists of three (3) members, including at least one (1) student and the faculty member from outside the college. If there is more than one hearing in a matter, or if the hearing continues over more than one session, the same three members must be present for all sessions. All findings and recommendations of the Committee shall require a majority vote of the Committee members present at the hearing.

   f. At the hearing, the parties shall have the right to present questions to witnesses through the Committee chair, to present evidence and to call witnesses in their own behalf, in accordance with the Committee's established internal procedures.

   g. The Committee shall not be bound by strict rules of legal evidence or procedure and may consider any evidence it deems relevant.

h. University legal counsel shall serve as a resource to the Committee and may be present at the hearing to provide guidance on substantive law and procedural matters.

i. In the hearing, the Committee must determine, by a preponderance of the evidence, whether the student engaged in the alleged academic misconduct. If the Committee answers this question in the affirmative, the Committee may then recommend any academic sanction it deems appropriate under the entire circumstances of the case, including but not limited to suspension or dismissal from the program or the University, or revocation of a student's degree or certificate.

j. The Committee shall make its findings and recommendations based only on evidence and testimony presented by the parties at the hearing. Committee members shall not conduct their own investigations, rely on prior knowledge of the facts or develop their own evidence.

k. If either party presenting to the Academic Appeals Committee fails to attend the hearing without good cause, the Committee may proceed with the hearing and take testimony and evidence and report its findings and recommendations to the dean of the college, or designee, on the basis of such testimony and evidence.

l. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the dean of the college, or designee, within ten (10) business days after the conclusion of the hearing. A report that recommends sanctions no more serious than a failing grad, shall be presented to the dean of the college offering the course. Reports recommending sanctions greater than a failing grade (e.g. suspension or dismissal) shall be presented to the dean of the student's home college.(16)

C. Review and Decision by the Dean or Designee

0. The dean of the college, or designee, shall consider the documentation submitted to the Committee and the findings and recommendations of the Committee in making a decision. Based upon such review, and without conducting further hearings, the dean of the college, or designee, shall, within ten (10) business days, take one of the following actions:

. For any recommendation other than dismissal or suspension from the University or revocation of a degree or certificate, accept the Committee's findings and recommendations and impose the recommended sanctions;

a. For a recommendation of dismissal or suspension from the University or revocation of a degree or certificate, concur with the Committee's findings and recommendations and refer the matter with a confirming recommendation to the cognizant vice president for a decision;

b. Return the report to the Committee chair[17], requesting that the Committee reconvene to reconsider or clarify specific matters, materials, and issues, and forward to the dean of the college, or designee, a second report of its findings and recommendations relating to the specific matters referred by the dean of the college, or designee, for further consideration. (If a report to the dean recommends sanctions greater than a failing grade and has originated from a Committee outside of the dean's college, the dean may refer the matter to the chair of his/her own college Academic Appeals Committee for further review and recommendations.); or

c. Reject all or parts of the Committee's findings and recommendations, stating reasons and actions to be taken therefore. The dean may impose (or recommend to the cognizant vice president) a greater or lesser sanction than recommended by the Committee.

1. Written notification of the dean's, or designee's, decision shall be communicated to the parties, to the chair of the Academic Appeals Committee and to the cognizant senior vice president within ten (10) business days of receipt of the Committee's findings and recommendations.

2. The dean's, or designee's, decision is final unless appealed to the cognizant senior vice president within ten (10) business days.

D. Appeal to Cognizant Senior Vice President (or to the President when appropriate)[18]

0. Within ten (10) business days of receipt of the dean's, or designee's, decision, any party may appeal the decision by filing a written notice of appeal with the senior vice president for academic affairs or the senior vice president for health sciences, as appropriate, and delivering a copy to the other party. The

other party may file a response to the appeal with the vice president within five (5) business days of receipt of the appeal. In the case of an appeal:

0. The vice president shall consider the appeal and response to the appeal, and may solicit whatever counsel and advice the vice president deems appropriate to arrive at a final decision. The vice president may also convene an ad hoc committee composed of students and faculty members from outside the college or department to determine if there were substantial defects that denied basic fairness and due process. After receiving the appeal, the vice president shall, within ten (10) business days, or within twenty (20) business days if an ad hoc committee is formed, take one of the following actions:

   . Accept the decision of the dean of the college or his/her designee;

   a. Return the report to the dean of the college, or his/her designee, requesting that he/she clarify specific matters, materials and issues, and forward to the vice president a second report of his/her decision relating to the specific matters referred by the vice president for further explanation; or

   b. Reject all or parts of the dean's, or designee's, decision, stating reasons and actions for imposing a greater or lesser sanction than determined by the dean.

1. Written notification of the vice president's decision and the basis for that decision shall be communicated to the parties, to the chair of the Academic Appeals Committee and to the dean within ten (10) business days after receipt of the appeal, or within twenty (20) business days after receipt of the appeal if an ad hoc committee is formed.

2. The decision of the vice president is final.

E. Suspension or Dismissal from a Program or from the University, or Revocation of a Degree or Certificate

   0. The sanctions of suspension and dismissal and revocation for academic misconduct may be imposed: (1) if recommended by the Academic Appeals Committee to the dean; (2) if deemed appropriate by the dean notwithstanding the recommendation from the committee; or (3) by the cognizant vice president notwithstanding the decision (or recommendation) of the dean. A

student who has been suspended or dismissed from the University shall be denied all privileges accorded to a student.

0. Suspension from a Program or from the University.

   . Suspension shall be for a minimum time of one semester following the semester the student is found responsible for academic misconduct.

   a. The dean of the relevant college shall notify the student in writing of the suspension, conditions for reinstatement, and of the obligation of the student to petition for reinstatement.

   b. Petitions for reinstatement shall be submitted to the relevant dean and shall explain how the conditions for reinstatement have been met.

   c. The relevant dean shall consider the petition and shall issue a decision regarding the student's reinstatement within fifteen (15) business days of receipt of the petition.

   d. The relevant dean may grant conditional reinstatement contingent upon the student meeting written requirements specified in the original sanction (e.g., minimum grade point average requirement, ineligibility to participate in specified student activities or on specified student committees).

   e. The notice of the dates for which the student is suspended will remain on his/her transcript until he/she has been reinstated to the program or to the University, or for five (5) years if he/she is not reinstated to the program or to the University. [SeeProcedure 6-400-Sec.VII #1]

1. Dismissal from a Program or from the University.

   . Dismissals from a program or from the University are final. A student dismissed from a program or from the University for academic misconduct may not petition for reinstatement.

   a. Permanent records of dismissal shall be kept in the office of the registrar.

b. The dismissed student's transcript will reflect his/her dismissal. [See Procedure 6-400-Sec.VII #1]

c. Dismissal should be reserved for only the most egregious of offenses.

2. Revocation of a Degree or Certificate.

. Decisions to revoke a degree or certificate are final.

a. Permanent records concerning the revocation of a degree or certificate shall be kept in the office of the registrar.

b. The revocation of a degree or certificate shall be noted on the student's transcript. [See Procedure 6-400-Sec.VII #1]

c. Revocation of a degree or certificate should be reserved for only the most egregious of offenses.

F. Copies of Documents to Department Chair

0. During the appeals process and at the time they are submitted, the following documents should be copied to the chair of the department considering the academic misconduct: the first written complaint and recommendations, the first written appeal, all subsequent appeals, all responsive documents, and all written recommendations or decisions made at each level of the appeal.

G. Programs That Do Not Report to Academic Deans

0. In cases where a program does not report directly to an academic dean, the program director will serve as department chair, and the cognizant associate vice president will serve as dean for purposes of these proceedings. Any ambiguity concerning procedures set forth in this policy for courses offered in a program (e.g. determination of the relevant Academic Appeals Committee) shall be resolved by the program director, in consultation with the cognizant associate vice president, and in a manner that preserves the spirit and intent of this policy.

H. Implementation of Sanction for Academic Misconduct

0. At the conclusion of the appeals process, the chair of the department or dean of the college considering the academic misconduct shall take appropriate action to implement the final decision. If the student is found responsible for academic misconduct, the chair or dean shall notify, in writing, the student's

department or program of study of the violation, the proceedings, and the final decision.(19) If the sanction involves suspension or dismissal from a program or from the University or revocation of a degree or certificate, the chair or dean shall also convey the decision to the office of the registrar for notation on the transcript. [SeeProcedure 6-400-Sec.VII #1]

I. Reporting of Academic Misconduct

   0. No University employee shall provide information to a person or entity concerning a student's academic misconduct without fully complying with The Family Educational Rights and Privacy Act (20 U.S.C.A. § 1232g) and the Government Records Access and Management Act (Utah Code Title 63G - Chapter 2). In most circumstances, such as requests from a licensing body or an employer, information may only be provided with the prior written consent of the student. In some circumstances, however, such as requests from other institutions where the student seeks or intends to enroll, information may be provided without the consent of the student but only after following appropriate procedures outlined in these statutes.

J. Other University Proceedings

   0. If the filing of a complaint or an appeal relating to academic misconduct raises other issues concerning behavioral or professional misconduct, the cognizant senior vice president, or designee, the dean of students, and the involved University administrator shall determine the appropriate procedure(s) for processing the complaint or the appeal.

K. Retention of Records of Proceedings

   0. Records of proceedings under the Student Code shall be confidential to the extent permitted by law. Records of academic misconduct shall be kept in the office of the registrar, and a copy may be retained in other academic departments as appropriate.

6. Section VI: Student Professional and Ethical Conduct

. Standards of Professional Conduct

   0. In order to ensure that the highest standards of professional and ethical conduct are promoted and supported at the University, students must adhere to the prescribed professional and ethical standards of the profession or discipline for which the student is preparing, as adopted or recognized as authoritative by the relevant academic program.

A. Professional Misconduct

0. A student who engages in professional misconduct (see Section I.B.) may be subject to academic sanctions including but not limited to a grade reduction, failing grade, probation, suspension or dismissal from the program or the University, revocation of a student's degree or certificate, or comparable professional credentialing sanctions. Sanctions may also include community service, a written reprimand, and/or a written statement of misconduct that can be put into an appropriate record maintained for purposes of the profession or discipline for which the student is preparing.

    0. Any person who observes or discovers that a student has engaged in professional misconduct should file a written complaint with the office of the dean of the college within forty-five (45) business days of the date of discovery of the alleged violation.

    1. Upon receipt of the complaint, the dean of the college shall notify the department chair or program director, and within a reasonable time discuss the alleged misconduct with the accused student and give the student an opportunity to respond. The dean of the college may interview the complaining party and any other persons believed to have pertinent factual knowledge of the allegations. The dean of the college may also review any other relevant evidence, including documentary evidence. The dean may delegate the above responsibilities to a designee, who will report his/her findings to the dean.

    2. The dean of the college shall determine whether there is a reasonable basis to believe that the student engaged in professional misconduct.

    3. If the dean of the college determines that there is no reasonable basis to believe that the student engaged in professional misconduct, the dean of the college, or designee, shall, within twenty (20) business days of receipt of the complaint, notify the student and the matter will be dismissed.

    4. If the dean of the college determines that there is a reasonable basis for believing that the student engaged in professional misconduct, he/she shall determine whether efforts at informal resolution are appropriate and, if so, shall take whatever steps are useful to that end within twenty (20) business days of receipt of the complaint. If an informal resolution is reached and the responding student complies with the terms and conditions of the

resolution, no further action against the student will be taken and the matter will be closed.

5. If informal resolution is inappropriate, or if efforts at informal resolution are not successful, the dean of the college shall, within twenty (20) business days of receipt of the complaint, refer the complaint, including his/her recommendation for academic sanctions, to the Academic Appeals Committee for proceedings in accordance with Section C, below, and so notify the student in writing.

B. Proceedings Before the Academic Appeals Committee

0. Written Complaint and Recommendations. The written complaint and recommendations shall be delivered to the chair of the Committee, with a copy to the student.

1. Response to Complaint and Recommendations. The student responding to the complaint and recommendations may deliver his/her response to the chair of the Academic Appeals Committee, with a copy to the dean, no later than five (5) business days after receipt of the complaint and recommendations.

2. Makeup of the Committee. The dean of each college shall ensure that an Academic Appeals Committee(19) is constituted according to college procedures, subject to the following parameters. Two faculty members shall come from the college. The Personnel and Elections Committee of the Academic Senate shall appoint one faculty member from outside the college. The faculty members shall be appointed to the Committee for staggered three-year terms. The dean shall appoint two undergraduate student members and two graduate student members who are either from the relevant Student Advisory Committee or listed as a major within the college. Undergraduate student and graduate student members will be appointed for staggered two-year terms(20). No more than one faculty member and two Committee members in total may come from the same department in a multi-department college. The members of the Committee who shall hear the case are the three faculty members and the two students from the peer group of the student accused of professional misconduct (i.e., undergraduates or graduates). The dean shall designate one of the faculty members to serve as chair of the Committee. The Committee shall establish internal procedures consistent with the Student Code.

3. Conflicts of Interest. Upon written request of one of the parties or Committee members, the dean may excuse any member of the Committee if the dean determines that the member has a conflict

of interest. The dean shall select an appropriate replacement for the excused member (i.e., student or faculty member).

4. Scheduling Hearings Before the Committees. When a complaint and recommendations together with a response are filed in a timely manner, the Committee chair shall schedule a hearing date if:

   . The documents raise material issues of disputed fact;

   a. The Committee chair determines that a hearing is necessary or otherwise desirable to aid in the resolution of the issues; or

   b. The possible sanctions against the student may include dismissal from the University, dismissal from a program, suspension from either for longer than ten (10) business days, or revocation of the student's degree or certificate.

      0. If the Committee chair determines that no circumstances exist that require a hearing, as provided above, the chair shall notify the student and the dean of the college (the parties) in writing of this determination and within a reasonable time convene a closed meeting of the Committee to consider the documentation submitted by the parties. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the cognizant senior vice president, or designee, within ten (10) business days after the Committee meeting.

5. Notice of Hearings Before Committees. If the Committee chair determines that a hearing is required, the chair shall schedule a hearing date and notify the parties in writing of the date of the hearing, the names of the Committee members, and the procedures outlined below at least fifteen (15) business days prior to the hearing.

6. Hearing Procedures. Hearings shall be conducted according to the following procedures:

   . Hearings shall be conducted within a reasonable time after the Committee's receipt of the written complaint and recommendations and the response.

   a. At least five (5) business days prior to the date of the hearing, the parties shall make available to each other and to the Committee a list of their witnesses and a list of the documents to be offered at the hearing. In exceptional circumstances, the Committee may allow a party to call witnesses not listed or submit additional documents at the hearing.

b. The parties have a right to be accompanied by any person as advisor, including legal counsel, who will be permitted to attend, but not directly participate in, the proceedings.

c. Hearings shall be closed to the public.

d. All hearings, except Committee deliberations and voting, shall be recorded and a copy made available to any party upon request. Committee deliberations and voting shall take place in closed sessions.

e. The Committee must have a quorum present to hold a hearing. A quorum consists of three (3) members, including at least one (1) student and the faculty member from outside the college. If there is more than one hearing in a matter, or if the hearing continues over more than one session, the same three members must be present for all sessions. All findings and recommendations of the Committee shall require a majority vote of the Committee members present at the hearing.

f. At the hearing, the parties shall have the right to present questions to witnesses through the Committee chair, to present evidence and to call witnesses in their own behalf, in accordance with the Committee's established internal procedures.

g. The Committee shall not be bound by strict rules of legal evidence or procedure and may consider any evidence it deems relevant.

h. University legal counsel shall serve as a resource to the Committee and may be present at the hearing to provide guidance on substantive law and procedural matters.

i. In the hearing, the Committee must determine, by a preponderance of the evidence, whether the student engaged in the alleged professional misconduct. If the Committee answers this question in the affirmative, the Committee may then recommend any academic sanction it deems appropriate under the entire circumstances of the case.

j. The Committee shall make its findings and recommendations based only on evidence and testimony presented by the parties at the hearing. Committee members shall not conduct their own investigations, rely on prior knowledge of the facts or develop their own evidence.

k. If either party presenting to the Academic Appeals Committee fails to attend the hearing without good cause, the Committee may proceed with the hearing and take testimony and evidence and report its findings and recommendations to either the senior vice president for academic affairs, or senior vice president for health sciences, as appropriate, on the basis of such testimony and evidence.

l. The Committee chair shall prepare a written report of the Committee's findings and recommendations and present it to the cognizant senior vice president within ten (10) business days after the conclusion of the hearing.

C. Review and Decision by the Cognizant Senior Vice President

0. The vice president shall consider the documentation submitted to the Committee and the findings and recommendations of the Committee in making a decision. Based upon such review, and without conducting further hearings, the vice president shall, within ten (10) business days, take one of the following actions:

   . Accept the Committee's findings and recommendations;

   a. Return the report to the Committee chair, requesting that the Committee reconvene to reconsider or clarify specific matters, materials, and issues, and forward to the vice president a second report of its findings and recommendations relating to the specific matters referred by the vice president for further consideration; or

   b. Reject all or parts of the Committee's findings and recommendations, stating reasons and actions to be taken therefore. The vice president may impose greater or lesser sanctions than recommended by the Committee.

1. Written notification of the vice president's decision shall be communicated to the parties, to the chair of the Academic Appeals Committee and to the president within ten (10) business days of receipt of the Committee's findings and recommendations.

2. The vice president's decision is final unless appealed to the president within ten (10) business days of receipt of the decision.

D. Appeal to President

    0. Within ten (10) business days of receipt of the vice president's decision, any party may appeal the decision by filing a written notice of appeal with the president and delivering a copy to the other party. The other party may file a response to the appeal with the president within five (5) business days of receipt of the appeal. In the case of an appeal:

        0. The president shall consider the appeal and response to the appeal and may solicit whatever counsel and advice the president deems appropriate to arrive at a final decision. The president may also convene an ad hoc committee composed of students and faculty members from outside the college or department to determine if there were substantial defects that denied basic fairness and due process. After considering the appeal, the president shall, within ten (10) business days, or within twenty (20) business days if an ad hoc committee is formed, take one of the following actions:

            . Accept the decision of the vice president;

            a. Return the report to the vice president, requesting that he/she clarify specific matters, materials, and issues, and forward to the president a second report of his/her decision relating to the specific matters referred by the president for further explanation; or

            b. Reject all or parts of the vice president's decision, stating reasons and actions for imposing a greater or lesser sanction than determined by the vice president.

        1. Written notification of the president's decision and the basis for that decision shall be communicated to the student, to the academic dean or dean's designee, to the vice president, and to the chair of the Academic Appeals Committee within ten (10) business days after receipt of the appeal, or within twenty (20) business days after receipt of the appeal if an ad hoc committee is formed.

        2. The decision of the president is final.

E. Suspension or Dismissal from a Program or from the University, and Revocation of a Degree or Certificate

    0. The sanctions of suspension, dismissal, and revocation for professional misconduct may be imposed: (1) if agreed upon in

informal resolution between the responding student and the dean of the college; (2) if recommended by the Academic Appeals Committee to the cognizant vice president; (3) by the vice president notwithstanding the recommendation from the committee; or (4) by the president notwithstanding the decision of the vice president. A student who has been suspended or dismissed from the University shall be denied all privileges accorded to a student.

0. Suspension from a Program or from the University.

   . Suspension shall be for a minimum time of one semester following the semester the student is found responsible for professional or academic misconduct.

   a. The dean of the relevant college shall notify the student in writing of the suspension, conditions for reinstatement, and of the obligation of the student to petition for reinstatement.

   b. Petitions for reinstatement shall be submitted to the relevant dean and shall explain how the conditions for reinstatement have been met.

   c. The relevant dean shall consider the petition and shall issue a decision regarding the student's reinstatement within fifteen (15) business days of receipt of the petition.

   d. The relevant dean may grant conditional reinstatement contingent upon the student meeting written requirements specified in the original sanction.

   e. The notice of the dates for which the student is suspended will remain on his/her transcript until he/she has been reinstated to the program or to the University, or for five (5) years if he/she is not reinstated to the program or to the University. [SeeProcedure 6-400-Sec.VII #1]

1. Dismissal from a Program or from the University.

   . Dismissals from a program or from the University are final. A student dismissed from a program or from the University for professional misconduct may not petition for reinstatement.

a. Permanent records of dismissal shall be kept in the office of the registrar.

b. The dismissed student's transcript will reflect his/her dismissal. [See Procedure 6-400-Sec.VII #1]

c. Dismissal should be reserved for only the most egregious of offenses.

2. Revocation of a Degree or Certificate.

. Decisions to revoke a degree or certificate are final.

a. Permanent records concerning the revocation of a degree or certificate shall be kept in the office of the registrar.

b. The revocation of a degree or certificate shall be noted on the student's transcript. [See Procedure 6-400-Sec.VII #1]

c. Revocation of a degree or certificate should be reserved for only the most egregious of offenses.

F. Internal Reporting of Professional Misconduct

0. The dean shall take appropriate action to implement the final decision. If the student is found responsible for professional misconduct, the dean shall notify, in writing, the student's department or program of study of the violation, the proceedings, and the final decision. If the sanction involves suspension or dismissal from a program or from the University or revocation of a degree or certificate, the dean shall also convey the decision to the office of the registrar for notation on the transcript. [See Procedure 6-400-Sec.VII #1]

G. Administrative Suspension to Protect the University Community or the Public

0. The senior vice president for academic affairs (or designee) or the senior vice president for health sciences (or designee) may suspend a student from the University prior to an initial inquiry and hearing before the Academic Appeals Committee if such action appears necessary to protect the health or well-being of any member of the University community, any member of the public or to prevent serious disruption of the academic process. Prior to, contemporaneous with, or immediately after the suspension, the vice president shall give the student written

notice of the suspension specifying the alleged misconduct and setting forth briefly the relevant facts and supporting evidence. The vice president shall then provide the student with an opportunity to meet with him/her to present the student's views and object to the suspension. This meeting shall take place prior to the suspension taking effect or as soon as possible thereafter. The vice president shall thereafter immediately refer the complaint to the appropriate University administrator for proceedings under the code, and the suspension will be in effect pending a final determination of the matter. The vice president shall notify other University administrators of the suspension as appropriate.

H. Reporting of Professional Misconduct

0. No University employee shall provide information to a person or entity concerning a student's professional misconduct without fully complying with The Family Educational Rights and Privacy Act (20 U.S.C.A. § 1232g) and the Government Records Access and Management Act (Utah Code Title 63G - Chapter 2). In most circumstances, such as requests from a licensing body or an employer, information may only be provided with the prior written consent of the student. In some circumstances, however, such as requests from other institutions where the student seeks or intends to enroll, information may be provided without the consent of the student but only after following appropriate procedures outlined in the statutes.

I. Other University Proceedings

0. If the filing of a complaint or an appeal relating to professional misconduct under the Student Code raises other issues concerning behavioral or academic misconduct, the cognizant senior vice president, or designee, the dean of students, and the involved University administrator shall determine the appropriate procedure(s) for processing the complaint or the appeal.

J. Retention of Records of Proceedings

0. Records of proceedings under the Student Code shall be confidential to the extent permitted by law. Records of professional misconduct shall be kept in the office of the registrar, and a copy may be maintained in other academic departments as appropriate.

7. Section VII: Student Records

. General

0. The privacy and confidentiality of all student records shall be preserved as outlined in relevant federal and local laws (i.e. The Family Educational Rights and Privacy Act (20 U.S.C.A. § 1232g) and the Government Records Access Management Act (Utah Code Title 63G - Chapter 2). University interpretation of the Family Educational Rights and Privacy Act as it pertains to University of Utah students is available from the office of the vice president for student affairs.

1. Official student records shall be maintained only by members of the University staff employed for that purpose. Separate record files may be maintained under the following categories: (i) academic, academic counseling, financial aid, and placement; (ii) disciplinary; (iii) medical, psychiatric, and health counseling. When justified by legitimate law enforcement needs, the campus security agency may maintain confidential records relating primarily to its investigative function.

A. Access and Challenge of Accuracy of Records

0. Access to the student's official records and files is guaranteed every student subject to the limitations set forth in relevant federal and local laws (i.e. The Family Educational Rights and Privacy Act (20 U.S.C.A. § 1232g) and the Government Records Access and Management Act (Utah Code Title 63G - Chapter 2). Students with complaints, inquiries, or requests for review of official records are directed to the vice president for student affairs.

B. Matters Prohibited in Official Records

0. Except as required by law or governmental regulations or as authorized by written consent of the student involved, official student records will not contain information regarding a student's race, religion, disability, political opinions, social opinions, or membership in any organizations other than honorary and professional organizations directly related to the educational process. Except as required by law or applicable governmental or University regulations, information regarding marital status shall not be included in the official student records of any student who has filed a written objection to the inclusion of that information in his/her records and has not filed a subsequent written revocation thereof.

C. Official Disciplinary Records

0. Records of behavioral or academic sanctions imposed by the Student Behavior Committee, by the Academic Appeals Committee, or by any authorized official of the University shall be

maintained in the office of the dean of students and/or the office of the registrar. Records of behavioral, academic or professional misconduct may also be maintained in the official files of a department or program, and by the senior vice president for academic affairs or senior vice president for health sciences. No notation of behavioral or academic sanctions shall be entered or made on the student's academic transcripts except in the following circumstances: 1) when the student is suspended from a program or from the University for academic or professional misconduct; 2) when the student is dismissed from a program or from the University for behavioral, academic or professional misconduct; or 3) when the student's degree or certificate has been revoked. In a case of dismissal, suspension, or revocation, the entry on the transcripts of the student shall merely state: "Dismissed from the University for Behavioral Misconduct" or "Dismissed/Suspended from the [program]/University for Academic/Professional Misconduct" or "Degree/Certificate Revoked for Academic/Professional Misconduct" and the date of such action. Notices of dismissal or revocation shall not be removed from the student's academic transcripts. Notices of suspension shall be entirely removed from the student's academic transcripts after the student is reinstated in the program or at the University. If the student is not reinstated due to his/her failure to fulfill the conditions of the suspension, the notice shall be removed five (5) years after the suspension is first imposed. [See Procedure 6-400-Sec.VII #1]

D. Confidential Character of Student Records

0. The University must conform to the requirements of the statutes referred to in Subsection A "General" and Subsection B "Access to and Challenge of Accuracy of Records" forbidding the release of personally identifiable student education records or files, or personal information contained therein, without the written consent of the student. Subject to applicable legal requirements, it is the policy of the University that:

    0. Members of the administration and the instructional staff will have access to student records for legitimate purposes such as student advising, administrative planning and statistical reporting.

    1. Directory information, such as the student's name, address, telephone number, date and place of birth, major field of study, participation in officially recognized activities or sports, weight and height of members of athletic teams, dates of attendance, degrees and awards received, the most recent previous educational agency or institution

attended by the student, current semester class schedule, and other similar information may be disclosed to an inquirer unless the student specifically withholds permission to do so.

2. Authorized representatives of federal and state governments may have access to student records to the extent necessary for audit and evaluation of federally supported education programs or of compliance with federal legal requirements relating to such programs, and subject to the limitation that personally identifiable data shall not be disclosed except to the extent specifically authorized by federal law.

3. The right of access to a student's records without the consent of the student is not extended to the parents of the student unless the student has been established as a “dependent” as defined in Section 152 of the Internal Revenue Code of 1954.

4. Records created or maintained by a physician, psychologist, or other recognized professional or para-professional acting in that capacity, which are created, maintained, and used only in connection with treatment of a student are not available for review except by an appropriate professional of the student's choice, or in compliance with an order from a court of competent jurisdiction.

E. Treatment of Official Records Following Graduation or Withdrawal

0. Upon graduation or withdrawal from the University, the official records of former students shall continue to be subject to the provisions of this code.

0.

---

1. [Note: Parts IV-VII of this Regulation (and all other University Regulations) are Regulations Resource Information – the contents of which are not approved by the Academic Senate or Board of Trustees, and are to be updated from time to time as determined appropriate by the cognizant Policy Officer and the Institutional Policy Committee, as per Policy 1-001 and Rule 1-001.]

8. Rules, Procedures, Guidelines, forms and other related resources.

0. Rules (reserved)

1. Procedures
    0. University Procedure 6-400-Sec.VII #1
2. Guidelines (reserved)
3. Forms (reserved)
4. Other related resource materials (reserved)

9. Contacts

0. The designated contact officials for this Policy are:
    . Policy Owner (primary contact person for questions and advice): Dean of Students.

    A. Policy Officers: Sr. Vice President for Academic Affairs and the Sr. Vice President for Health Sciences.
        0. These officials are designated by the University President or delegee, with assistance of the Institutional Policy Committee, to have the following roles and authority, as provided in University Rule 1-001:
        1. "A 'Policy Officer' will be assigned by the President for each University Policy, and will typically be someone at the executive level of the University (i.e., the President and his/her Cabinet Officers). The assigned Policy Officer is authorized to allow exceptions to the Policy in appropriate cases.... "
        2. "The Policy Officer will identify an 'Owner' for each Policy. The Policy Owner is an expert on the Policy topic who may respond to questions about, and provide interpretation of the Policy; and will typically be someone reporting to an executive level position (as defined above), but may be any other person to whom the President or a Vice President has delegated such authority for a specified area of University operations. The Owner has primary responsibility for maintaining the relevant portions of the Regulations Library... .[and] bears the responsibility for determining -requirements of particular Policies... ." University Rule 1-001-III-B & E

10. History:

0. Renumbering: Renumbered as Policy 6-400 effective 9/15/2008, formerly known as PPM 8-10, and formerly as University Regulations Chapter X.

1. Revision History:
    0. Current version: Revision 8
        0. Editorially revised: July 9, 2009
    1. Earlier versions:
            0. Revision 7:effective dates July 1, 2009 to July 8, 2009
                1. Legislative History of Revision 7

    Revision 6:
        effective dates February 3, 2006 to July 1, 2009.
        1. Revision 5: effective dates May 10, 2004 to February 2, 2006
        2. Revision 4: effective dates February 10, 2003 to May 9, 2004
        3. Revision 3: effective dates July 14, 1997 to February 9, 2003

1. Allegations of sexual harassment generally will be handled by OEO/AA in accordance with Policy and Procedures 5-210. However, allegations of student to student sexual harassment may be handled under the Student Code, rather than by the office of OEO/AA.
2. Oral complaints presented to the dean of students shall be recorded by the dean's office either electronically or in transcribed form.
3. The parties to a complaint before the Student Behavior Committee are the responding student, the complaining party, and the dean of students.
4. If the academic action results from a decision of a committee, e.g., the Promotions Committee of the School of Medicine, the chair of the committee is the “faculty member” for purposes of these procedures.
5. In colleges without departments, the student shall appeal in writing to the dean of the college. The dean of the college shall appoint one or more faculty members from the college to serve as chair for purposes of these procedures. In cases where the appeal occurs in a program that does not report directly to an academic dean, but rather to an associate vice president, the cognizant program director shall serve as department chair, and the cognizant associate vice president shall serve as dean for purposes of these procedures.
6. Colleges or departments offering only graduate programs may appoint only graduate student members.
7. Claims of misconduct in sponsored research will be handled in accordance with Policy and Procedures 7-001. In addition, such claims may also be consider under this code.
8. If a student attempts to withdraw from a course after engaging in academic misconduct, withdrawal may be denied by the University whether or not the attempt is made before the official withdrawal date and a failing grade may be imposed for the course.
9. In colleges without departments, the faculty member shall notify the dean of the college.
10. If the student's home department is unknown or undecided, the faculty member should report the academic misconduct to the senior vice president for academic affairs or the senior vice president for health sciences and the Associate Dean for Advising, University College.
11. See FN 10.
12. If the student's home college is unknown or undecided, the person pursuing the complaint should report the academic misconduct to the senior vice president for academic affairs, or the senior vice

president for health sciences. The action for misconduct may then be pursued through the Academic Appeals Committee of the college offering the course.
13. If the student appeals a failing grade or other lesser sanction imposed for the last act of misconduct, the dean or vice president for the student's home college may delay action under this section until ten (10) business days following notice of the determination on the student's appeal.
14. If the student's home college is unknown or undecided, proceedings for misconduct should be pursued through the Academic Appeals Committee of the college in which the last act of misconduct occurred.
15. See FN 6.
16. See FN 10.
17. In cases where the dean recommends a sanction of suspension or dismissal from the University or revocation of a degree or certificate, which sanction is implemented by the cognizant vice president, the appeal shall be made directly to the president of the University.
18. See FN 10.
19. When necessary to comply with accreditation or licensing standards, a department may establish a departmental Academic Appeals Committee in lieu of the college Academic Appeals Committee to hear allegations of professional misconduct. The departmental committee shall be composed of two faculty members and two students from the department (or professional program within the department) and one faculty member from outside the department. Hearings by the departmental committee shall be conducted in accordance with the procedures established in Part VI.C, for the college Academic Appeals Committee.
20. See FN 6.

Appendix F: Computer Science Undergraduate Track Elective Suggestions

Computer Science Undergraduate Track Elective Suggestions

3/26/15

Undergraduate certificates awarded upon graduation if required number of courses are taken in a specific track area (optional).

Software

Software Development *(Choose 5)*
3470: Scripting Language/Design
4230: Parallel Programming
4480: Computer Networks
4540: Web Software Architecture
5140: Data Mining
5460: Operating Systems
5470: Compilers
5530: Database Systems
5540: Human Computer Interaction
5785: Adv. Embedded Software

Web/Mobile Development *(Choose 4)*
3470: Scripting Language/Design
4480: Computer Networks
4540: Web Software Architecture
4962: iPhone/Android Development
5530: Database Systems
5540: Human Computer Interaction

Computer Systems

Computer Systems *(Choose 3)*
4230: Parallel Programming
4480: Computer Networks
5460: Operating Systems
5470: Compilers
5490: Network Security
5530: Database Systems

Programming Languages *(Choose 3)*
3470: Scripting Language/Design
5100: Foundations of CS
5470: Compilers
5510: Programming Languages

Artificial Intelligence

Robotics *(Choose 3)*
4300: Artificial Intelligence
5310: Robotics
5350: Machine Learning
*6320: 3D Computer Vision
*6330: Intro to Robot Control
*6370 Geometric Motion Planning

Artificial Intelligence *(Choose 4)*
4300: Artificial Intelligence
4640: Image Processing Basics
5100: Foundations of CS
5130: Computational Statistics
5140: Data Mining
5320: Computer Vision
5340: Natural Language Processing
5350: Machine Learning

Data

Information *(Choose 3)*
4300: Artificial Intelligence
5140: Data Mining
5340: Natural Language Processing
5350: Machine Learning
5530: Database Systems
5630: Visualization
*6150: Adv. Algorithms

Theory

Theory *(Choose 3)*
3100: Models of Computation
5100: Foundations of CS
5130: Computational Statistics
5140: Data Mining
5350: Machine Learning
*6150: Adv. Algorithms

Graphics

Visual Computing *(Choose 4)*
3200: Intro Sci Comp
4600: Intro Computer Graphics
4640: Image Processing Basics
5320: Computer Vision
5350: Machine Learning
5610: Interactive Comp Graphics
5630: Scientific Visualization
5650: Perception for Graphics

Hardware

Computer Organization *(Choose 4)*
3700: Digital System Design
3710: Computer Design Lab
5460: Operating Systems
5710: Digital VLSI Design
5830: VLSI Architecture

Embedded Systems *(Choose 4)*
3710: Computer Design Lab
4480: Computer Networks
5470: Compilers
5780: Embedded System Design
5785: Adv. Embedded Software
5789: Embedded Sy/Kinetic Art

CAD for Digital Systems *(Choose 4)*
5710: Digital VLSI Design
5740: Computer-Aidied Design
5745: Testing/Verif. Digital Circuits
5750: Synthesis/Veri. VLSI Sys.
5830: VLSI Architecture

*Open to undergrads with instructor consent & permission code.

Appendix G: Computer Science BS Degree Requirements

Student Name: ______________________________ uID: ______________

COMPUTER SCIENCE 2016-2017 B.S. Degree Requirements

CS undergraduate advising: ugrad-help@cs.utah.edu or 801-581-8224

PRE-MAJOR REQUIREMENTS:_____

C- or better in each course, and a minimum 3.0 average GPA (overall and within pre-major courses) required ***to apply*** *for full major status.*

1. CS 1030, Foundations of CS________________________(3)
2. CS 1410, Object-Orient. Prog.______________________(4)
3. CS 2420, Algrthms/Data Struct.______________________(4)
4. Math 1210, Calculus I (QR) ________________________(4)
5. Math 1220, Calculus II (QR) _______________________(4)

GENERAL EDU. REQUIREMENTS:__

Honors options also accepted for WR2, CW, and AI requirements.

1. Wrtg 2010, Intermediate Writing (WR2) ______________(3)
2. Wrtg 3012 or 3014 or 3015 (CW) _________________(3)
3. American Institutions (AI) ______________________(3)

SIX ***Intellectual Exploration (IE) courses*** *required. TWO must be upper division (3000-level or above), ONE must satisfy the Diversity requirement and ONE must satisfy the International requirement.*

4. Fine Arts (FF): ______________________________(3)
5. Fine Arts (FF): ______________________________(3)

6. Humanities (HF): ____________________________(3)
7. Humanities (HF): ____________________________(3)

8. Social/Behavioral Science (BF): __________________(3)
9. Social/Behavioral Science (BF): __________________(3)

- Upper Division (3000+ level IE) ____________
- Upper Division (3000+ level IE) ____________
- Diversity (DV) ____________
- International (IR) ____________

MATH / SCIENCE ELECTIVES:_____

C- or better required in all math/ science courses. PHYS 2210 Required. CHOOSE 2 of 3: Math 2210-Calc III, Math 2270-Linear Algebra, CS 3130-Eng. Prob. Stats.

TWO additional electives must be 3+ credits each, as follows:

Accepted: Math, science or engineering courses with Math 1220 as a pre- or co-requisite (See DARS). Biol 1210, Chem 1210 also accepted.

NOT Accepted: CS courses (except CS 3130). Math 2200, Math 3010. Math 2250 not accepted if Math 2270 and/ or Math 2280 are taken. Math 5010 and/or 3070 not accepted if CS 3130/ ECE 3530 is taken.

1. Physics 2210, Physics I_________________________(4)
2. Choose 1: Math 2210 (QR), Math 2270 (QR) or CS 3130_( )
3. Choose 1: Math 2210 (QR), Math 2270 (QR) or CS 3130_( )
4. ____( )
5. ____( )

The following requirements are restricted to FULL Majors:

C- or better required in all CS courses. CR/NC grading option not allowed for any major requirement. 2.5 GPA (overall & CS courses) required to graduate.

MAJOR REQUIREMENTS:____________

1. CS 2100, Discrete Structures__________________________(3)
2. CS 3500, Software Practice I__________________________(4)
3. CS 3505, Software Practice II_________________________(3)
4. CS 3810, Computer Organization (QI) ___________________(4)
5. CS 4150, Algorithms (QI) ____________________________(3)
6. CS 4400, Computer Systems (QI) ______________________(4)

CS ELECTIVES:_________________

Choose 7 total CS courses, 3000-level or above, 3-4 credits each. Seminars, CS 3992, CS 3130 not accepted.

1. CS __________/____________/______________( )
2. CS __________/____________/______________( )
3. CS __________/____________/______________( )
4. CS __________/____________/______________( )
5. CS __________/____________/______________( )
6. CS __________/____________/______________( )
7. CS __________/____________/______________( )

No more than 3 of the following may be accepted above as CS electives:

- *(1) CS 4010, Internship*
- *(1) CS 4940, Research (if not used for capstone)*
- *(1) CS 4950, Independent Study*
- *(1) EAE course (3000+ level, 3+ credits)*
- *(1) Combination of 1-2 credit CS courses (3 credits total): CS 3011, 3020, 4190, 5040 and 1-2 credit special topics courses*

THEORY RESTRICTED ELECTIVE_____

Choose ONE: (If both classes are taken, one will count as a CS elective above)

CS 3100, Models of Computation (QI) ______________________(3)

or

CS 3200, Scientific Computing____________________________(3)

CAPSTONE REQUIREMENT:__________

Choose ONE set: (Permission required from Undergraduate Director for thesis)

CS 4000, Senior Capstone Design _________________________(3)
CS 4500, Senior Capstone Project _________________________(3)

or

CS 4940, Undergraduate Research _________________________(3)
CS 4970, Bachelor's Thesis _____________________________(3)

See the CS Undergraduate Handbook online for complete details, restrictions & requirements
Updated 5/10/16

Appendix H: Computer Science BS Degree Requirements EAE

COMPUTER SCIENCE 2016-2017 B.S. Degree Requirements
Entertainment Arts & Engineering (EAE) emphasis

CS undergraduate advising: ugrad-help@cs.utah.edu or 801-581-8224

PRE-MAJOR REQUIREMENTS:_____

C- or better required and a minimum 3.0 average GPA (overall and within pre-major courses) required ***to apply*** *for full major status.*

1. EAE 1030, Foundations of CS________________________(3)
2. EAE 1410, Object-Orient. Prog.______________________(4)
3. EAE 2420, Algrthm/Data Struct.______________________(4)
4. Math 1210, Calculus I (QR) _________________________ (4)
5. Math 1220, Calculus II (QR)_________________________ (4)

GENERAL EDU. REQUIREMENTS:__

Honors options also accepted for WR2, CW, and AI requirements. See minimum grade requirements in handbook.

1. Wrtg 2010, Intermediate Writing (WR2) ________________(3)
2. FA 3600, Writing for New Media (CW) _________________(3)
3. American Institutions (AI) __________________________(3)

Six ***Intellectual Exploration (IE) courses*** *required. TWO must be upper division (3000-level or above), ONE must satisfy the Diversity requirement and ONE must satisfy the International requirement.*

4. **ART 1020,** Basic Drawing (FF): ____________________(3)
5. Fine Arts (FF): ________________________________(3)

6. Humanities (HF): _____________________________(3)
7. Humanities (HF): _____________________________(3)

8. Social/Behavioral Science (BF): ___________________(3)
9. Social/Behavioral Science (BF): ___________________(3)

- Upper Division (3000+ level IE) ______________
- Upper Division (3000+ level IE) ______________
- Diversity (DV) ___________________
- International (IR) ___________________

MATH / SCIENCE ELECTIVES:______

C- or better required in all math/ science courses.

ONE additional math/science elective is required (3+ credits). Choose any non-CS, math or science class with Math 1220 (Calculus II) as a pre- or co-requisite. Physics 2220 will also be accepted. ***Math 2200, 3010, 5010, 3070 not allowed. Math 2250 not accepted if Math 2270 is taken.***

1. Physics 2210, Physics I___________________________(4)
2. Choose 1: Math 2210 (QR) or Math 2270__________(3)
3. CS 3130, Eng Prob & Stats (QI) ____________________(3)
4. ____( )

FILM REQUIREMENTS:___________

C- or better required in all FILM courses.

1. FILM 2700, Intro to Video Games __________________(3)
2. FILM 3500, Film Production ______________________(4)

The following requirements are restricted to FULL Majors:

C- or better required in all CS, EAE & Film courses. CR/NC grades not allowed for any major requirement. 2.5 GPA (overall & within CS) required to graduate.

MAJOR REQUIREMENTS:_____________

1. CS 2100, Discrete Structures____________________________(3)
2. CS 3500, Software Practice I____________________________(4)
3. CS 3505, Software Practice II___________________________(3)
4. CS 3810, Computer Organization (QI)______________________(4)
5. CS 4150, Algorithms (QI)_______________________________(3)
6. CS 4400, Computer Systems (QI)__________________________(4)

EAE REQUIREMENTS:________________

1. EAE 3600, 3D Modeling ______________________________ (3)
2. EAE 3660, Machinima ________________________________ (3)
3. CS 4300, Artificial Intelligence _________________________(3)
4. CS 5530, Databases __________________________________ (3)

5. *Choose ONE: (If both classes are taken, one counts as a CS elective below)*

   CS 5460, Operating Systems ___________________________(4)

   or

   CS 5470, Compilers _________________________________(4)

CS ELECTIVE_______________________

Choose TWO: 4000+ level CS courses (3-4 cr). Seminars and EAE courses not accepted. (Suggested: CS 4480, 4540, 4600, 5350, 5630)

1. ____(3)
2. ____(3)

SERIES REQUIREMENT:_______________

Choose ONE:

ANIMATION SERIES

FILM 3610, Computer Animation I ______________________ (4)
FILM 3620, Computer Animation II _____________________(4)

or

GAME DESIGN SERIES

FILM 3710, Traditional Game Development ________________ (4)
FILM 3720, Alternative Game Development ________________(4)

THEORY RESTRICTED ELECTIVE______

Choose ONE:

CS 3100, Models of Computation (QI)______________________(3)

or

CS 3200, Scientific Computing_____________________________(3)

CAPSTONE REQUIREMENT____________

1. EAE 4500, Senior Project I ____________________________ (3)
2. EAE 4510, Senior Project II ___________________________ (3)

Appendix I: Sample for Computer Engineering Degree Program

Sample Computer Engineering Degree Program 2015-2016

First Year

Fall Semester			Spring Semester		
ECE 1900	0.5	Freshman Seminar	ECE 1250‡	4.0	Electrical & Computer Engineering Design
CS 1410-030	4.0	Object Oriented Programming	CS 2420†	4.0	Intro Alg & Data Structures
Math 1310*‡	4.0	Engineering Calculus I	Math 1320*‡	4.0	Engineering Calculus II
Wrtg 2010‡	3.0	Intermediate Writing			
or EAS 1060		Expository Writing for EAS			
LEAP 1501	3.0	Ethical Implications of Engineering	Phys 2210‡	4.0	Physics for Scientists & Engineers I
	14.5			16.0	

* Students must take Math 2210 Calculus III as a Math/Science technical elective if took Math 1210, and 1220.

Second Year

Fall Semester			Spring Semester		
ECE 2240†	4.0	Introduction toElectric Circuits	ECE 2280†	4.0	Fundamentals of Engineering Electronics
CS 3500†	4.0	Software Practice I	CS/ECE 3700	4.0	Digital System Design
CS/ECE 3810†	4.0	Computer Organization	Math 2250‡	4.0	Differential Equat & Linear Algebra
Phys 2220‡	4.0	Physics for Scientists & Engineers II	Math/Science	3/4	Elective
	16.0			15/16	

Third Year

Fall Semester			Spring Semester		
CS/ECE 3710	3.0	Computer Design Laboratory	CS/ECE 3992	3.0	Pre-Thesis/Pre-Project
CS/ECE 3991	1.0	CE Junior Seminar	CS/ECE 5780	4.0	Embedded System Design
CS 2100†	3.0	Discrete Structures	ECE 3530†	3.0	Eng Probability & Statistics
CS 4400	4.0	Computer Systems	Gen Ed	3.0	Elective
ECE 3030†	3.0	Tech Comm & Wrtg for Engineers	Gen Ed	3.0	Elective
Gen Ed	3.0	Elective		16.0	
	17.0				

Fourth Year

Fall Semester			Spring Semester		
CS/ECE 4710	3.0	CE Senior Project	CE	3.0	Technical Elective
CE	3.0	Technical Elective	CE	3.0	Technical Elective
CE	3.0	Technical Elective	CE	3.0	Technical Elective
Gen Ed	3.0	Elective	CE	3.0	Technical Elective
Gen Ed	3.0	Elective		3.0	American Institutions
	15.0			15.0	

This table gives an eight (8) semester example program leading to a B.S. in Computer Engineering. It is meant only as a guide, since the scheduling of electives and General Education Classes depends upon which ones are selected. This schedule assumes adequate high school preparation in mathematics.

Students may apply for major status during any semester in which all pre-major classes (highlighted above) are completed. The current GPA for admission to major status is 2.5 for all University of Utah classes, and a 2.8 on for all pre-major classes. Students must also have a minimum grade of "C-" in all pre-major courses. Apply for the CE major at http://www.ece.utah.edu/forms .

†These classes are taught both fall and spring semesters.
‡These classes are taught all semesters. Summer classes will only be taught if minimum enrollment is met.

Visit the Computer Engineering website at www.ce.utah.edu

Appendix J: Minor in Computer Science Requirements

Minor in Computer Science

2016-2017

A minimum grade of C- or better in each course and a 3.0 GPA (overall and within the pre-minor classes) is required in order ***to apply*** to the CS minor. Please note that the 3.0 GPA is the minimum requirement to apply, and may not result in placement as a minor. See the CS undergraduate handbook online for full details.

Pre-Minor Requirements

*CS 1030: Foundations of Computer Science *(3 credits)*
CS 1410: Object-Oriented Programming *(4 credits)*
CS 2420: Algorithms & Data Structures *(4 credits)*
Math 1210: Calculus 1 *(4 credits)*

CS Minor Requirements:

CS 2100: Discrete Structures *(3 credits)*
CS 3500: Software Practice *(4 credits)*
CS Elective Course 3000+ *(3 credits)*

Total ≈ 25 Credits

*CS 1030 may be waived by test-out if student has sufficient, prior programming experience. See www.cs.utah.edu/undergraduate for details.

A minimum of three CS required minor courses must be taken from the School of Computing at the University of Utah.

Computer Engineering (CE) majors are not eligible for a CS minor, and should instead consider a double major with computer science.

Applications to the minor accepted after completion of the pre-minor requirements. The application is available on our Web site www.cs.utah.edu. Major declaration is required before adding a minor.

If you have questions, please email ugrad-help@cs.utah.edu

Appendix K: Computer Science Suggested Plan

Computer Science Suggested Plan

Track A: Students must test out of CS 1030 and be Calculus-ready for Track A.

	Fall Semester			Spring Semester		
Freshman (28 credits)	CS 1410	Object Or Prog.	4	CS 2420	Data Str/ Algrthm	4
	Math 1210	Calc I	4	Math 1220	Calc II	4
	General Ed		3	Wrtg 2010	Writing	3
	General Ed		3	General Ed		3
Sophomore (29 credits)	CS 3500	Software Prac. I	4	CS 2100	Discrete	3
	CS 3810	Comp. Org.	4	CS 3505	Software Prac. II	3
	Math 2210	Calc III	3	CS elective		3
	American Institutions (AI)		3	CS elective		3
	General Ed		3			
Junior (33 credits)	CS 4400	Comp. Systems	4	CS Theory or CS elective		3
	CS elective		3	CS 4150	Algorithms	3
	CS elective		3	CS elective		3
	Math 2270 or CS 3130		4 or 3	Wrtg 3012, 3014 or 3015		3
	(IR)/ Upper Division		3	Phys 2210	Physics I	4
Senior (32 credits)	CS 4000 or CS 4940		3	CS 4500 or CS 4970		3
	CS Theory or CS elective		3	CS elective		3
	Math/ Science elective		4	Math/ Science elective		4
122 total credits	General Ed/ Upper Division		3	General Ed/ (DV)		3
	(Free elective if needed)		3	(Free elective if needed)		3

Track B: Students who test into CS 1030 and/ or may not be Calculus-ready in the fall will follow Track B.

	Fall Semester			Spring Semester		
Freshman (30 credits)	*CS 1030	Foundations of CS	3	CS 1410	Object Or Prog.	4
	*Math 1210 Calc I		4	Math 1220	Calc II	4
	General Ed		3	Wrtg 2010	Writing	3
	General Ed		3	General Ed		3
				General Ed/ (DV)		3
Sophomore (32 credits)	CS 2420	Data Str/ Algrthm	4	CS 3500	Software Prac. I	4
	Math 2210	Calculus III	3	CS 3810	Comp. Org.	4
	Phys 2210	Physics I	4	CS elective		3
	American Institutions (AI)		3	Math/ Science elective		4
				Wrtg 3012, 3014 or 3015		3
Junior (32 credits)	CS 2100	Discrete	3	CS 4150	Algorithms	3
	CS 3505	Software Prac. II	3	CS elective		3
	Math 2270 or CS 3130		4 or 3	CS elective		3
	CS elective		3	Math/ Science elective		4
	Gen Ed/ (IR)/ Upper Division		3	(Free elective if needed)		3
Senior (28 credits)	CS 4000 or CS 4940		3	CS 4500 or CS 4970		3
	CS Theory or CS elective		3	CS Theory or CS elective		3
	CS 4400	Comp. Systems	4	CS elective		3
122 total credits	CS elective		3	General Ed/ Upper Division		3
				(Free elective if needed)		3

* If you test out of CS 1030, but are not Calculus-ready, follow Track B. Replace CS 1030 with another gen ed in your first semester. Replace Calc I with your first level of math.

Choose CS 3100 (fall) *or* CS 3200 (spring) as the required theory course. Students are encouraged to take summer courses to ease the fall & spring semester schedule. See advisor for alternative schedule.

Appendix L: Computer Science EAE Plan

Computer Science EAE Plan

Track A: Students must test out of EAE 1030 and be Calculus-ready for Track A.

	Fall Semester	Spring Semester
Freshman (31 credits)	EAE 1410 Object Or Prog. 4 Math 1210 Calc I 4 Art 1020 Basic Drawing 3 General Ed 3	EAE 2420 Data Str/ Algrthm 4 Math 1220 Calc II 4 Wrtg 2010 Writing 3 Film 2700 Video Games 3 American Institutions (AI) 3
Sophomore (26 credits)	EAE 3600 3D Modeling 3 CS 3500 Software Prac. I 4 CS 3810 Comp. Org. 4 Math 2210 Calc III 3	EAE 3660 Machinima 3 CS 2100 Discrete 3 CS 3505 Software Prac. II 3 FA 3600 Wrtg New Media 3
Junior (34 credits)	CS Theory or CS elective 3 CS 4400 Comp. Systems 4 CS 3130 Eng Prob Stats 3 FILM 3500 Film Production 4 FILM 3610 or 3710 4	CS 4150 Algorithms 3 CS 5530 Database 3 CS elective 3 FILM 3620 or 3720 4 General Ed/ DV 3
Senior (33 credits) 124 total credits	EAE 4500 Senior Project I 3 CS 4300 Artif. Intelligence 3 Math/ Science elective 4 General Ed/ IR/ Upper Division 3 General Ed 3	EAE 4510 Senior Project II 3 CS Theory or CS elective 3 CS 5460 or CS 5470 4 Phys 2210 Physics I 4 General Ed/ Upper Division 3

Track B: Students who test into EAE 1030 and/ or may not be Calculus-ready in the fall will follow Track B.

	Fall Semester	Spring Semester
Freshman (27 credits)	*EAE 1030 Foundations of CS 3 *Math 1210 Calc I 4 Art 1020 Basic Drawing 3 General Ed 3	EAE 1410 Object Or Prog. 4 Math 1220 Calc II 4 Wrtg 2010 Writing 3 Film 2700 Video Games 3
Sophomore (34 credits)	EAE 2420 Data Str/ Algrthm 4 EAE 3600 3D Modeling 3 CS 2100 Discrete Structures 3 Math 2210 Calculus III 3 FILM 3500 Film Production 4	EAE 3660 Machinima 3 CS 3500 Software Prac. I 4 CS 3810 Comp. Org. 4 FA 3600 Wrtg New Media 3 General Ed/ IR/ Upper Division 3
Junior (33 credits)	CS 3505 Software Prac. II 3 CS 3130 Eng Prob Stats 3 CS Theory or CS elective 3 Math/ Science elective 4 FILM 3610 or 3710 4	CS 4150 Algorithms 3 CS 5530 Database 3 CS Theory or CS elective 3 FILM 3620 or 3720 4 General Ed 3
Senior (33 credits) 127 total credits	EAE 4500 Senior Project I 3 CS 4400 Comp. Systems 4 CS 4300 Artif. Intelligence 3 Phys 2210 Physics I 4 American Institutions (AI) 3	EAE 4510 Senior Project II 3 CS 5460 or CS 5470 4 CS elective 3 General Ed/ DV 3 General Ed/ Upper Division 3

* If you test out of EAE 1030, but are not Calculus-ready, follow Track B. Replace EAE 1030 with another gen ed in your first semester. Replace Calc I with your first level of math.

Choose CS 3100 (fall) *or* CS 3200 (spring) as the required theory course. Students are encouraged to take summer courses to ease the fall & spring semester schedule. See advisor for alternative schedule.

Appendix M: Program of Study Examples for MS

Program of Study for MS

(Application for Admission to Candidacy for the Master's Program
Due at least 2 months preceding semester of graduation)
** Please type information before printing out. **

ENTERED

Today's Date: 10/27/2014

Full legal name: Yadav (Last) Nitin (First) (Middle) UofU ID#: u0886812

Date of Admission 07/05/2013 Handbook Year Used 2013-14

Degree(s) previously received (BS, BA, MS, etc.): B.Tech. Institution: VNIT, Nagpur Year: 2010

Request for admission to candidacy for the degree: Computer Science ☐ Computing ☒ Track Data Management and Analysis

This degree is expected to be completed at the end of: Spring Year: 2015 Thesis Yes Project No Course No

Proposed thesis title: Measuring strength of membership of individual nodes in graph-communities

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): No

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title		Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 1999	Acct-XXXX	Example Course Title		3	A
		-	**Required Courses for Degree**			
University of Utah	Fall 2013	CS 6530	Database Systems		3	A
University of Utah	Spring 2014	CS 6140	Data Mining		3	A
University of Utah	Fall 2014	CS 6150	Advanced Algorithms		3	
University of Utah	Fall 2014	CS 6630	Visualization		3	
		-				
		-				
		-				
		-				

		-	**Additional Courses for Degree Completion**			
University of Utah	Fall 2013	CS 7960	Models of Computation for Massive Data		3	A
University of Utah	Fall 2013	CS 6340	Natural Langauge Processing		3	A
University of Utah	Spring 2014	CS 6950	Independent Study (Thesis work)		4	A
University of Utah	Fall 2014	CS 6970	Masters Thesis Research (Thesis work)		3	
University of Utah	Spring 2015	CS 6970	Masters Thesis Research (Thesis work)		2	
University of Utah	Spring 2015	CS 6955	Clustering		3	
		-				

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Suresh Venkatasubramanian (Chairperson) Signature: [illegible] Date 10/30/14

Name: Signature: Date

Name: Jeff Phillips Signature: [illegible] Date 10.27.14

Name: Vivek Srikumar Signature: [illegible] Date 10.31.2014

Name: Jeff Phillips (Track Director) Signature: [illegible] Date 10.27.14

Program of Study for MS

(Application for Admission to Candidacy for the Master's Program.
Due at least 2 months preceding semester of graduation)
** Please type information before printing out. **

Today's Date: 08/23/2016

Full legal name: ______________________ UofU ID#: ______________
Last First Middle

Date of Admission 03/06/2015 Handbook Year Used 2016-2017

Degree(s) previously received (BS, BA, MS, etc.): BTech Institution: Pondicherry University Year: 2014

Request for admission to candidacy for the degree: Computer Science ☐ [✓ Computing] Track Data management and Analysis

This degree is expected to be completed at the end of: Spring Year: 2017 Thesis ______ Project yes Course ______

Proposed thesis title: ______________________

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): ______________________

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title		Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 1999	Acct-XXXX	Example Course Title		3	A
		-	**Required Courses for Degree**			
U of Utah	Fall 2015	CS - 6150	Advanced Algorithms		3	A
U of Utah	Fall 2015	CS - 6630	Visualization		3	A
U of Utah	Spring 2016	CS - 6140	Data Mining		3	A
U of Utah	Fall 2016	CS - 6530	Databases		3	
		-				
		-				
		-				
		-				

		-	**Additional Courses for Degree Completion**			
U of Utah	Fall 2015	CS - 6350	Machine Learning		3	A-
U of Utah	Spring 2016	CS - 6390	Information Extraction		3	A
U of Utah	Spring 2016	CS - 6190	Probabilistic Modeling		3	A
U of Utah	Fall 2016	CS - 6945	Graduate Internship		1	A
U of Utah	Fall 2016	CS 6340	Natural Language Processing		3	
U of Utah	Fall 2016	CS - 11070	Independent Study		3	
U of Utah	Spring 2017	CS - 11070	Independent Study		3	
		-				

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Feifei Li Signature: [illegible] Date 8/23/16
Chairperson

Name: Ellen Riloff Signature: [illegible] Date 8/23/16

Name: Jeff M. Phillips Signature: [illegible] Date 8.23.16

Name: ______________ Signature: ______________ Date ______

Name: Jeff M. Phillips Signature: [illegible] Date 8.23.16
Track Director

Application for Admission to Candidacy for the Master's Program
(Due at least 2 months preceding semester of graduation)

Date: 10/09/2013

Date of Admission 08/20/2012 Handbook Year Used 2012

Full legal name: BASKAR (Last) ARUN (First) (Middle) UofU ID#: 00811338

Degree(s) previously received (BS, MS, etc.): BE Institution: CEG, CHENNAI, INDIA Year: 2010

Request for admission to candidacy for the degree: MS in Computing Track: DATA MANAGEMENT & ANALYSIS

This degree is expected to be completed at the end of: December Year: 2013 Thesis ___ Project ___ Course ✓

Proposed thesis title: ___

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): ___

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title	Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 1999	Acct-XXXX	Example Course Title	3	A
			Required Courses for Degree		
U of Utah	Fall 2012	CS 6150	Advanced Algorithms	3	A
U of Utah	Fall 2012	CS 6630	Scientific Visualization	3	A
U of Utah	Spring 2013	CS 6955	Data Mining	3	A
U of Utah	Fall 2013	CS 6530	Database Systems	3	
			ENTERED		
			Additional Courses for Degree Completion		
U of Utah	Fall 2012	CS 6340	Natural Language Processing	3	A
U of Utah	Spring 2013	CS 6230	High Performance Computing & Parallelization	3	A
U of Utah	Spring 2013	CS 5470	Compilers	4	A
U of Utah	Fall 2013	CS 6350	Machine Learning	3	
U of Utah	Fall 2013	CS 6810	Computer Architecture	3	
U of Utah	Fall 2013	BMI 6470-001	Biomed Info Retrieval	2	

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: FEIFEI LI Signature: [illegible] Date 10/7/13
Chairperson
Name: ELLEN RILOFF Signature: [illegible] Date 10/7/13
Name: JEFF PHILLIPS Signature: [illegible] Date 10.07.13
Name: FEIFEI LI Signature: [illegible] Date 10/7/13
Track Director

Program of Study for MS

(Application for Admission to Candidacy for the Master's Program.
Due at least 2 months preceding semester of graduation)
** Please type information before printing out. **

Today's Date: 30-Aug-2016

Full legal name: ______________________ UofU ID#: ____________
Last First Middle

Date of Admission 24-Aug-2015 Handbook Year Used 2015-2016

Degree(s) previously received (BS, BA, MS, etc.): B.E Institution: SRIT Year: 2010

Request for admission to candidacy for the degree: Computer Science ■ Computing ☐ Track ____________

This degree is expected to be completed at the end of: August Year: 2017 Thesis <.> Project _____ Course _____

Proposed thesis title: Fast NVMe Layer for a Decomposed Kernel

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): ____________

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title		Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 1999	Acct-XXXX	Example Course Title		3	A
		-	**Required Courses for Degree**			
		CS 6150	Advanced Algorithms		3	
		CS 6460	Operating Systems		4	A
		CS 6810	Computer Architecture		3	A-
		-				
		-				
		-				
		-				
		-				

		-	**Additional Courses for Degree Completion**			
		CS 6963	Distributed Systems		3	A-
		CS 6964	Computer Security Research		3	B+
		CS 6950	Independent Study		6	
		CS 6970	Research Credits		8	
		-				

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Anton Burtsev Signature: [illegible] Date 08/30/16
Chairperson
Name: Ryan Stutsman Signature: [illegible] Date 9/1/16
Name: Robert Ricci Signature: [illegible] Date 9/1/16
Name: Jacobus Van der Merwe Signature: [illegible] Date 9/1/2016
Name: Jacobus Van der Merwe Signature: [illegible] Date 9/1/2016
Track Director

Program of Study for MS

(Application for Admission to Candidacy for the Master's Program.
Due at least 2 months preceding semester of graduation)
** Please type information before printing out. **

ENTERED

Today's Date: 05/20/2016

Full legal name: ______ UofU ID#: ______
Last First Middle

Date of Admission 08/24/2014 Handbook Year Used 2014-15

Degree(s) previously received (BS, BA, MS, etc.): B.Tech Institution: JNTU, Hyderabad India Year: 2011

Request for admission to candidacy for the degree: Computer Science ■ Computing ☐ Track General CS

This degree is expected to be completed at the end of: August Year: 2016 Thesis ____ Project X Course ____

Proposed thesis title: Building Resiliency within the Uintah Computation Framework.

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): ______

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title		Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 1999	Acct-XXXX	Example Course Title		3	A
		-	**Required Courses for Degree**			
University of Utah	Fall 2014	CS 6150	Advanced Algorithms		3	B
University of Utah	Spring 2015	CS 6810	Computer Architecture		3	B+
University of Utah	Spring 2015	cs 6460	Operating Systems		4	B+
		-				
		-				
		-				
		-				
		-				

		-	**Additional Courses for Degree Completion**			
University of Utah	Fall 2014	CS 6210	Advanced Scientific Computing I		3	A
University of Utah	Fall 2015	CS 6340	Natural Language Processing		3	B+
University of Utah	Fall 2015	CS 6963	Distributed Systems		3	B+
University of Utah	Spring 2016	CS 6140	Data Mining		3	A
University of Utah	Spring 2016	CS 6235	Programming for Many Core		3	A
University of Utah	Fall 2015	CS 6950	Independent Study		3	A
University of Utah	Spring 2016	CS 6950	Independent Study		3	A
		-				

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Martin Berzins Signature: [illegible] Date 5-21-16
Chairperson

Name: Hari Sundar Signature: [illegible] Date 5-20-16

Name: Chris Johnson Signature: [illegible] Date 5-20-16

Name: ______ Signature: ______ Date ______

Name: Kobus Van Der Merwe Signature: [illegible] Date 5/20/2016
Track Director

Program of Study for MS

(Application for Admission to Candidacy for the Master's Program.
Due at least 2 months preceding semester of graduation)
** Please type information before printing out. **

ENTERED

Today's Date: July 11, 2016

Full legal name: ______________________ UofU ID#: ______________
Last First Middle

Date of Admission Fall 2015 Handbook Year Used 2015-2016

Degree(s) previously received (BS, BA, MS, etc.): BE Institution: University of Science and Technology of China Year: 2015

Request for admission to candidacy for the degree: Computer Science ■ Computing ☐ Track ______________

This degree is expected to be completed at the end of: Fall Year: 2016 Thesis ____ Project ____ Course Y

Proposed thesis title: ______________________________

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): ______________

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title		Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 1999	Acct-XXXX	Example Course Title		3	A
		-	**Required Courses for Degree**			
University of Utah	Fall 2015	CS -6150	Advanced Algorithms		3	A-
University of Utah	Spring 2016	CS -6460	Operating Systems		4	A
University of Utah	Spring 2016	CS -6810	Computer Architecture		3	A-
		-				
		-				
		-				
		-				
		-				

		-	**Additional Courses for Degree Completion**			
University of Utah	Fall 2015	CS -6350	Machine Learning		3	A
University of Utah	Fall 2015	CS -6600	Mathematics of Computer Graphics		3	A
University of Utah	Spring 2016	CS -6110	Rigorous System Design		3	A-
University of Utah	Spring 2016	CS -6300	Artificial Intelligence		3	A
University of Utah	Spring 2016	CS -6510	Functional Programming		3	A
University of Utah	Fall 2016	CS -6530	Database Systems		3	
University of Utah	Fall 2016	CS -6640	Image Processing		3	
		-				

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Matthew Flatt Signature: [illegible] Date 7/11/16
Chairperson
Name: Suresh Venkatasubramanian Signature: [illegible] Date 7/11/16
Name: ZVONIMIR RAKAMARIC Signature: [illegible] Date 7/11/16
Name: ______________ Signature: ______________ Date ______
Name: ______________ Signature: ______________ Date ______
Track Director

Appendix N: Program of Study Examples for PhD

Program of Study & Full Committee for PhD
School of Computing

ENTERED

please type information before printing

Date: May 5, 2015

Full legal name: ______________________ UofU ID#: ________
Last First Middle

Degree: Computer Science ☐ Computing ■ Track Image Analysis

Date of Admission Fall 2012 Handbook Year used 2012

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): N/A

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title	Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 2010	CS - XXXX	Example Course Title	3	A
			Required Courses for Degree		
UTAH	Fall 2012	CS 6640	Image Processing	3	A
UTAH	Spring 2012	CS 7640	Adv Image Processing	3	A-
UTAH	Spring 2015	CS 6320	3D Computer Vision	3	A
UTAH	Fall 2013	CS 6350	Machine Learning	3	A-

			Additional Courses for Degree Completion		
UTAH	Fall 2012	CS 6630	Scientific Visualization	3	A
UTAH	Spring 2013	CS 6957	Probabilistic Modeling	3	A
UTAH	Fall 2014	CS 6665	Character Animation	3	A
UTAH	Spring 2014	CS 6140	Data Mining	3	A
UTAH	Spring 2014	CS 6959	Computational Inverse Problems	3	A
UTAH	Fall 2014	MATH 6880	Optimization	3	A
UTAH	Fall 2012	CS 7970	PhD Dissertation Research	3	A
UTAH	Spring 2013	CS 7970	PhD Dissertation Research	3	A
UTAH	Fall 2013	CS 7970	PhD Dissertation Research	3	A
UTAH	Spring 2014	CS 7970	PhD Dissertation Research	3	A
UTAH	Fall 2014	CS 7970	PhD Dissertation Research	3	A
UTAH	Spring 2015	CS 7970	PhD Dissertation Research	3	

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Ross Whitaker Signature: [illegible] Date 5/6/15
Chairperson

Name: Jeff Phillips Signature: [illegible] Date 5.7.15

Name: Tom Fletcher Signature: [illegible] Date 5/7/2015

Name: Sarang Joshi Signature: [illegible] Date 5/12/2015

Name: Braxton Osting Signature: [illegible] Date 5/11/2015

Name: Guido Gerig Signature: [illegible] Date 5/7/2015
Track Director

ENTERED

Program of Study & Full Committee for PhD
School of Computing

please type information before printing

Date: July 11, 2015

Full legal name: ______ Last First Middle UofU ID#: ______

Degree: Computer Science ■ Computing ☐ Track* ______

Date of Admission August 20, 2012 Handbook Year used 2014-2015

Human Subjects Committee Clearance Required? (if Yes, attach a copy of approval form): No

If work from another university is to be included in the course work listed below, please check with Admissions to verify that official transcripts have been evaluated and recorded on the University of Utah record.
List chronologically work required by the Committee for the proposed degree being sure to include thesis hours in the quarter/semester taken. Graduate work that might be counted toward a doctorate but that is not required for the Master's degree should NOT be listed.

Institution	When Registered	Department and Course No.	Course Title	Qtr/Sem Hours	Grade
U of XXXXXXX	Sem 2010	CS - XXXX	Example Course Title	3	A
			Required Courses for Degree		
U of Utah	Fall 2012	CS - 6150	Advanced Algorithms	3	A-
U of Utah	Fall 2012	CS - 6810	Computer Architecture	3	A
U of Utah	Spring 2013	CS - 6460	Operating Systems	4	A

			Additional Courses for Degree Completion		
U of Utah	Fall 2012	CS - 7970	PhD Dissertation Research	4	A
U of Utah	Spring 2013	CS - 7520	Program Lang Semantics	3	A
U of Utah	Spring 2013	CS - 7970	PhD Dissertation Research	3	A
U of Utah	Fall 2013	CS - 6475	Advanced Compilers	3	A
U of Utah	Fall 2013	CS - 7970	PhD Dissertation Research	8	A
U of Utah	Spring 2014	CS - 6140	Data Mining	3	A
U of Utah	Spring 2014	CS - 6510	Functional Programming	3	A
U of Utah	Spring 2014	CS - 7970	PhD Dissertation Research	5	A
U of Utah	Fall 2014	CS - 7970	PhD Dissertation Research	9	A
U of Utah	Spring 2015	CS - 6300	Artificial Intelligence	3	A
U of Utah	Spring 2015	CS - 6350	Machine Learning	3	A
U of Utah	Spring 2015	CS - 7970	PhD Dissertation Research	3	A

The program of study as outlined has been approved by the applicant's supervisory committee listed below:

Name: Matthew Flatt Signature: [illegible] Date 7/20/15
Chairperson

Name: Eric Eide Signature: Eric Eide Date 7/17/15

Name: Sean McDirmid Signature: [illegible] Date 7/13/15

Name: Matthew Might Signature: [illegible] Date 7/13/15

Name: John Regehr Signature: [illegible] Date 7/17/15

Name: J. F. Van der Werwe Signature: [illegible] Date 8/6/2015
Track Director

Appendix O: Recent Qualifying Exams

Qualifier Examination

Sriram Aananthakrishnan

[Q1] MPI-aware Compiler Optimizations for Improving Computation-Communication Overlap - Anthony Danalis, Lori Pollock, Martin Swany, John Cavazos

MPI is the dominant parallel programming paradigm for HPC applications. MPI communication operations can be viewed as memory access operations incurring very high latency. To amortize this cost these operations are overlapped with computations. Compilers typically perform various optimizing transformations that hides processor memory latency. However, opportunities to hide communication latency is lost as compilers treat MPI calls as black box functions. The goal of this paper is to hide the communication latency by increasing the overlap between the computation and the communication.

Consider the code shown in the table. The code on right side is the result of applying loop fission to the code on left side. To perform such a transformation, compilers should be aware of the side-effects or dependencies of MPI operations.

`for j = 1, 10 {`
`  A[j] = j`
`  MPI_Isend(A[j],.. r[j])`
`  MPI_Wait(r[j]..)`
`  B[j] = 2*j`
`}`
	`for j = 1, 10 {`
	`  A[j] = j`
	`  MPI_Isend(A[j],.. r[j])`
	`}`
	`for j = 1, 10`
	`  B[j] = 2*j`
	`for j = 1, 10`
	`  MPI_Wait(r[j]..)`

Safety Analysis: For safety, the dataflow analysis should interpret MPI operations for their side-effects and establish that the dependencies between those statements are not violated. This paper associates side-effecting semantics for MPI operations in two layers.

(i) **Application-layer:** Application level side-effects of MPI operations are captured using two sets `DEF:` set of variables defined , `USE:` set of variables referenced by the operation. The table below shows the semantics of MPI_Irecv and MPI_Wait.

`MPI_Irecv(buf, count, dtype, src, tag, com, req)`	`DEF={buf, req}`
	`USE={count, dtype, src, tag, com}`

`MPI_Wait(req, stat)`	`DEF={req, stat}, USE={req}`

```
S1: MPI_IRecv(b,..r1)
S2: MPI_Wait(r1)
S3: a[i] = b[i]
```

(ii) **Library-layer:** Library level side-effects are described as array-to-array copying. To motivate the need for this, consider the code shown on the left. $S2$ and $S3$ cannot be interchanged as the buffer cannot be read until wait is completed. Application level semantics (def-use) are not sufficient to capture this as there is no dependence between $S2$ and $S3$. Such a transformation is illegal and therefore array-to-array copy semantics are introduced to model these effects. The semantics for MPI_Irecv and MPI_wait is shown in the table below. The array variables `inMesg`, `artificialVar`, `whichbuf` do not exist in real code nor do the array-to-array copying assignments. They simply model the effect of MPI_Irecv operation. For Irecv, the first assignment to `buf[0:N-1]` copies N

MPI_Irecv	`buf[0:N-1]=inMesg[src][tag][comm][0:N-1]+artificialVar`
	`whichbuf[req]=buf`
	`inMesg[src][tag][comm][0:N-1]=artificialVar`

MPI_wait	`whichbuf[req][0:N-1]-=artificialVal`

elements from incoming message denoted by `inMesg` identified by `src, tag, comm`. `N` here is determined by the send operation as count is not known on the receiver side. The `artificialVar` is added to the `buf` to restrict it from being read before wait. The assignment `whichbuf[req]=buf` associates `req` and `buf` which is later used in the semantics of MPI_Wait that modifies `buf` by subtracting `artificialVar` enabling `buf` to be read after wait. The third assignment to `inMesg` models the fact that the messages are read only once. Now, if we substitute these semantics for statements $S1, S2$, an artificial true dependence is introduced between $S2$ and $S3$ due to the write to `buf` by MPI_Wait which rejects the illegal transformation of swapping them. The paper also discusses rules for control-flow code motion and segmentation of MPI operations ie call to a single MPI_Send is broken into M operations where MPI_Send = $M\times$ MPI_Send. The MPI_Recv is also

replaced by M receives with appropriate variable renaming of buffers and request handles. Note that for segmentation, the matchings (send-receive pair) should be known as before.
Optimizations: The optimization algorithm takes as input (i) function of the MPI program (ii) set of data transfers (matched send-receive pair) (iii) summarized inter-procedural analysis information. They do not solve the message matching problem and they rely on external tool to provide this information. For each send-receive pair they apply the following sequence of transformations (i) blocking to non-blocking (ii) replace MPI calls with library calls optimized for communication (iii) insert dataflow independent computation inside the communication window to increase overlap (iv) variable cloning to eliminate dependencies (v) loop fission to split loops into one that depends on communication and the other that is independent (vi) communication and computation tiling and pipelining (CCTP) - combination of loop tiling, communication call segmentation, loop fusion, loop alignment, loop peeling transformations. The legality of these transformations are determined by the safety analysis that interprets MPI operations for its side-effects as discussed above. The transformations are chosen such that they do not reverse the results of the other. The sequence of transformations are repeated until no more overlapping is possible. They have demonstrated the efficiency of these optimizations by manually performing these transformations and show that the execution time of real kernels can be decreased by 30%.

Compare the communication analysis you are developing with that described in the paper in support of the optimizations

The analyses described in our work [1] can be classified as non-separable dataflow analyses ie data flow across MPI communication is critical to the precision and correctness of the analysis. Reaching constants over MPI operations is an example of a non-separable dataflow analysis. Program slicing is another example that requires the knowledge of MPI matchings to reproduce a behavior involving communication. The optimizations described in this paper [2] do not require non-separable dataflow analyses. The critical information needed for these transformations are side-effects of MPI operations with respect to computation so as to facilitate code motion to increase overlap window between computation and communication. However, their work requires the knowledge of MPI communication or matchings which they do not solve and rely on another tool to provide this information. For example, the library layer side-effecting semantics for MPI_Irecv is described as an array copy of `N` elements where `N` is determined by the corresponding send operation. In our work, we provide a dataflow analysis to determine MPI communication and the results should enable their analyses and transformations (MPI call segmentation which transforms send and its correponding receive operation), although there are other ways to obtain these results (annotations, profiling).

Will your analysis support the set of proposed optimizations

Lets briefly review each transformation and the safety check required for its legality.
(i) Preliminary Transformations: The preliminary transformations proposed in this paper are loop unrolling, constant propagation and folding, dead code elimination of redundant code introduced by loop unrolling and redundant store elimination. These transformations are based on standard semantics and do not require any analysis to interpret MPI semantics.
(ii) Communication Library Specific Transformation: In their work [3], they transform specific MPI operations to Gravel library optimized for the underlying interconnect. The transformation assumes send-receive matching in the form of pragma and our analysis can automatically enable this transformation without the pragma directives.
(iii) Overlap Window Expansion: Each send-receive pair has two overlap windows on the sender side and the receiver side. The goal of this transformation is to increase the temporal length of this overlap window. It achieves this by moving computation that has dataflow such that it can be brought in safely inside this window. The safety analysis performed is based on the side-effecting semantics described earlier to compute the dependency information and the transformation is performed such that the resulting code does not violate the dependencies due to the array-to-array-copy semantics introduced by safety analysis. However, the array-to-array copy semantics require message matchings which can be provided by our work.
(iv) Variable Cloning: Variable cloning can remove data dependencies and enable further transformations. The transformations they do are scalar renaming, array expansion and array renaming. These transforma-

tion are safe and do not require any analysis.
(v) Loop Fission: To expand the overlap window further, this transformation splits a computation loop into two loops such that one dependes on the communication and other independent. The analysis needed is the safety analysis with side-effecting semantics described above.
(vi) CCTP: Communication and Computation Tiling and Pipelining is a sequence of the following transformations – loop tiling, MPI call segmentation, loop fusion, loop alignment and loop peeling. The main goal here is to tile the outer loop resulting in message segmentation and overlap the segmented messages with tiled computations. The code on the left is before the CCTP transformation and the code on the right is after CCTP transformation. The correctness of this transformation depends on the correctness of loop fusion which requires the safety analysis with side-effecting semantics. Loop tiling is applied on single or the outermost loop and therefore its safe. MPI call segmentation is safe as long as the variable renamings and vectorization of resources (MPI_request) are correct. Loop alignment and peeling are trivial transformations where loop alignment delays MPI_Wait by one iteration and loop peeling accounts for MPI_Wait in the final iteration.

```
MPI_Irecv(rB[1], N,..rReq)
for i = 1, N
  sB[i] =
MPI_Isend(sB[1], N,.. sReq)
MPI_Wait(sReq)
MPI_Wait(rReq)
```

```
for T=1, N with T=T+K {
  MPI_Irecv(rB[T], K,.., rReq[T])
  for i=T, T+K-1
    sB[i] =
  MPI_Isend(sB[T], K,..sReq[T])
  if(T>1) {
    MPI_Wait(sReq[T-K])
    MPI_Wait(rReq[T-K])
 }
}
MPI_Wait(sReq[N-K])
MPI_Wait(rReq[N-K])
```

To summarize, the transformations proposed in the paper do not require non-separable dataflow analysis which is the primary focus of our work. This paper interprets MPI operations for its side-effects and in our work we interpret MPI operations concretely. Our work therefore lacks the ability to see the side-effects and cannot be used directly to check the legality of these transformations. However, our work can be extended to account for side-effects. Another observation is that this paper applies the optimizing transformations for all MPI operations which may or may not improve performance. With the help of non-separable dataflow analysis that flows the message length across MPI operations, this framework can pick the operations to apply the corresponding transformations.

References

[1] Sriram Aananthakrishnan, Greg Bronevetsky, Ganesh Gopalakrishnan, *Hybrid Approach for Data-flow Analysis of MPI Programs*, ICS 2013 [Poster Paper]

[2] Anthony Danalis, Lori Pollock, Martin Swany, John Cavazos, *MPI-aware Compiler Optimizations for Improving Computation-Communication Overlap*, ICS 2012

[3] Anthony Danalis, Lori Pollock, Martin Swany, John Cavazos, *Implementing an Open64-based Tool for Improving the Performance of MPI Programs*, Open64 Workshop, CGO 2008.

[Q2] Direct Product vs Sequential Composition of Analyses:

To analyze real-world programs, composing program analyses is of paramount importance. There are two popular approaches to combine program analyses.

- **Sequential:** The analyses are run one after the other in a sequence. This is a straight-forward way to compose program analyses. The challenge here lies in communicating the results.
- **Product:** This method of combining analyses was first introduced by Cousot et al [1]. It is based on the theory of abstract interpretation. The analyses are weaved together and executed in tandem by computing (i) direct product of abstract domain and their operators (ii) direct product of their transfer functions.

The two approaches have their advantages and disadvantages. Let's compare the two approaches from the following perspectives (i) Precision (ii) Complexity (iii) Implementation (iv) Ordering.

Precision:

The key to leverage precision of co-operating analyses lies in the efficient communication of their results. For example, to perform constant propagation of pointer dereferences the constant propagation should be aware of the points-to information.

Product Composition: The product combination of analyses has a clear advantage over sequential combination when improving the precision results. The following diagram from [2] best illustrates the idea.

$$\begin{tikzcd}
\hat{\varsigma} \arrow[r] & \hat{\varsigma}' \arrow[r] & \hat{\varsigma}'' \arrow[r] & \cdots \\
\Pi \arrow[r, dashed] & \Pi' \arrow[r, dashed] & \Pi'' \arrow[r, dashed] & \cdots
\end{tikzcd}$$

Figure 1: Sequential Composition

$$\begin{tikzcd}
\hat{\varsigma} \arrow[r] \arrow[dr, dashed] & \hat{\varsigma}' \arrow[r] \arrow[dr, dashed] & \hat{\varsigma}'' \arrow[r] \arrow[dr, dashed] & \cdots \\
\Pi \arrow[r, dashed] \arrow[ur] & \Pi' \arrow[r, dashed] \arrow[ur] & \Pi'' \arrow[r, dashed] \arrow[ur] & \cdots
\end{tikzcd}$$

Figure 2: Product Composition

$\hat{\varsigma}$ is the abstraction domain of one abstract interpreter and Π is the other. As illustrated in Fig. 2, product composition at every step of the transition communicates information between them leading to improved precision. At every step of the transition, each abstract interpreter refines its domain based on the information from the other. This is called domain refinement and the quality of the precision depends on the product operator employed. [3] surveys different product operators employed in abstract interpretation. Lets briefly look at various product composition techniques in abstract interpretation and their domain refinement capabilities.

- **Cartesian Product:** Let $\mathcal{A} = (A, \sqcup, \top, \bot), \mathcal{B} = (B, \sqcup, \top, \bot)$ be two abstract domains. The cartesian product of $\mathcal{A} \times \mathcal{B}$ is given by direct product of the two domains where the domain operators (meet, join) and the semantic operators (abstract semantics for programming language constructs in the respective domain) are applied component wise. It does not employ any domain refinement operator. For example, consider interval and parity domains $(2..4, O), (2..3, O), (3..4, O), (3..4, O)$ represent the same information $(3, O)$. As pointed out by Cousot [1], the analyses do not learn much from each other in this style of product composition.
- **Reduced Product:** The reduced product of the two domains $\mathcal{A}, \mathcal{B}$ is given by the cartesian product of the two domains $\mathcal{A} \times \mathcal{B}$ and an additional reduction operator $\rho : \mathcal{A} \times \mathcal{B} \to \mathcal{A} \times \mathcal{B}$. The domain

operators and the semantic operators are applied component wise as before. After, each transition the reduction operator refines the respective domains. The reduction operator is applied in iterative manner (iterated reduced product) to compute minimal information. Reduced product is the popular product composition method employed when combining two abstract interpreters. For example Logic Flow Analysis (LFA) [2] combines two abstract interpreters (i) generic flow analysis on some arbitrary domain ς (ii) theorem prover Π : set of predicates with implication relation as $\sqsubseteq$. The syntactic rules of the theorem prover Π describes the transitions in this domain. The semantic inference rules for the theorem prover takes into account the abstract state of flow analysis $\hat{\varsigma}$ and describes the reduction mechanisms to refine the set of predicates in this domain. The reduction operator in the other direction (theorem prover to flow analysis) depends on the particular domain ς that a flow analysis operates on such as parity or interval.

- **Reduced Cardinal Power:** The reduced cardinal power allows to track disjunctive information such as when x is odd y is in interval [0..10]. The precision of this is better than the reduced product but expensive to compute.
- **Logical Product:** Gulwani et al [4] proposed the construction of logical product (special case of reduced product) where they restrict their domains to theories that are convex, disjoint and stably infinite. The reduction operator defined for this composition is based on Nelson-Oppen procedure [7] used to combine decision procedures in SMT solvers. The precision of this is better than the reduced product.

Sequential Composition: The challenge to improve precision for this composition lies in communicating the results. The straight-forward way to do this is code-rewriting. [6] uses code-rewriting in their sequential and tight composition (product style) of dataflow analyses. For example, a pointer analysis replaces all pointer dereferences with the variables pointed by them. However, code-rewriting communicates only must information and analyses cannot benefit from may information thereby limiting the precision. Fuse [2] permits both sequential and product composition of analyses. In Fuse when composing the analyses sequentially, the results of an analysis is represented as an Abstract Transition System (ATS) (abstract state transition graph computed by dataflow analysis) and a query interface allows a second analysis to access information computed by previous analyses. The query interface returns information in terms of abstract objects which may denote sets of memory locations, sets of values and sets of control locations. The precision is determined by these sets and the operations on them.

Complexity:

The complexity involved in combining the two analyses for product and sequential composition is discussed below.
Product Composition: The complexity of running cartesian product of two abstract interpreters is the sum of semantic operators in the respective domains. For reduced product, it is sum of the semantic operators and in addition the reduction operator. Note that the reduction operator does not always compute minimal information and has to be applied iteratively (iterated reduced product) to compute a fix point and the complexity increases due to the reduction operator. For example consider interval and parity domain, $([1..1], E)$ can be reduced to $(\bot_I, E)$ which can be further reduced to $(\bot_I, \bot_P)$ For reduced cardinal power, every time lattice or semantic operator is applied it has to update all the disjunctive information and therefore expensive. In Logical product the complexity grows quadratically on the input size (number of theories).
Sequential Composition: The complexity for this style of composition is the sum of the complexities of the individual analyses. Considering the complexity involved in communicating the results (i) code-rewriting is linear in the size of the program (ii) in Fuse, every transfer function invokes the query interface which is also linear. Sequential composition is slightly better than reduced product with respect to complexity.

Implementation:

We will compare the implementation cost and issues in the two styles of composition.
Product Composition: Implementing the reduction operator is highly specific for the domains we are refining and therefore composing two arbitrary abstractions involves design and implementation cost. This requires tight co-ordination between different groups working on different abstractions. To add another

abstract interpreter to this composition would require defining the reduction operator pair-wise between all existing domains which grows quadratically with the number of abstract interpreters in the composition. As pointed out by [3] widening is not free in reduced product (cannot be applied component wise) which is needed for termination when operating on lattices of infinite height. In Logical product the operators come for free (automatic way to combine the operators is described) but its restrictive in communicating only must equalities between abstract interpreters (convex theories only define must equal operator). The implementation costs of product composition can be ameliorated by executing the two interpreters in parallel.
Sequential Composition: This style of composition presents no additional implementation costs however code-rewriting warrants the presence of transformation framework. Fuse permits composition of arbitrary abstractions however there is an implementation cost involved in defining abstract objects based on analysis information for the query interface. It is difficult to compare the effort involved to implement the abstract objects against implementing the reduction operators.

Ordering:

Composing analyses in a sequence fixes an order in which the analyses are applied. The order affects the precision of the composition. Currently there is no automatic way of determining the order and it is user driven. Automating this would require demand-driven, incremental analysis capabilities with a fix-point algorithm which adds more complexity and implementation costs. Product composition presents no such issues however it is difficult to add new abstract interpreter to this composition as discussed before.

References

[1] *Systematic Design of Program Analysis Frameworks*, Patrick Cousot, Radhia Cousot, POPL 1979

[2] *Logic-Flow Analysis of Higher-Order Programs*, Matt Might, POPL 2007.

[3] *A Survey on Product Operators in Abstract Interpretation*, Agostino Cortesi, Giulia Costantini, Pietro Ferrara, Festschrift for Dave Schmidt 2013

[4] *Combining Abstract Interpreters*, Sumit Gulwani, Ashish Tiwari, PLDI 2006

[5] *Combining Program Analysis via Abstract Transition Systems*, Greg Bronevetsky, Michael G Burke, Jisheng Zhao, Sriram Aananthakrishnan, Vivek Sarkar, LLNL-TR 2013

[6] *Composing Dataflow Analyses and Transformations*, Sorin Lerner, David Grove, Craig Chambers, POPL 2002

[7] *Simplification by Cooperating Decision Procedures*, Greg Nelson, Derek Oppen, TOPLAS 1979.

[Q3] Monotone Dataflow Analysis Framework

Lattice: A Lattice is given by $\mathcal{L} = (L, \sqsubseteq, \sqcup, \sqcap, \top, \bot)$ where L is a set, $\sqsubseteq$ is a partial order relation on L, $\sqcup$ is the least upper bound operator, $\sqcap$ is the greatest lower bound operator, $\top$ the greatest element and $\bot$ the lowest element. In a lattice $\mathcal{L}$, for any pair of elements $x, y \in L$ there exist a least upper bound written $x \sqcup y$ and a greatest lower bound written $x \sqcap y$. For dataflow analysis, we will typically work on cartesian product of finite lattices (lattice of finite height) $L_1 \times L_2 \times ..L_n = (x_1, x_2, ..x_n)|x_i \in L_i$. Note that the cartesian product is also a lattice where the $\sqsubseteq, \sqcup, \sqcap$ are defined point-wise.

CFG: A control-flow graph is a directed graph $G = (V, E)$ where V is set of program locations and E set of edges that represent possible flow of control.

Monotonicity: A function $f : \mathcal{L} \to \mathcal{L}$ is monotone if $\forall x, y \in \mathcal{L} : x \sqsubseteq y \implies f(x) \sqsubseteq f(y)$.

Theorem: 1. *In a Lattice $\mathcal{L}$ with finite height, every monotone function has a unique least fixed point*

Fixed-point for System of Equations: Let $\mathcal{L}$ be a lattice of finite height. Consider the system of equations,

$$\begin{aligned} x_1 &= F_1(x_1, x_2, ..x_n) \\ x_2 &= F_2(x_1, x_2, ..x_n) \\ &\vdots \\ x_n &= F_n(x_1, x_2, ..x_n) \end{aligned} \tag{1}$$

where $x_1, x_2, ..x_n \in \mathcal{L}$ and $F_i : \mathcal{L}^n \to \mathcal{L}$ is a collection of monotone functions. Every such system has a unique least fixed point solution due to the monotonicity of F_i (Theorem 1) . Let $F : \mathcal{L}^n \to \mathcal{L}^n$ be the function that computes the least fixed-point solution of the system and it is given by

$$F(x_1, x_2, ..x_n) = (F_1(x_1, x_2, ..x_n), F_2(x_1, x_2, ..x_n), \dots, F_n(x_1, x_2, ..x_n)) \tag{2}$$

Monotone Dataflow Analysis Framework Dataflow analysis starts with a lattice $\mathcal{L}$ of finite height and a CFG. For every node c in CFG, we will use the notation $[\![c]\!]$ to denote the dataflow value at c where $[\![c]\!] \in \mathcal{L}$. The dataflow constraint or abstract semantics is defined for each construct of the programming language. If the CFG has nodes $V = \{c_1, c_2, ..c_n\}$, we work with the lattice $\mathcal{L}^n$ ie we associate a lattice value for each $c_i \in V$. The abstract semantics enables us to compute the dataflow value at each c_i given by $[\![c_i]\!] = F_i(c_1, c_2, ..c_n)$ – our transfer functions. For each c_i in CFG, $[\![c_i]\!]$ generates F_i giving rise to system of equations similar to equation 1.

$$\begin{aligned} [\![c_1]\!] &= F_1([\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!]) \\ [\![c_2]\!] &= F_2[\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!]) \\ &\vdots \\ [\![c_n]\!] &= F_n([\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!]) \end{aligned} \tag{3}$$

As before we solve equation 3 using a procedure $F : \mathcal{L}^n \to \mathcal{L}^n$ – our worklist algorithm.

$$F([\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!]) = (F_1([\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!]), F_2([\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!]), \dots, F_n([\![c_1]\!], [\![c_2]\!], ..[\![c_n]\!])) \tag{4}$$

Dataflow analysis captures the program behavior on some abstract domain ($\mathcal{L}$). The dataflow analysis is sound (any property that was verified to be hold by the analysis on abstract domain must hold for all concrete executions) provided the abstract semantics are correct. The analysis is conservative as the solutions are over-approximate and therefore doesn't guarantee completeness (some program behaviors as computed by the analysis may not be possible – false-positives).

Algorithm 1 Naive

```
1: $x^n = (\bot, \bot, ..\bot)$
2: **repeat**
3:     $t^n = x^n$;
4:     $x^n = F(x^n)$;
5: **until** $x^n \neq t^n$
```

Fixed-point Algorithm for Dataflow Equations: A naive fixed point algorithm for the dataflow equations (equation 3) is show in Algorithm 1. Clearly the algorithm is very inefficient as the information for all nodes is recomputed in every iteration. Most often $[\![c_i]\!]$ depends only on few $[\![c_j]\!]$ such as its successors. Lets define $succ(c_i) = \{c_j | [\![c_j]\!] \text{ depends on } [\![c_i]\!]\}$. The worklist algorithm is shown in Algorithm 2.

Algorithm 2 Worklist

```
1: $[\![c_1]\!] = \bot \ldots [\![c_n]\!] = \bot$;
2: $q = [c_1 \ldots c_n]$;
3: **while** $q \neq []$ **do**
4:     assume $q = [c_i \ldots]$;
5:     $[\![y]\!] = F_i([\![c_1]\!] \ldots [\![c_n]\!])$;
6:     $q = q.tail()$;
7:     **if** $[\![y]\!] \neq [\![c_i]\!]$ **then**
8:         **for** $c \in succ(c_i)$ **do**
9:             $q.append(c)$;
10:        **end for**
11:        $[\![c_i]\!] = [\![y]\!]$
12:    **end if**
13: **end while**
```

Constant Propagation Analysis: The constant propagation lattice $\mathcal{C}$ is shown in Fig 3.

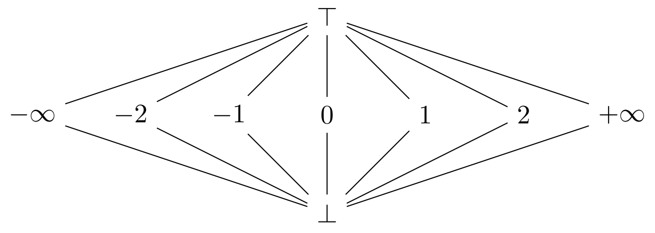


Figure 3: Constant Propagation Lattice

Abstract Semantics (Dataflow Constraints) for Constant Propagation: The full lattice for this analysis is the map lattice $Vars \mapsto \mathcal{C}$ where $Vars$: set of variables and $\mathcal{C}$: Constant propagation lattice. We have

$$\mathcal{CP} : Vars \mapsto \mathcal{C} = \{[id_1 \mapsto c_1, id_2 \mapsto c_2 \ldots, id_n \mapsto c_n] | c_i \in \mathcal{C}\}$$

The operator $\sqcup$ is applied pointwise where we combine the values associated with respective variables and it merges the information from different paths. We will use the notation $\mathcal{A}[op]$ to denote dataflow constraint or abstract semantics for each construct of the programming language.
For variable declaration C_{decl}, Let

$$\mathcal{A}[C_{decl}] = \lambda(e).e[id_1 \mapsto \bot, id_2 \mapsto \bot \ldots id_n \mapsto \bot] \ JOIN(C_{decl})$$

where e is the map or the environment that maps each variable id to a value $c \in \mathcal{C}$. Here it is bound to $JOIN(C_{decl})$. The operation $e[id_1 \mapsto ? \ldots]$ is used to denote update to the map e. The semantics of C_{decl} is to set the variables $id_1, id_2 \ldots id_n$ it declares to $\perp$.

For assignment $C_=$, we have

$$\begin{aligned}
\mathcal{A}[C_=] &= \lambda(e).e[id \mapsto eval(e, expr)]\ JOIN(C_=) \\
eval(\sigma, id) &= \sigma(id) \\
eval(\sigma, const) &= const \\
eval(\sigma, e_1 + e_2) &= \lambda e_1, e_2.(if\ e_1 \neq \top \wedge e_2 \neq \top)(eval(\sigma, e_1) + eval(\sigma, e_2))\ else\ \top \\
eval(\sigma, e_1 - e_2) &= \lambda e_1, e_2.(if\ e_1 \neq \top \wedge e_2 \neq \top)(eval(\sigma, e_1) - eval(\sigma, e_2))\ else\ \top \\
eval(\sigma, e_1 * e_2) &= \lambda e_1, e_2.(if\ e_1 \neq \top \wedge e_2 \neq \top)(eval(\sigma, e_1) * eval(\sigma, e_2))\ else\ \top \\
eval(\sigma, input) &= \lambda()(\top)
\end{aligned}$$

e is bound to $JOIN(C_=)$. $eval$ evaluates the expression and returns a value from the lattice $\mathcal{C}$. σ is the current environment, $\sigma(id)$ returns the current value, $const$ is a constant value from $\mathcal{C}$.

For entry node, the semantics is given by

$$\mathcal{A}[\text{entry}] = []$$

For any other node $C_\#$, the semantics is defined as follows:

$$\mathcal{A}[C_\#] = JOIN(C_\#)$$

Finally, the $JOIN(C)$ is defined as

$$JOIN(C) = \bigsqcup_{w \in pred(C)} [\![w]\!]$$

where $pred(C)$ is the predecessors of C.

Simple Example:

```
var x,y,z;
x = 27;
y = input;
z = 2 * x + y;
if (x < 0)
  y = z -3;
else
  y = 12
print y;
```

The CFG for the example program is straight forward and is not shown here. The $\mathcal{CP}$ lattice is initialized to [] for all the nodes in the control-flow graph. The semantics gives rise following equations

$$
\begin{aligned}
[\![\text{entry}]\!] &= [] \\
[\![\text{var x, y, z}]\!] &= \lambda(e).e[x \mapsto \bot, y \mapsto \bot, z \mapsto \bot]\ [\![\text{var x, y, z}]\!] \sqcup [\![\text{entry}]\!] \\
[\![x = 27]\!] &= \lambda(e).e[x \mapsto eval(e, 27)]\ [\![x = 27]\!] \sqcup [\![\text{var x, y, z}]\!] \\
[\![y = input]\!] &= \lambda(e).e[y \mapsto eval(e, input)]\ [\![y = input]\!] \sqcup [\![x = 27]\!] \\
[\![z = 2 * x + y]\!] &= \lambda(e).e[z \mapsto eval(e, 2 * x + y)]\ [\![z = 2 * x + y]\!] \sqcup [\![x = 27]\!] \\
[\![\text{if } (x < 0)]\!] &= [\![\text{if } (x < 0)]\!] \sqcup [\![z = 2 * x + y]\!] \\
[\![y = z - 3]\!] &= \lambda(e).e[y \mapsto eval(e, z - 3]\ [\![y = z - 3]\!] \sqcup [\![\text{if } (x < 0)]\!] \\
[\![y = 12]\!] &= \lambda(e).e[y \mapsto eval(e, 12]\ [\![y = 12]\!] \sqcup [\![\text{if } (x < 0)]\!] \\
[\![\text{print y}]\!] &= [\![\text{print y}]\!] \sqcup [\![y = z - 3]\!] \sqcup [\![y = 12]\!]
\end{aligned}
$$

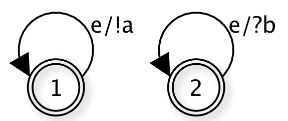
Note that $[\![c]\!]$ indicates the dataflow value at c. In this case, it is the map lattice. The worklist algorithm begins with $q = [[\![\text{entry}]\!], [\![\text{var x, y, z}]\!] \ldots [\![\text{print y}]\!]]$ The corresponding transfer function F_i is applied for the respective q_i and q_i is removed. The $succ(q_i)$ is determined by the edges of the control-flow graph. The worklist algorithm leaves us with following least fixed point.

$$
\begin{aligned}
[\![\text{entry}]\!] &= [] \\
[\![\text{var x, y, z}]\!] &= [x \mapsto \bot, y \mapsto \bot, z \mapsto \bot] \\
[\![x = 27]\!] &= [x \mapsto 27, y \mapsto \bot, z \mapsto \bot] \\
[\![y = input]\!] &= [x \mapsto 27, y \mapsto \top, z \mapsto \bot] \\
[\![z = 2 * x + y]\!] &= [x \mapsto 27, y \mapsto \top, z \mapsto \top] \\
[\![\text{if } (x < 0)]\!] &= [x \mapsto 27, y \mapsto \top, z \mapsto \top] \\
[\![y = z - 3]\!] &= [x \mapsto 27, y \mapsto \top, z \mapsto \top] \\
[\![y = 12]\!] &= [x \mapsto 27, y \mapsto 12, z \mapsto \top] \\
[\![\text{print y}]\!] &= [x \mapsto 27, y \mapsto \top, z \mapsto \top]
\end{aligned}
$$

Note that we did not track expressions in our $\mathcal{CP}$ lattice. Normally, expressions are also tracked. In SSA, this is straightforward as every expression gets a new variable and the $succ(q_i)$ will also be fewer as SSA directly encodes $def - use$ relations.

[Q4] Asynchronously Communicating Visibly Pushdown Systems:

Problem: Programming in asynchronous message passing paradigm is hard and reasoning about these systems is even more difficult due to the complexity of dealing with concurrency. Having a computationally tractable formal model for these systems simplifies the reasoning and enables various tools to be developed using these models. One such model is Communicating Finite State Machines (CFSM) where each process can be modeled as a FSM. Each FSM communicates through reliable unbounded queues. The FSMs read symbol from the queue, make a transition and possibly write symbols to output queues (act as a transducer). We can think of this model as a multi-tape automaton (n-tape). Let Σ be the alphabet and this multi-tape automaton operates on $(\Sigma \times \ldots \times \Sigma)^*$, The head of each transducer (FSM) reads a symbol at any state write symbols to their output queues. At any state, the snapshot of output queues of each transducer contains words that belong to regular languages (Finite state transducers produce regular languages). Lets define the queue configuration at any state of a multi-tape automaton to be a tuple of these languages. Any reachability of global control state or configuration involves examining the queue configuration for various states. The problem is unfortunately undecidable for general CFSMs. The decidability depends on relations between these queue languages. Pachl [3] observed that if the relations between these queue languages is recognizable then the reachability is decidable. A relation between tuple of regular languages is recognizable if concatenating all languages yield a regular language.



Consider the two finite state transducers shown in the figure on the left. Let the input symbol e denote ϵ. Let $!x$ indicate a symbol for send operation and $?y$ indicate a symbol for receive operation. They have only one state which is both initial and final. The first automata on ϵ move keeps sending and writes the symbol $!a$ to output queue for every send. The second automata keeps receiving and writes the symbol $?b$ to output queue. The words of their queues represent the following languages $L(Q_1) = a^+, L(Q_2) = b^+$ which is a recognizable relation. Recognizable relations can model simple protocols however they are limited in expressiveness and cannot express complex dependencies between queues. For example, consider the relation (a^n, b^n) – a client posts n request operations and server counts the requests and acknowledges them later. It responds exactly to those n requests. Such a protocol requires counting and the relation is not recognizable (no finite automaton accepts $a^n b^n$ which is a CFL). The expressiveness of such recognizable relations are limited. Furthermore, finite state machines are very coarse abstractions of real programs which have recursive structure.

Extensions: Babic et al in their paper [1] propose extensions in two directions. First, models for processes are visibly pushdown automata that can mimic control-flow of real programs with possible recursion. Second, for decidability of reachability queries they relax the recognizable relation constraint by allowing synchronizable relations which are more expressive. The key insight is the following – Decidability of reachability queries require language inclusion property between the queue languages. The language inclusion property is decidable for recognizable and synchronizable relations. By allowing synchronizable relations they can express more complex protocols. The two contributions are orthogonal. Lets briefly review Visibly Pushdown Automata (VPA) and synchronizable relations of regular languages relevant to the context of this work.

Visibly Pushdown Automata (VPA): Visibly Pushdown Automata introduced by Alur et al [4] is a pushdown automaton that operates over an alphabet set that is partitioned into three disjoint sets of calls, returns and local symbols. Reading a call symbol causes the automaton to push a symbol into the stack. Similarly reading a return symbol causes the automaton to pop a symbol from stack and local symbols cause no stack operations. VPA makes the actions of pushdown automaton explicit by controlling the input alphabet. A language over partitioned alphabet set is a Visibly Pushdown Language (VPL) if there is a VPA that accepts it. Formally a VPA on finite words over $< \Sigma_c, \Sigma_r, \Sigma_l >$ is a tuple $M = (Q, Q_{in}, \Gamma, \delta, Q_F)$ where Q is set of finite states, Γ is finite set of stack alphabets, Q_{in} set of initial states, Q_F set of final states. δ is a set of transition relations defined based on symbol from Σ_c or Σ_r or Σ_l. VPL is a strict sub-class of CFL and inherits all closure properties of regular languages. If L_1, L_2 are VPL, then $L1 \bigcup L_2, L_1 \bigcap L_2, L_1.L_2, L_1^*$ are also VPLs. The universality and inclusion problem are decidable for VPLs. Another useful property shown by [4] is that VPL correspond to regular tree languages ie each word of a VPL correspond to a tree from a regular tree set. [4] shows a construction of converting VPL words to stack-trees that is relevant to this paper. For verification using VPLs, we can encode a boolean program $\mathcal{P}$ (predicate abstraction) and its transitions using the alphabets $< \Sigma_c, \Sigma_r, \Sigma_l >$. $\mathcal{P}$ is now a generator of VPL. We can encode the specification ϕ as another VPL and check if $L(\mathcal{P}) \subseteq \phi$.

Synchronizable Relations: A relation between tuple of regular languages is synchronizable if the concatenation of the languages can be accepted by a synchronized n-tape automaton or synchronized automata. Let Σ be the alphabet and a synchronized automata operates over $\Sigma \times \ldots \times \Sigma)^*$. The heads of the automaton move in a synchronized lock-step fashion. For example (a^m, b^m) form a synchronized relation as two-tape automata can accept (a^m, b^m) Additionally (a^m, b^k) such that $k \geq m$ can also be recognized by synchronized automata by padding the tape with additional symbols. Synchronizable relations have the decidable language inclusion property and are strictly more expressive than recognizable relations. The relation a^m, b^k can model protocols of arbitrary message passing programs. Resynchronizable relations are those accepted by n-tape automaton whose tapes are not synchronized but the distance between the tape head is apriori bounded. For example $(b^m aab^k, c^m c^k)$ is a resynchronizable relation. After reading (b^m, c^m), the first tape head scans two extra symbols and the two tape heads can sync for the rest of the word. Resynchronizable relations can be cast as synchronizable relations which is exploited in this work. N-tape automata which are completely asynchronous give rise to rational relations whose language inclusion property is undecidable.

Communicating Visibly Pushdown Transducers (CVPT): Each process in an asynchronous system receive words of input messages, process them and produces words of output messages. The individual processes can be modeled as a transducer. Instead of FSMs, each process is modeled as a VPA with finite state control, unbounded stack, finite set of reliable queues of unbounded length. The output of each such machine produce is a VPL. A CVPT is defined as $T = (\Sigma_{rcv}, \Sigma_{snd}, Q, S, I, F, \Gamma, \Delta)$ where Σ_{rcv} is disjoint union given by $\Sigma_{rcv} = \Sigma_c \cup \Sigma_r \cup \Sigma_l$, Σ_{snd} is a set of output alphabet which is a disjoint union given by $\Sigma_{snd} = \bigcup_{q_i \in Q} \Sigma_{q_i}$ where each Σ_{q_i} is a set of alphabet for a queue q_i, Q is a finite set of unbounded FIFO queues, S is set of states, I is set of initial states and F is set of final states. The transition relation is classified into three (i) call (ii) return (iii) local. Call transitions pushes symbols into the stack, return pops them and local does not modify stack. All transitions can remove a symbol $?m_i$ $(?x \in \Sigma_{rcv})$ from a $q_i \in Q$. Similarly all transitions can add a symbol $!m_2$ $(!x \in \Sigma_{snd})$ from a $q_i \in Q$. Each machine reads symbols from Σ_{rcv} from a queue $q_i \in Q$ and outputs symbols from Σ_{snd} to another queue $q_j \in Q$. The state configuration C is given as tuple $C = (s, \sigma, \vec{\rho})$ where σ is the word on stack, let ρ_i be the content of $q_i \in Q$ and $\vec{\rho} = \rho_1 \rho_2 .. \rho_{|Q|}$.

Asynchronous System of CVPTs: The CVPTs are composed into system of CVPTs given by $M = (T_1, \ldots T_n)$ where each $T_i = (\Sigma_{rcv_i}, \Sigma_{snd_i}, Q_i, S_i, I_i, F_i, \Gamma_i, \Delta_i)$. The composite configuration is given by $\vec{C} = (C_1, C_2 \ldots C_n)$ where each $C_i = (s_i, \sigma_i, \vec{\rho_i})$. Let the vectors $\vec{s}, \vec{\sigma}, \vec{\rho}$ denote the composite state configuration, stack configuration and queue configuration respectively where each s_i, σ_i, ρ_i is from respective T_i. For a given state let $L_q(\vec{s}) = \vec{s}$, let $\vec{C}.\varrho$ denote full queue configuration and let $L_{qs}(\vec{s}) = \vec{C}.\varsigma, \vec{C}.\varrho$ represent the full stack-queue configuration. Each of these configurations $L_q(\vec{s}), L_{qs}(\vec{s})$ form a n-ary relation and note that the words in the queue are from VPL.

Stack-Tree Relations: We know that the decidability of reachability queries require that relations between regular languages be synchronizable to handle relations such as a^n, b^n. The words produced by the each transducer CVPT is a VPL. The result by Alur et al [4] show that VPL correspond to regular tree languages which have all the properties of regular language. Each word in VPL can be translated into a stack-tree based on the construction by [4]. Each VPL language in the queue configurations L_q, L_{qs} now correspond to a regular (stack-) tree language. We have tuples of regular (stack-) tree languages in the configurations which have a recognizable relation (concatenation all of regular language is a regular language). Using a technique called overlap encoding, they extend the relation between regular tree languages to a synchronizable tree relation denoted by $Sync^{\mathcal{V}}$.

Main Result: Reachability is shown to be decidable for a system of CVPTs that satisfy composition and the synchronized configuration property (L_q, L_{qs} satisfy $Sync^{\mathcal{V}}$). CVPTs can generate CFLs as output.

To restrict this, composition property which restricts each CVPTs to produce only VPLs is introduced. With the composition and synchronization property satisfied the reachability query on a system of CVPTs is decidable. It is easy to see that a system of CVPTs can model message passing and asynchronous task based systems by having each process/task modeled as a CVPT and the communication using queues of unbounded length.

Applicability to HPC applications: To be applicable to HPC applications, the formal model based on communicating visibly push down systems should handle full MPI semantics. Non-blocking operations can be modeled by counting on the stack ie push a symbol to stack when we see non-blocking operation and pop them when we see a corresponding wait. VPAs can write to multiple queues at a transition and therefore collectives can also be modeled. To handle non-determinism, two techniques (i) FIFO queues replaced with multi-set queues (bag-like) (ii) non-deterministic selection of queue to read from by a receiver are proposed. Note that VPA also have determinization property. The only restrictions to apply these models are the properties (i) composition (VPA should not generate CFL) (ii) synchronizable relation need to be satisfied. The composition property can be visualized as follows: if G is a graph with nodes as CVPTs, composition property requires a directed edge between T_i and T_j if they communicate. MPI applications do not have complex control-flow and would always allow for satisfying this requirement. Synchronizable relations are surprisingly expressive that can capture various communication protocols in HPC applications. Consider the BSP model typically employed in which process frequently communicate between neighbors to exchange ghost cells. The neighbors are fixed and protocol does not have any dependency with neighbors. The techreport [2] demonstrate a non-trivial example in which a client send n requests for jobs, the server postpones the request but it counts the number of requests and for each request the server responds with a data which acknowledged again by a client. This example encodes typical client server model. MPI-Blast can be modeled as a variant of this example. A study of various protocols of HPC applications may reveal if they fall into synchronizable relations. However, computing if a relation is synchronizable is undecidable which makes this even more difficult. Furthermore, while the reachability problem is decidable for this model, it is computationally very expensive

Extension to Unbounded Actors: The communication between CVPT is through finite set of FIFO queues of unbounded length. Arguably this set can be arbitrarily large. To handle infinite actors, each CVPT must be associated with infinite set of FIFO queues. The consequence of this is that each CVPT could generate words of infinite length. The VPL languages are now ω-VPL recognizable by a ω-VPA shown by Alur et al [4] (similar to Buchi automata for ω-regular languages). It is not clear if ω-VPL correspond to ω-regular tree languages. Furthermore, the foundations of this work depends on synchronizable relations which is based on n-tape automaton. If each of these automaton of n-tape machine are buchi-automaton to accept ω-regular stack-tree language, then the question is if the language inclusion property still holds. Perhaps there are opportunities for this extension following the approach of pCFG. Each VPA now models a set of processes and we have finite number of VPAs. Each queue now models communication between two sets of processes. pCFG's splitting and merging can be defined as operations over queues. It is however worth investigating if such queue operations are possible and would also result in VPL.

References

[1] *Asynchronously Communicating Visibly Pushdown Systems*, Domagoj Babic, Zvonimir Rakamaric, FORTE, 2013.

[2] *Asynchronously Communicating Visibly Pushdown Systems*, Domagoj Babic, Zvonimir Rakamaric, UCB-TR, 2012.

[3] *Protocol Description and Analysis Based on a State Transition Model with Channel Expressions*, J.K. Pachl, PSTV, 1987.

[4] *Visibly Pushdown Languages*, Rajeev Alur, P. Madhusudhan, STOC, 2004.

[Q5] Conditional Model Checking: A Technique to Pass Information between Verifiers – Dirk Beyer, Thomas A Henzinger, M. Erkan Keremoglu, Philipp Wendler

Model checking of software (an undecidable problem) is an automatic search based procedure to verify that a given model for the program such as a labeled transition system satisfies a specification such as a temporal logic property. The outcome of model checking algorithm is one of the following (i) program satisfies the property (ii) program violates the property (iii) unknown – model checker runs out of resources (time, memory). The key observation in this paper [1] is that significant resources are used before computing unknown and yet no useful result is reported. The intuition is that if the state space that was already proved safe is summarized a second tool can focus the search on unverified state space. The goal of this paper is to summarize the result of a model checker for further verification efforts. For this purpose, this paper proposes *conditional model checking* which reformulates model checking problem as follows: Given a program, a property and an input condition the model checker emits an output condition Ψ such that program satisfies the property under the condition Ψ. Ψ represents the state space that has been verified (reachable states that do not violate the property). The paper makes two contributions (i) defines *Conditional Model Checking (CMC)* (ii) Sequential Combination of Model Checkers – applies the CMC technique to the problem of sequentially composing several model checking techniques. CMC provides a representation for model checker summary (Ψ) that enables information to be passed to other model checkers.

```
void main() {
  if(*) {
    for(int i = *; i < 1000000; i++);
    assert(i >= 1000000);
  } else {
    int x = 5, y =6;
    int r = x * y;
    assert(r >= x);
  }
```

To motivate need for CMC, consider the code shown on the left. Model checking based on predicates rely on SMT solvers and can only verify theories supported by them. In the example, given predicate $i \geq 1000000$ the predicate analysis will be easily able to prove the assertion in the true branch. However it cannot prove the assertion in the false branch as it requires non-linear relations which are modeled as uninterpreted functions and the model checker gives up. Consider explicit-value analysis of integers. It will not be able to prove the assertion in the true branch as it keeps unwinding and gives up exhausting available resources. However it can easily verify assertion in the false branch. Predicate analysis can restrict the state space for explicit-value analysis by providing condition Ψ that covers the first assertion and therefore explicit-value analysis can easily verify the second assertion. The example demonstrates that the combination of model checkers is beneficial and the experiments show that CMC improves verification coverage and efficiency.

Conditional Model Checking: In CMC, a model checker needs to output a condition that summarizes its work. This condition should be representable in a format that can be easily read by another model checker. Model checking is typically performed on abstract domain which yields abstract reachability tree (ART). In ART, nodes correspond to abstract states and edges correspond to transition relation. For the format of output condition Ψ, CMC reduces the ART to produce an assumption automaton. Each transition of this assumption automaton corresponds to a control-flow edge and is labeled with an assumption by the model checker that processed this edge. An assumption is produced for a transition if the analysis decides to skip certain path or if it failed to provide a full safety proof (output of the model checker is unknown). Subtrees of ART for which all transitions are labeled *true* are folded into one sink node T. Naturally a path that has been verified will be folded into a single edge to T in the assumption automaton. For all other parts of ART, there is a one-one correspondence between ART nodes and states of assumption automaton. The assumption automaton serves as input to a second model checker. This model checker is marched whenever the assumption automaton can make a transition. The exploration of a path can be stopped when the automaton reaches T. The second model checker only analyzes those paths in the assumption automaton that contains at least one transition labeled not *true* as the verified paths transitions directly to T.

The experiments demonstrate practical benefits of conditional model checking applied to sequential combination several model checkers with information passing. From the benchmark suite, CMC technique was able to verify 8 additional programs that were not possible earlier. The performance numbers also demonstrate significant improvement in efficiency. CMC techniques increases efficiency of model checking tools and improves verification coverage.

Discuss how you may be able to benefit from their ideas in the context of FUSE project

The key to compose analyses or model checkers is to find a common representation for results. CMC summarizes the results of model checkers as assumption automaton. $Fuse$ [2] summarizes the results of dataflow analysis as an Abstract Transition System (ATS) and provides a query interface to access these results. For comparing CMC and $Fuse$, the following two questions are useful.

- *What is the difference between the two representation – assumption automaton and ATS?*
- *Can $Fuse$ and CMC interact in both directions?*

Difference between the two representation – assumption automaton and ATS: An assumption automaton is a reduced representation of an Abstract Reachability Tree (ART). ART and ATS are closely related as both of them are constructed on top of control-flow graph – their nodes have one-one correspondence with the nodes of control-flow graph and their edges correspond to edges of the control-flow graph. Each node of an ART is an abstract state defined by a predicate over some abstract domain $\mathcal{D}_1$ as computed by the model checker. Furthermore, the edges of an ART are annotated with assumption predicates by the model checker. Each node of an ATS denotes an equivalence class of sub-executions where two sub-executions are in this equivalence class if they satisfy a predicate (computed by dataflow analysis) over some abstract domain $\mathcal{D}_2$. The edges of an ATS are annotated with dataflow state which is accessible to other analysis through query interface defined by $Fuse$. A *Galois connection* $(\mathcal{D}_1, \alpha, \gamma, \mathcal{D}_2)$ can be defined between the two domains and a relation can be formulated between the nodes of ART and ATS. The key difference however is that ART can potentially be an infinite tree where as ATS is a finite graph. It is due to the following – ART is computed by model checking which is path-sensitive whereas ATS is computed by dataflow analysis that merges information flowing from different paths (path-insensitive). However it is possible to define flow analysis with no join operation in which case it would compute an ART. For assumption automaton, ART is reduced by folding execution paths that are already proven correct (edges are labeled $true$ on the entire path). An assumption automaton prunes out the verified paths leaving only unverified paths for a subsequent model checker thereby communicating effectively what has been already verified. However, the lack of query interface restricts the information that can be communicated such as abstractions computed by previous model checker where as $Fuse$'s communication mechanism with query interface is much more powerful.

Can $Fuse$ and CMC interact in both directions: Having established the similarity between ART and ATS, it is possible to encode an assumption automaton (reduced ART) as an ATS and vice-versa. Consider the example shown earlier which required model checking with predicates and model checking with explicit-value to co-operate. The second assertion in the example can be easily proved by a constant propagation analysis. The assumption automaton can be encoded as an ATS on which constant propagation analysis can be run to prove the second assertion. A dataflow analysis in $Fuse$ can run on this ATS and focus only on the unverified paths. The precision of the analysis can potentially improve as it has to focus only on certain paths. $Fuse$ can also produce an assumption automaton by reducing the ATS. If a path-sensitive analysis can prove the safety of a path, $Fuse$ can eliminate the path to produce an assumption automaton on which CMC can focus on unverified paths. In the example, $Fuse$ will be able to verify the second assertion and reduce the ATS to an assumption automaton on which CMC can verify the first assertion. For interprocedural, with path-sensitivity and context-sensitivity the ATS computed by $Fuse$ resembles an ART.

References

[1] *Conditional Model Checking: A Technique to Pass Information between Verifiers*, Dirk Beyer, Thomas A Henzinger, M. Erkan Keremoglu, Philipp Wendler, FSE, 2012

[2] *Combining Program Analysis via Abstract Transition Systems*, Greg Bronevetsky, Michael G Burke, Jisheng Zhao, Sriram Aananthakrishnan, Vivek Sarkar, LLNL-TR 2013

PhD Written Qualifier Exam

Examiners
Ganesh Gopalakrishnan
Ryan Stutsman
Zvonimir Rakamarić
Matthew Flatt

Examinee
Mohammed S. Al-Mahfoudh
mahfoudh@cs.utah.edu

May 2, 2016 – May 9, 2016

Contents

1 Ganesh Gopalakrishnan Questions

Read a paper on the "Global Sequence Protocol" by Burkhardt et al [3].

1.1 Describe its advantages vis-a-vis maintaining global state and propagating replicas.

There are several advantages for GSP over global states propagating replicas:

1. *Syntactic clarity, simplicity, and conciseness*: Replicated data types can be declared as distributes in a similar way to local objects (by declaring it using *cloud* keyword, that is after defining their data model. Operations on them are called exactly the way they are called on *local objects* [3].
2. *Replication and location transparency*: Data objects that are declared distributed, get replicated automatically and transparently for the client code. More over, if a read/update is issued for a certain data type, the protocol handles reading/updating from/to local store or waiting for updates to take effect on the local state (when strong consistency or synchronization is specified inside the abstract model operations implementation) then returns the appropriate value [3, 5].
3. *Network failure transparency*: If a network connection fails, either on the server side or client side, data are cached (updates and reads) in a buffer as well as local replica to preserve asynchrony and responsiveness, respectively. When the connection is restored, those updates are propagated to the server to broadcast them to clients, so they *eventually* converge to the same state [3, 5].
4. *Communication transparency*: A client code never communicates explicitly with any participating party. Yet, its updates get delivered appropriately to others and others updates to it [3, 5].
5. *Transaction handling transparency*: When transactions are to be implemented, due to communication transparency, clients completely ignore *how* the transaction is handled *as long as* they provide the right data model and implementation of its higher-level abstract operations [3, 5].
6. *Transactions never fail*: If a partition happened that didn't last forever between clients and the GSP server, these transactions are resumed as if there was no interruptions. This is in main part due to the local buffering, caching and total ordering of updates the protocol enforces [3].
7. *Flexibility in data consistency model*: It is completely up to the *data model* (defined by the developer as a sequential-looking data object) to control the granularity of transactions as well as synchronization. For example, operations of these models can be implemented using the transactional extension primitives (pull,push, and flush) to encode as strong or as weak consistency model (linearizable, sequentially consistent, causally consistent, eventually consistent, or quiescently consistent) [3, 5]. All of this, without exposing client code to *any* explicit communication and/or concurrency interleaving/handling. Developers though need be careful how they design their data models operations to avoid surprises such as *data races* [3, 5].
8. *Complete abstraction of network and communication*: If it was not emphasized enough already, client code is unaware of any communications nor it needs to in order to participate in replication of shared state. Everything is done by the protocol and the data model support implemented by the clients [3, 5].

The above list of advantages is *at least* impressive looking to how elegant the protocol is, and how composable and extensible its data model is. However, it does come with some concurrency problems as mentioned.

1.2 What are the semantic clarity advantages claimed (describe in 1-2 pages with illustrations).

The *semantic clarity* comes from few facts that will be apparent after giving an overview of GSP.

Overview of GSP. The inspiration of the GSP protocol is the Total Store Order (TSO) [3, 5]. The latter is used by multi-core processors to model weak memory behaviors, i.e. cache coherence between individual cores-caches and the main memory in shared memory systems [3, 5]. TSO has clear specification and many developers can both reason about it informally [3, 5], and use it to reason about weak memory models. It updates a local state, and buffers non-local updates but waits for other updates to arrive synchronously [5]. GSP only adds another buffer for incoming updates to accommodate asynchronous reading [5]. This is in addition to reading-its-own-writes and merging updates from others to its writes using the total ordering of updates the protocol provides [5]. Another fact is that GSP relies on Reliable Total Order Broadcast (RTOB) protocol, which orders all updates and then broadcasts them its clients. In addition, GSP protocol builds on *abstract data types*, that are abstractly descried and implemented by the developer, regardless of distribution concerns [1]. Only after using the abstract data model, and indicating (using a keyword) that the object is a replicated data type (i.e. distributed shared state), the protocol can handle everything for them [3]. All consistency models are determined using the implementation of those operations of the abstract data type, using the Transactional GSP primitives pull, push, flush [3]. The first two (i.e. pull and push) are used to batch read and batch updates, respectively. Together and the flush operation, that waits for confirmation of updates [3, 5], provide transactional support to clients.

The semantic clarity advantages. It is for these four elements that GSP has the semantic clarity advantages claims:

1. *A unified abstract data types model*: supporting all kinds of replicated data types by categorizing a data type operations under *Update* and *Read* types, from which a binding for the *rvalue:Read* $\times$ *Update** $\rightarrow$ *Value* is determined, i.e. the returned value based on initial state read and a sequence of updates (updates are data type abstract operations modifying its state, and reads are those that read but do not modify the state) [3].
2. *Total Order of Updates*: GSP server orders updates, and clients keep track of pending updates by them or other clients. Once clients receive confirmations it removes those pending operations and places them in the same order they were done in its *known* state. Incoming updates conflicting with local updates are also merged correctly using ordering [3].
3. *Similarity in operation and buffering of TSO*: This is the reason why it was clear to represent distributed shared state as an initial state and a sequence of operations done on it, i.e. a *Delta* object specific to a certain data type.
4. *All leads to clean code*: The code resulting from a GSP compliant application doesn't contain any communication and faults handling, looking more like normal sequential code [3, 5]. That code is easy to *reason about*, can use *judicious synchronization*, and constitutes a *robust* implementation of a distributed protocol and consistency model(s) [3].

Illustrated TSO. [2] TSO (shown on Figure 1 [5]) has the ability to *synchronously* perform reads, from its local cache, and *asynchronously* (i.e. can be delayed) to shared memory [5]. Synchronous reads take into considerations the buffer content for the write operations, hence TSO *reads its own writes* to keep its reads coherent with its writes [5].

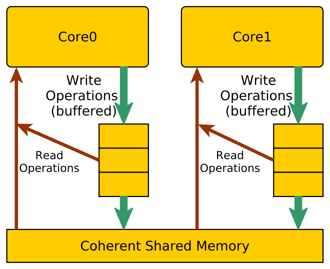


Figure 1: Total Order Store (TSO).

Illustrated GSP. It is *almost* exactly like TSO, but now reads are buffered [3, 5]. That is, updates coming from the RTOB protocol (the cloud) are buffered, and then the read operation chooses weather to read its own replica, or merges updates from the buffer then read [5]. This is illustrated in Figure 2 [5]. An important detail is that now *writes*, also, update local replicas. So the client can see its own writes and reads them immediately (*read-your-own-writes*) [5]. In addition, the clients of GSP have full replicas of the shared data.

[1]With the exception of data races that can be avoided by providing higher-level operations on those data types.
[2]Illustrations on both TSO and GSP are completely reproduced from GSP presentation on youtube [5]

Subtle but significant differences between GSP and TSO. Two main differences between GSP and TSO. First, TSO reads are synchronous (no buffer) while GSP reads are asynchronous (there is a buffer for incoming updates though it can read from local state if synchronization isn't needed) [5]. Second, while TSO relies on a processor's clock to order updates, GSP relies on a distributed protocol (RTOB) to order its updates [5]. These subtle differences is what makes GSP applicable to distributed setups, flexible to any consistency model, and yet easy to reason about [5].

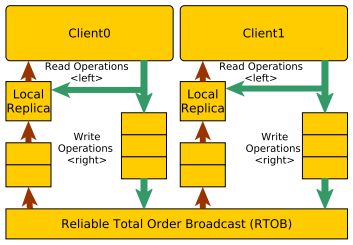


Figure 2: GSP protocol

Examples illustrating GSP abstract data model. Many replicated abstract data types can be developed, some examples of which follow:

- **Register**: whose *Update*[3] set of operations is $\{wr(v)\}$, and *Read* set of operations is $\{rd\}$ [3, 5].
- **Counter**: with $Update = \{add(v)|v \in Int\}$, $Read = \{rd\}$ [3, 5].
- **Key-Value store**: with $Update = \{wr(k, v)|k, v \in Value\}$ and $Read = \{rd(k)|k \in Value\}$ [3, 5].

About examples and proofs. There are several examples in the detailed technical report [3] and animated ones in the video presentation [5] that illustrates how easy to reason about a race condition in terms of update operations. That illustrates some of the semantic clarity in GSP and its abstract data model. More over, proofs at the end of the technical report seem to be readable [4] and intelligible, thanks in part to the clarity of the protocol's abstract model [3]. I is evident how semantically clear the global sequence protocol is, even in the most complex activities such as proving properties.

1.3 What are the implementation overheads (half a page).

GSP comes with few overheads as it is implemented so far. The following is a set of taxonomized paragraphs of what overheads are caused by which parts of the implementation.

Developer overheads. For data models that are not defined yet, developers need to provide their own implementation for their *abstract data models*. Also, they need to make sure they correctly use synchronization and transactional primitives provided by GSP, or their code may end up with *data races* [3]. It is this part where our model exploration can help detecting failure situations.

Network bandwidth overhead. The state transferred is transferred in combination with the extra *delta objects* (encoding the update sequences). That delta object is an extra bandwidth overhead on the already limited network bandwidth. This is especially true since every update sequence is broadcast back and fourth between the GSP server and clients [3].

Processing and storage overheads. Delta objects are the objects encoding and *reducing* the update sequences to transfer over network to other parties [3]. The reduction step of the update sequence needs be *processed* and then *stored* till other participants acknowledge their receipt [3]. In addition, the participants Keep track of the update sequences (*known* and *pending*). That is both an overhead on storage and processing on the clients side of the protocol, and it grows worse for richer data models [3]. More over, keeping a copy of the state object on *all* parties' cache occupies possibly scarce storage [3].

Throughput overheads. The very same *network* and *processing* overheads above, cause reduced throughput and performance of the whole distributed system.

[3]Capitalization is to emphasize the equivalence of these operations to the Update and Read set of operations, equivalent to the abstraction of data model provided by GSP

[4]However, I did not have the luxury of time to go through all of the content, I only glimpsed to make sure of this fact.

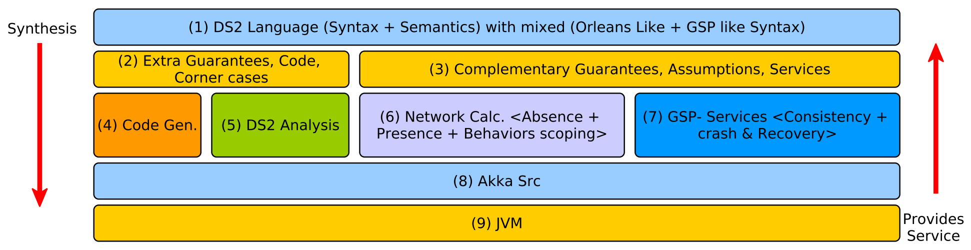

Synthesis
(1) DS2 Language (Syntax + Semantics) with mixed (Orleans Like + GSP like Syntax)
(2) Extra Guarantees, Code, Corner cases
(3) Complementary Guarantees, Assumptions, Services
(4) Code Gen.
(5) DS2 Analysis
(6) Network Calc. <Absence + Presence + Behaviors scoping>
(7) GSP- Services <Consistency + crash & Recovery>
(8) Akka Src
(9) JVM
Provides Service

Figure 3: DS2 Runtime: How each module contributes to the runtime services and guarantees

Response time overhead. Round trip to and from the GSP server for strongly consistent updates, and when at worst the GSP server can stall for long times, affects responsiveness. However, in many cases this can be mitigated with higher level operation for the data models [3]. In addition, reduced throughput leads to longer response times.

1.4 Where do you think it impacts your work on DS2? (one page).

It impacts my work on DS2 in several areas:

1. *Designing the runtime of DS2*: By making GSP a baseline (complemented by the network calculus (NC) publish/subscribe guarantees [4, 10]) for our synthesized code, we can ignore synthesizing code for handling those cases already covered by GSP, such as consistency model. Instead, we provide facilities in the *DS2 language syntax* to express what is express-able by GSP, and then rely on those expressions to rule out kinds of consistency/fault concerns covered by GSP [3, 5].

2. *Model checking/exploration of distributed system*: GSP represents guarantees by design, so it serves as a filter for large amount of error/faults sources in a distributed system. Assuming we would develop that runtime to take benefit from the complementary GSP [3, 5] and NC [4, 10], then all that remains for the model checking is to explore the remaining corner cases of GSP (and those not covered by NC). Then it warns the user about them. Later, DS2 synthesizes *best effort* code that can handle those situations.

3. *Higher level syntax expressing consistency*: GSP is very expressive of all gradients of *Strong Consistency* (i.e. Linearizability, and Sequential Consistency) [5] and *Eventual Consistency* that models asynchronous update propagation (i.e. causal consistency, eventual consistency, and quiescent consistency) [5]. Hence, providing similar syntax based on the transactional version of GSP primitives (i.e. push, pull, flush) and the finer-grained primitives of Core-GSP (Read, Update) makes a great step forward abstract, yet expressive, specification of distributed systems [3, 5]. That would provide the composability Concurrent Replicated Data Types (CRDT) lack and makes DS2 intelligible by developers and easy to *informally* reason about [3, 5]. However, even if it was also easier to reason about informally, still formal reasoning is needed to assure stronger guarantees [3, 5].

4. *Further simplifying syntax*: If we chose to ignore DS2 and Orlean-like syntax [1] in favor of a completely sequential-looking code, then we can do so by declaring distributed shared state as say "cloud" or "distributed", then we let GSP takes care of things, and the model exploration algorithm to explore where the programmer failed to conform to GSP semantics, e.g. data race detection.

5. *Introducing network/location transparency*: Whether we opt for similar syntax shown in the GSP paper [3] or not, GSP can provide network transparency for DS2 developers on the cheap.

6. *Elevated from data consistency analysis*: Since data consistency models can all be handled with GSP, DS2 need not re-invent the wheel. It can very much rely on the same protocol. Some concerns are stated in the highlights section.

For an overview of a tentative stack of DS2 runtime incorporating GSP (as well as Orleans [1] and the network as a construct [4]), please refer to Figure 3. Layer (1) provides syntax with multiple granularity of

[5]this is true as long as there is no network partition

communication control, GSP being the highest level, followed by Orleans-like (async RPC [8] and RMI [6] like) syntax for finer control. Once the developer drops to Orleans syntax, data structures involved in the communications are auto-removed from GSP support, if they were declared using *cloud* keyword. As a result, they are subjected to more analysis by layer (5) and more code may need be generated by layer (4) to take care of anticipated corner cases. Layer (3) provides guarantees relying on services from layers (6) and (7). Layer (2) provides what layer (3) misses in guarantees and provides analysis. Layer (5) generates code with help of layer(4) analyses results. All code generated is Akka code, layer (8), which is run on top of Java Virtual Machine (JVM).

1.5 Provide any other highlights, making your total answer 5 pages.

Related work. Closest work to GSP is **Bayou's weakly consistent replication** [9], yet it requires writing a merge function provided by the user. GSP does this automatically by ordering updates [3]. **CRDT's** provide optimized distributed protocols as in GSP but they are not composable nor easily customize-able [3]. This is due to the the protocols logic being not cleanly separated from the data model [3]. Protocols of CRDT's are also specialized to certain data type, as opposed to GSP's generic composable formulation [3]. **The Jupiter system** [7] provides similar streaming model as in GSP [3] but uses *operational transformation* (OT) algorithm to transform conflicting updates. GSP simply orders updates sequentially [3]. The OT approach is a powerful conflict resolution mechanism, however it is confusing, error-prone and not scalable [3]. The paper mentions that this choice of avoiding OT may be revised in the future [3]. Other than that, GSP **relies on a protocol called Reliable Total Order Broadcast(RTOB)** [2] in its operation and is highly inspired by the Total Store Order (TSO) protocol [3, 5]. GSP, hence, is safely called the "the TSO of distributed systems", but they are clearly *not equivalent*.

Relation to DS2. Although GSP provides a great abstraction and reasoning model, it still does not completely cover the case of **data races**. For example, the developer is left alone to figure out if his/her code is exhibiting data races or not. They are required to provide *more abstract* operations so that the protocol can handle them. DS2 analysis infrastructure may play an important role detecting those. Another issue, the **use of GSP in embedded setups** (those who have limited data storage, processing power, and network bandwidth) brings some *performance and responsiveness* concerns. Such systems are to be addressed by DS2. If DS2 is to use GSP to implement part of its runtime, optimizations may need to be figured out. This is to make sure they stay performant and responsive, if at all possible. That can be avoided by dropping to lower level syntax, e.g. RPC like as discussed in §1.4. However, even if the developer is to drop to lower level abstractions, DS2 analysis module should be of great help.

References

[1] P. A. Bernstein et al. *Orleans: Distributed Virtual Actors for Programmability and Scalability*. Tech. rep. MSR-TR-2014-41. 2014. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=210931.

[2] R. Boichat and R. Guerraoui. "Reliable and total order broadcast in the crash-recovery model". In: *J. Parallel Distrib. Comput.* 65.4 (2005), pp. 397–413.

[3] S. Burckhardt et al. *Global Sequence Protocol: A Robust Abstraction for Replicated Shared State (Extended version)*. Tech. rep. MSR-TR-2015-11. Microsoft Research, 2015. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=240462.

[4] T. Garnock-Jones, S. Tobin-Hochstadt, and M. Felleisen. "The Network as a Language Construct". In: *ESOP*. Vol. 8410. Lecture Notes in Computer Science. Springer, 2014, pp. 473–492.

[5] *Global Sequence Protocol: A Robust Abstraction for Replicated Shared State*. https://www.youtube.com/watch?v=7rQ0Ul3iPmo, Retrieved May 2, 2016.

[6] *Java remote method invocation*. https://en.wikipedia.org/wiki/Java_remote_method_invocation, Retrieved May 3, 2016.

[7] D. A. Nichols et al. "High-latency, Low-bandwidth Windowing in the Jupiter Collaboration System". In: *Proceedings of the 8th Annual ACM Symposium on User Interface and Software Technology*. UIST '95. New York, NY, USA: ACM, 1995, pp. 111–120. ISBN: 0-89791-709-X. DOI: 10.1145/215585.215706. URL: http://doi.acm.org/10.1145/215585.215706.

[8] *Remote procedure call*. https://en.wikipedia.org/wiki/Remote_procedure_call, Retrieved May 3, 2016.

[9] D. Terry et al. "Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System". In: *In Proceedings of the Fifteenth ACM Symposium on Operating Systems Principles*. 1995, pp. 172–183.

[10] *The Network as a Language Construct*. http://www.ccs.neu.edu/home/tonyg/esop2014/, Retrieved May 2, 2016.

2 Ryan Stutsman Questions

Orleans [2] is an example of a high-level Actor framework deployed at Microsoft in the real world and at a large scale. Read the paper and comment on the following.

2.1 In what ways does Orleans raise (or even lower) the level of abstraction beyond lower-level frameworks like Akka?

Orleans mostly raises abstraction levels over actors frameworks such as Akka and the likes. It, however, lowers abstractions for performance or efficiency (supporting scalability) reasons.

Ways it raises abstractions. Orleans **raises** abstraction over lower level frameworks such as Akka [1] [2, 7, 6] in several ways:

1. It leverages an intuitive and widely familiar OOP interface with asynchronous methods (returning `Task/Task<RsltT>` to optionally block on till they are *fulfilled*), instead of messages sent around [2, 6, 8]. This allows complex modeling of distributed systems without the need for reasoning about them using lower level primitives such as message passing, delivery guarantees, and such. These methods are identical in semantics to Akka's and DS2's **ask** pattern [3][1].
2. Compiler checked method calls in Orleans, instead of runtime-checked messages passed in Akka-like frameworks [2].
3. The concept of an ever existing *virtual actor*s makes it possible to manage the physical actors backing the runtime automatically and without much involvement of application code. All of this while the application code assumes *always* existing virtual actors (called Grains) [2, 7, 6, 8].
4. Removing race conditions completely while running a single actor's tasks but only when a non-*reentrant* virtual actor is used for modeling concurrency.
5. The runtime takes full responsibility for handling failures and recovery of *physical actors* as well as hardware [2] and does that automatically.
6. Abstracting away the management of actors *lifecycle*, i.e. creation, killing of actors removing the need for that to bloat the application code, and avoiding hindrance of developers productivity.
7. Orleans provides placement of actors (and thus distributed resource management such as load balancing, deactivation and reclamation of actor used resources, and recovery after failures). That, in turn, provides perfect *location transparency* to application code [2]. Since actors are addressed by virtual references in comparison to Akka, whose actors are addressed by a URL part of which specifies the host/port pairs and actor identifier [1].
8. Bringing the *stateful* vs *stateless* distinction is a great way for avoiding consistency checks on *stateless* actors, and to indicate that this is only a processor actor [2].
9. Automatic scaling through providing *single activation* mode (default) or *stateless worker* mode that utilizes multiple activations (to a certain limit) to increase throughput (and hence performance) [2]. However, similar features can be *configured/enabled* in Akka [1] using elastic clusters.

Orleans also lowers few abstractions in comparison to Akka. Some abstractions from frameworks such as Akka and that are brought to lower level by Orleans, are mostly due to performance or scalability reasons:

1. Delivery guarantees provided by Orleans is *at-least-once* potentially causing duplication of messages sent [2], while frameworks such as Akka have a higher level guarantee *at-most-once* [1][2].
2. Normally Akka [1] has a FIFO ordering on the receiver side, Orleans chose to lower that to be un-ordered for efficiency/performance reasons. This is because FIFO ordering of delivered messages is rarely needed in the application domains using Orleans [2]. In addition, maintaining a lot of actors all with FIFOs and sequence numbers assigned by each actor to messages going out constitutes a $O(n^2)$ storage requirement (by accounting for each pair of actors) [2]. This is too much to account for if Orleans is to stay performant under millions or more of concurrently active entities/clients [2].

3. Akka provides an implementation for transitive transactions that can ensure invocation of a *set of actors on a single host* is *atomic*, Orleans leaves that to the application [2].
4. Akka, also, can load *new code* into an actor *during runtime*, Orleans doesn't support those features [2]. However, this is barely a feature instead of an abstraction level.

2.2 Would a higher-level model like Orleans be more or less appropriate for DS2?

First, let us define some terms. We have *DS2 model* which is the generic distributed systems analysis infrastructure that is based on Akka [1]. In addition, we have *DS2 language* which is the higher level language for synthesizing distributed systems, from *higher level intent* stated as rules. An implementation of DS2 language is non existent yet and all we have for it is a subset of its specification, with operational semantics explained in English [5, 4, 3]. The very feature that makes Orleans higher level, is the same that hinders model checking/analysis capabilities. While Orleans runtime abstracts from details of lower level frameworks, which is a feature desired by developers, it also hides these details from analyzing algorithms who rely on them to do their job. These details are available only in the lower level actor frameworks such as Akka [1] for example, what physical actor sent the message, what other physical actor is waiting for a task to complete, is it waiting indefinitely or it would time out, . . . etc. Unfortunately, **It would be less appropriate for DS2 model** to use a higher-level model such as Orleans, since it hides important details needed by the analyzing algorithms. This will be clear in the next subsection/question.

2.3 In what ways might DS2 eventually help in reasoning about Orleans-like frameworks or Orleans-like applications?

It **will help** but before that we need to have some grounds laid out. Let us have some background about what entities are in Orleans applications and how they work in tandem with its runtime. This is to make it easier to understand how DS2 would help in reasoning about Orleans-like applications, in the context of Orleans itself.

Overview of how Orleans works. First, physical servers are represented by "Silos", analogous to Akka's Actor systems [2][1]. This is the runtime, i.e. the container (Orleans runtime) in which virtual/physical actors run, analogous to what Enterprise Java Beans container is to Enterprise Java Beans [2]. Inside these containers, there are two layers: a *physical actors* layer, and a *virtual actors* layer [2]. The physical one contains actual actors whose lifecycle (and other concerns specific to them) is managed by the runtime, i.e. they get creates on demand if they are non existent yet (called activation), and deactivated to free resources when they are not used [2]. The higher abstraction layer is represented by *virtual actors*, called "Grains". These virtual actors are immortal, i.e. they are ever existing and never die or experience a life cycle from an application point of view [2]. Grains communicate only using an **ask** [3] pattern (but in the guise of asynchronous OOP methods), and has only what is equivalent to a **blocking get**, no timed version that may time out like in Akka and DS2 [3][1]. The runtime manages *mapping and routing* between virtual and physical actors by maintaining a distributed map of which virtual actor represents which physical actor, and the where abouts of physical actors [2]. The reason why the runtime needs to know the location for physical actors is two fold: (1) to keep location transparency, and (2) so that if a server containing the physical actor crashes (read physical actor crashes), it recreates it on demand [2]. Although the default task scheduling of Orleans follow FIFO order in processing w.r.t. their scheduling from each actor by default [2]. That can be overridden easily to provide better performance [2]. Grains can be declared as *reentrant*, that is their tasks can be scheduled and processed *out of order*, which brings the potential of data races [2]. A grain may have one task that modifies its local state and another trying to read that very same local state [2], the definition of a race. Also, grains can be declared as *remindable grains*, that can receive scheduled events [2, 8]. This is similar to normal actors timed actions (or scheduled events) [3][1]. Timed events may suffer *time drift*, which may affect the outcome of processing them, leading to some problems such as whether the runtime restarts a crashed physical actor (due to timed event addressed to it) before or after it received an update from another actor (causing data loss) [2]. In addition, there are *stateful grains* and *stateless ones* [2, 8]. Stateless grains are there to be created in more quantities (to a certain limit) to increase *processing* throughput, and hence performance [2]. Stateful ones, however, are there for the rest of the work and they may be involved in data races mentioned earlier [2]. One last bit, there is support for persistence to persist data, but that also is done in different criteria specified by the user: on certain number of updates, periodically, or every single time the state of an actor changes [2]. Worse, if a physical actor that partially processed a task, has done some update to its and/or other actors state, crashes. The systems state is inconsistent (caught between lost updates, stale data, and data corruption) and it may never converge back to a consistent state. This is a major concern why we need to model check/verify distributed systems in the first place. That brings back many distributed systems problems e.g. consistency, eventual consistency, linearizability, deadlocks (due to blocking on a lost task being processed on a crashed actor) and so on.

What can go wrong, where and how. In virtual lands (i.e. Grains), data races may happen when a grain is declared reentrant [2]. In physical lands, many more things can go wrong. In brief, anything that can go wrong in normal actors model, can also go wrong in physical actors backing the runtime of Orleans. Although, the runtime of Orleans tries its best to act as soon as something wrong happens [2]. That is, if a physical actor crashes, then a timed event of the runtime detects it, it brings it back to life by creating it on the same host or on another (depending on whether the host crashed or not and/or utilization level) [2]. That, in turn, triggers an update to propagate and update the distributed map ASAP, so that cached versions of that map (or a subset of it) also get updated. But what happens in between the time that crash and detecting that crash? bad things happen, of course the usual things in lower-level abstractions of distributed systems such as Akka [1], and other Actors frameworks. More over, what about that *distributed* map, that is cached for performance, that keeps being updated on different physical actors? It is a pure distributed system implemented in physical actors, inside the runtime. That makes it vulnerable to any Distributed Hash Table (DHT), key-value stores, problems.

Some issues to be cleared. Our model is developed in Scala while theirs is in C# [2], which makes our framework not applicable unless they port it to C#. After all, the *methodology and the idea* is what matters and implementation details differ according to what language is used.

Mapping from Orleans to DS2 From now on, we will assume that: (1) Our DS2 framework was ported to C#, (2) we have a front end parsing Orleans applications to our model, and (3) An implementation for a scheduler aware of Orleans runtime is made available. This is in addition to including the distributed map the runtime uses to be translated to DS2 model, reflecting the fact that the map is implemented using *physical stateful actors*. So, explanation of how to map an Orleans application to our model follows:

- **Grains** are translated to agents with being marked, say in the local state of the agent, as *stateful* or *stateless* and as *virtual* (to avoid simulating lifecycle on them) or *physical* (to treat them like normal actors). Also, **Grains method calls** on other Grains or physical actors are translated by making the method name a message in DS2 (sent by the **ask** pattern to the callee Grain - soon to be an agent). Further, the payload of the message is set to the formal parameters values (a sequence) of that method call, making sure to treat them as pass *by-value*. Otherwise, an error can be given as a violation of the runtime semantics, since Orleans is a pass by value semantics [2]. Further, any `Task/Task<RsltT>` is replaced by the returned `Future` returned by the **ask** pattern in DS2.
- **Communications** between Grains and Grains are translated to an **ask** pattern in DS2, between Grains and physical actors as an **ask** pattern in DS2, and between physical actors to a normal asynchronous **send** in DS2.
- **The scheduler** is aware of all the specifics of Orleans runtime, so it knows how to squeeze schedules that are fruitful (i.e. find potential bugs) from any Orleans application.

It is clear from this point how things would proceed under the control of DS2 infrastructure (specifically the Orleans-aware scheduler mentioned above) to reveal issues of interest.

2.4 Are there lessons from Orleans that might improve DS2?

For improving DS2 model, no. All required primitives that are available to Actors, on which Orleans runtime is built are also available in DS2 model [2][3]. However, **for improving the concept of *DS2***

***language*, yes** indeed there are lessons learned from Orleans that can improve it significantly [2]. When I first imagined synthesizing distributed systems using the actor model, I knew first hand that actors can be created, destroyed/killed, migrated, and the whole system will be very dynamic and locations of actors may vary during runtime [4, 5]. What didn't cross my mind, however, is the elegant idea behind Orleans *virtual actors* and how it is implemented on top of a distributed map (providing indirection), that is cached for performance to benefit from locality [2]. It maps between virtual and physical actors, and their physical locations [2, 7, 6]. In addition, Orleans ruling out several complexities (mentioned in § 2.1) of actors programming makes it a definite wish for distributed systems developers. It is indeed higher level than programming with bare actors. The **lesson learned** is that synthesis of distributed systems from DS2 language *need not be completely* done by generating code. A better approach for DS2 language can be obtained by accompanying it with an Orleans-like runtime that takes care of so many complexities from both DS2 language synthesis subsystem, as well as its user-specified rules (with minimal performance hit) [2][3]. That, in turn, would make DS2 language a more productive and easier language to approach. Another lesson learned is the distinction between *stateless* actors (i.e. contain no state or read-only state) and *stateful* actors [2]. This distinction is an optimization for both analysis as well as runtime performance benefits. For example, we don't need to check if an update propagates to a stateless actor, for checking consistency for example. Similarly, the runtime need not persist the stateless actors' read-only data since they are given to it, rather than accumulated in it during runtime. At the same time, DS2 language would be even better if it provided guarantees about corner cases of Orleans runtime, those discussed in the previous subsection. That is, if we decided to implement one. There are, however, some lessons learned from Orleans that *raise caution* about performance, depending on some application patterns [2]:

1. Applications mixing *frequent bulk* operations on many entities with operations on individual entities cause degradation in performance. The reason is that isolation of actors makes bulk operations more expensive than on shared memory [2].
2. Due to lack of *temporal locality*, with presence of extremely large numbers of virtual actors (in billions), performance degrades [2].

DS2 language needs to avoid such situations if we chose to implement a runtime similar to Orleans. An additional lesson learned, that *hiding details* of distributed systems realities from developers point of view, while it helps developers live a nice dream, makes it more difficult (if not impossible) for them to reason about things when the underlying layers of a framework go hay-wire, making that dream a nightmare. An *all in one stack* of *reasoning and abstraction* is absolutely needed to bring that awareness and reasoning ability back to developers, while still keeping that level of abstraction. One last lesson, inspired by the concept of *function shipping paradime* described in the paper [2] is that some times data are much bigger to ship over shipping the whole *physical actor* with its state if its stateful (or a new instance of it call it *delegate*) to do the job at the locality of that big data. That is determined dynamically by the runtime, and it achieves better performance than the rigid *how free is a host* Orleans approach to just create physical actors [2]. I would definitely keep this and other lessons in mind when designing the DS2 language.

References

[1] *Akka*. http://akka.io/, Retrieved May 3, 2016.

[2] P. A. Bernstein et al. *Orleans: Distributed Virtual Actors for Programmability and Scalability*. Tech. rep. MSR-TR-2014-41. 2014. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=210931.

[3] *Distributed Systems Abstract Model*. http://formalverification.cs.utah.edu/ds2, Retrieved Jan 31, 2016. 2016.

[4] *DS2 Hardware rules/spec*. http://proof.cs.utah.edu:8080/index.php/Mohammed_ds_hw_rules, Retrieved May 3, 2016.

[5] *DS2 Software rules/spec*. http://proof.cs.utah.edu:8080/index.php/Mohammed_ds_sw_rules, Retrieved May 3, 2016.

[6] *Microsoft Research project Orleans simplify development of scalable cloud services*. https://channel9.msdn.com/Shows/Cloud+Cover/Episode-142-Microsoft-Research-project-Orleans-simplify-development-of-scalable-cloud-services, Retrieved May 3, 2016.

[7] *Orleans - Virtual Actors*. http://research.microsoft.com/en-us/projects/orleans/default.aspx, Retrieved May 3, 2016.

[8] *Project Orleans - Actor Model framework*. http://www.slideshare.net/nmackenzie/project-orleans, Retrieved May 3, 2016.

3 Zvonimir Rakamarić Questions

Read "Asynchronous Programming, Analysis and Testing with State Machines" paper [3].

3.1 Summarize the paper.

Overview. The paper [3] presents a programming language called P#, an extension to C# [2], for *asynchronous* programming, inspired by P [4]. The language has a *static data-race detection*, relying on *syntax and semantics* of the communicating state machines approach it uses. In addition, P# provides infrastructure to facilitate *systematic concurrency testing*. The work continues describing the experience of writing several distributed protocols, including an industrial scale system internal to their organization [3]. The driving factors of this work are: static data race detection, increased reliability and responsiveness of asynchronous programs generated, and better performing systematic testing exploration [3].

Motivation. The abundance in nowadays computational resources, specifically multi-core processors, require programmers to use concurrency and/or parallel programming techniques to utilize [3] said resources. Traditional asynchronous programming models/frameworks, such as multi-threading that relies on locking mechanisms, are prone to concurrency errors such as deadlocks and/or data races [3, 7, 2]. In addition, other event driven frameworks based on the actors model e.g. Scala Actors, and Erlang allow declarative complex asynchronous programming [2]. However, they do not provide the ability to analyze them [2]. The utilization of concurrency lies in breaking long running tasks into smaller and shorter running ones. The sheer amount of interleavings between these shorter tasks, that can both modify and/or read shared state, requires careful fine-grained coordination. This is a main source of concurrency bugs due to the fact that it is hard to achieve correct coordination without automated reasoning, more with informal reasoning. Pin pointing the root causes for such bugs is extremely challenging [7]. The main fruits intended from this work is to address both *correctness* (i.e. high-reliability) , and *responsiveness* of asynchronous and concurrent programming [3].

Solutions proposed by P#. P# tries to solve the said issues with three thrusts. First, it provides a framework for communicating state machines, similar to actors, *first class* citizens of the language [2]. Second, the language compiler is used to exploit the state machine structure, specific to P#, in static analysis in order to detect data races [2, 3]. Third, data race-freedom guarantees established by the static analysis step are further used to enhance performance of the systematic testing [2] step.

The P# language in some detail. Each program is composed of *multiple* communicating state machines [3]. They communicate using one primitive, asynchronous *send*, that sends events to other machines [3]. Each machine has a marked start state, and one of the machines is marked as *main* that starts the whole P# program [2, 3]. Events sent between machines, similar to messages, are handled by actions [3]. These actions are *sequential* C# methods, containing no multi threading primitives e.g. spawning threads, or using any synchronization primitives [3]. When an event that is mapped to an action is received by a state machine, it executes that action [3]. The actions being executed are *asynchronous tasks*, i.e. can run in parallel, and act on a *shared state* contained in the machine that scheduled these tasks [3]. This *shared state*, combined with *asynchrony* of the tasks, and the fact that events payloads are allowed to *reference* objects of the shared state is the potential data race source in P# [3]. P# static analysis leverages the state machine structure that schedule these tasks as well as the *absence* of concurrency within actions to track *ownership* of objects. That is what imposes the coordination [3]. Allowing events to reference shared state objects allows for efficient event-passing and to avoid deep copying/marshaling events and their payloads [3]. There is another catch to events payloads referencing shared state, however. That is, shared state references are only possible on distributed systems running on a single machine (with a shared memory store), rather than on a purely distributed (i.e. networked) system with multiple nodes that DS2 tries to address [5]. The way *ownership* of objects is handled follows: (1) An action assumes ownership of any payload it receives with events in addition to ownership of objects the action creates, and (2) An action gives up ownership of any payload it sends as part of an event [3]. Hence, data race freedom is based on objects having a unique owner at any given time, otherwise a data race is detected [3]. A race-free program still can suffer other concurrency bugs, it is why testing is still needed [3]. P# runtime has an embedded *systematic concurrency testing* (SCT) framework that allows for events to be controllably scheduled and explored [3]. That, in turn, allows for deterministic replay of bugs using tests [3]. The optimization made here is that being done with data-race detection during static time, fine-grained interleaving is left out during the testing phase [3]. Interleavings that are left to be explored in the testing phase are comprised of coarse granularity events interleavings, and this is where the performance benefits come from [3] in comparison with using CHESS alone. Syntax and *formal* semantics of the language are provided in the paper, yet they omit some syntax details for brevity [3].

How to write a P# program I will give a brief description of what needs to be done to implement state machines in a P# program, refer to Figure 1 in the paper [3] for examples. First, a user needs to *inherit* from the base abstract class `Machine`, then implementation is provided inside the inheriting class [3]. Implementation includes the states, which is provided by creating inner classes inheriting from the base abstract class `State` [3]. The wisdom behind inner classes to implement the state machine states is that it prevents these states from being accessed externally [3]. Each state class, has a method called `onEntry` that needs to be overridden by the user to provide the events-handling implementation(s) [3]. Actions are sequential C# methods [3]. One of those actions could simply be sending another event to the same/other machine(s) [3]. A single state *transitions* are defined using special syntax that maps from event-type to a state-type at the state level [3]. For example, `on EventType goto ProcessingState` to encode a transition upon receiving an event type, i.e EventType, that transitions the machine to an end state, e.g. `ProcessingState` in the example. Local state (i.e. attributes) and actions (C# methods) of a state machine in P# are declared in the regular way it is done in C# [3] in the top-most syntactic scope of that machine [3]. If a machine needs to communicate with another machine, the first needs to have a *reference* to the other machine [2, 3]. However, static fields are strictly disallowed inside state machines [3]. Events can be bound to methods to form *action bindings* for a specific state of a specific machine [3]. Start state must be indicated in the machine level (using `start` keyword), the start state `onEntry` method is invoked right after the machine starts [3]. The *main* state machine, that starts the whole system, is indicated with the keyword `main`.

Data race detection. Two transitions in a P# program are said to race if they (1) originate from different machine instances (2) Are not separated by any other transition, and (3) Access the same field with at least one is a write [2, 3]. Direct or indirect references to objects in an event payload is considered for data race detection using the *ownership* analysis, described earlier [3]. Further, any object fields whose ownership was given up by all machines, get given up as well [2, 3]. Then, a *give-up* set [6] of fields is computed per method in all machines, i.e. a set of formal parameters from which the raced-upon object can be reached directly/indirectly [2, 3]. This is referred to as the *gives-up analysis*, formulated as a *fixed point computation* [2, 3] to avoid non-termination on recursive methods. After that, it is enough to *data race freedom* can be established by checking:

1. a formal parameter fp is in the give-up set for method m'.
2. Another method m calls m', setting fp to variable v.
3. After calling m', m accesses a variable v'.

Finally, it suffices to check that there is no object o that can be accessed directly or indirectly from v and v' at the same time [2, 3]. The above is referred to as *respects ownership analysis* [2, 3], which is the algorithm that detects potential data races, including false positives. Finally, *cross-state analysis*, that considers the *transition graph* of state machines, is performed to discard the *majority* of false positives as shown by case studies [2, 3]. The way that is done is simply going through each machine transitions, and checking whether a field (in the give-up set) is *accessed* by a later state after it was given up by a former one [2, 3].

Contributions.

- A new language, P#, co-designed with static data race analysis and testing [3].

[6]Give-up set definition: For each formal parameter *fp* of method *m*, if there exists an object *o* such that if
(i) *o* is reachable from *fp*
(ii) ownership of *o* is transferred by m, then *fp* is in the give-up set.

- Sound, scalable, and precise static analysis for *proving* data race freedom of P# programs [3].
- Systematic and randomized testing strategies that utilize results from static-analysis to exceed the performance of state-of-the-art systematic concurrency testing (SCT) tool, namely CHESS [3].
- Results porting an industrial-scale system, internal to Microsoft [3].

Experiments and Conclusion. [3] The work shows numerous famous benchmarks ported using P# and the performance of static and testing is stated for each. Some of those are Chord, Paxos, MultiPaxos, Raft, the chain-replication protocol, and two-phase commit protocol. They compare precision of their analysis with SOTER (an actor oriented data race checking based on ownership) and mention that theirs outperforms it in precision. Finally, they state some numbers for variations of their scheduler implementation performance, in Table2. Their future target is further improving the analysis to avoid more false positives from static analysis.

3.2 Discuss its pros and cons in detail.

P# has many advantages and some disadvantages:

Advantages

- *Increased productivity* is shown by porting part of azure system fabric, dubbed "AsyncSystem", using 14 P# state machines that generated approximately 6 thousands lines of code [2].
- *Increased readability* of P# programs due to the structuring of its programs in a declarative manner [3], compared to programs written in other concurrency frameworks such as multi-threading.
- *More coverage of concurrency bugs* is exemplified while porting the "AsyncSystem". The analysis and testing infrastructure helped discover multiple bugs that were not discovered before, and were hard to find in the original code of the azure subsystem. One of those bugs was undetected in multiple releases of the same code [2].
- *Extending and integrating with C#* (i.e. P# as an embedded domain specific language in C#) allows for (1) the integration of P# in existing projects written in C#, and (2) the use of *sequential* C# code inside any P# program [2, 3]. In addition, this allows the use of the familiar tooling, IDE's and object-orientation in the design of these state machines, with the restriction that none of the C# code is allowed to spawn threads. All concurrency must be modeled by using P#'s state machines communicating through events [3].
- *Co-design of analysis and program development*, forming a *synergy*, allow for automated analysis and testing on unmodified programs written in P# [2, 3]. This is always a great advantage above model checkers such as Spin [6] that requires manual modeling apart from the implementation. That model, also, may not be a 100% implementation-representative model. This fact is due to the developer intent is to understand their implementation in the first place, so how can they make a representative model port of it at all. It takes several oscillations between model and implementation adjustments to converge on a representative-model, which is by itself a laborious additional effort.

Disadvantages

- *Lack of network distribution*: P# programs are distributed but within one machine. It doesn't address communication between state machines over a faulty network [3]. In purely distributed systems, i.e. involving faulty network and intermittent presence of communicating processes, more careful and more featured design is needed, this will be discussed in § 3.3.
- *Analysis and/or testing* is limited to concurrency stated in P#'s asynchronous state machines. This limitation comes from the fact that the compiler exploits the asynchronous *state machine structure* statically [2, 3], and doesn't address traditional multi-threading primitives such as threads, locks, . . . etc.
- *Forced thinking model on programmers* restricts their ability to use primitives and reasoning techniques they are used to in other concurrency models, e.g. Actor models, multi-threading, . . . etc.

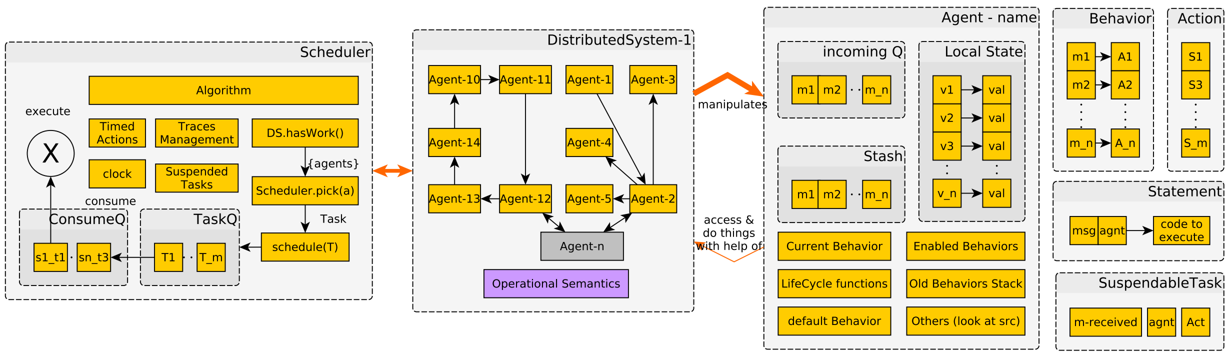


Figure 4: DS2 Architecture: DS2 Runtime data structures and their inter-relationship

3.3 Comment on whether some of the presented ideas could be connected to your own work (and how).

Yes, there are two ideas presented and could be connected to my work. The first one is the systematic concurrency testing and/or exploration, this has been a primary goal for DS2 since its inception. So I will skip commenting on it since how that is done will be apparent from explaining DS2 model (soon), and from Figure 4. The second idea is the data race detection using *ownership* static analysis in P#. I will elaborate on this one more, assuming our DS2 will synthesize code for *some* agents to run on a single machine and then they may pass messages using *shared memory* for efficiency reasons. Another assumption is that messages passed will be mutable between a single-host agents.

DS2 model Before I can relate to our work, I need to summarize the architecture of our model, shown in Figure 4. Our model offers key constructs and then builds upon a simple *strategy OO design pattern*. A Scheduler encapsulates an algorithm (i.e. the *strategy*) to explore any *realistic* distributed system situation, including fine grained interleaving e.g. enabling data race detection on shared state. In addition, we have a Distributed System object (i.e. the *context*) that represents the global state of the distributed system. It is essentially a set of agents, and implementation of our formal operational semantics. Agents are autonomous communicating processes with an incoming queue of messages, a current behavior (i.e. a map from messages received to actions to perform), a local state (i.e. a map from variables to values), and *life cycle* user override-able hooks. Life cycle hooks include `onStart`, `onJoin`, `onStop`, and so on. They are fixed in number, in contrast with P#'s states in a state machine that can grow big with many complex transitions in between states. An agent substitutes a state machine in P# and is started by sending it the `Start` special message. Behaviors are switch-able, stored in "enabled behaviors", and remember-able to backtrack to older behaviors. Behaviors are essentially event-action bindings, like in P# state machines having different event-action bindings [3] based on the current state. There can be only one behavior that is enabled at a time in an agent, called the "current behavior". Actions are, essentially, a sequence of *sequential statements* to execute, with other extra info to support blocking/pre-emption and such. Once an agent is started, the scheduler starts to *schedule* actions (from current behavior) matched using messages popped from its queue, setting that action formal parameter to the message that invoked it. After that, the action (called task now) is enqueued to the scheduler task queue. A scheduler can decide to schedule more tasks from other agents, *consume a statement* from already scheduled tasks, or to literally do anything else e.g. dropping or re-ordering messages in an agent's queue. The order at which statements are consumed from one task or another determines the *interleaving*. Then, executing the statements one by one from the front of the scheduler's consume queue performing then removing them. Note that a scheduler need not execute anything, statements have enough meta data to do targeted exploration without execution. In addition to that, a scheduler may *choose to* execute statements to check their effect on the local state of the agent that scheduled them, then back tracks to a whole runtime state at well; thanks to DS2's snapshot-resume feature.

Key differences DS2 and P# models have key differences, these need to be cleared before I explain how to apply P# data race analysis. P# assumes state machines can share data and claim ownership, so that the static analysis guided by the strict state machine semantics can decide whether an object shared is uniquely owned by a single state machine [3]. Our model, based on Akka actors [1], strictly prohibits sharing of any local state to other agents and the constraint is that payload of messages sent are treated as immutable. However, the constraint is not enforced in purpose so that data race checking and sharing of data between agents can still be achieved. The reason behind the constraint in DS2 is that we need to address distributed systems communicating through possibly faulty networks and with intermittently present processes, which is not addressed by P# [3]. Sending data over a network, needs (de)marshaling/(ds)serialization using *deep copying*, and passing references in payloads to *share* local data violates our model semantics. DS2 also has the same asynchronous **send** like in P#, as well as a more advanced **ask** pattern, we will focus on *send* only. Last, an action's parameters in DS2 are the message received and the agent whose local state is on which the action acts. Formal parameters of P#'s actions are formal parameters of the methods while in DS2 the *payload* of the message are the formal parameters on which *ownership* analyses is done. The payload of the message in DS2 is variable length sequence of values/references analyzed versus the local state content of the agents that sent and those receiving them.

How can we achieve static data race analysis in our work There are two ways to achieve data race detection presented in P#. Schedulers are aware of our strict semantics of agent's, just like the case in P#'s state machines. It is important to mention that statements have thorough meta data starting from static time, i.e. the construction of the distributed system model out of an implementation. Meta data has information about the following: the kind of a statement (send, modify state, ..etc), the formal parameters and their values of that statement, and if values are determined during runtime it also knows where those values are stored, what the effect of executing that statement would be even without executing it (with the exception for control-flow statements conditions that may need some more calculation be determined during runtime). The behaviors, also, provide additional information of what message received can cause exactly what in the receiver agent. Actually, we can know when exactly an agent would change behaviors from static time from statements meta data. So, one option is to completely implement the three analyses P# performs right inside a scheduler that *statically* checks *meta data* in statements (along with the structure of an agent) and builds both the the *give-up set* and the *control flow graph* (CFG). Later, it performs *respects ownership* analysis in the same way it was explained in P#, but on information now present inside a scheduler as a give-up set and CFG. The extension implemented in P# to reduce false positives can also be performed by the same, or different scheduler passing the computed info in the first scheduler. Other approaches may choose to implement everything in a front end parsing an implementation of Akka actors [1] (our primary target), or split the analyses between a scheduler and a front end. Those who plan on parsing implementations written in Akka [1], like our case, can add parts of the analyses in the front end. These approaches are possible however the designer of the analysis algorithms chooses to. However, to keep things modular and cleanly separated, the first choice I explained is considered the best.

References

[1] *Akka*. `http://akka.io/`, Retrieved May 3, 2016.

[2] *Asynchronous Programming, Analysis and Testing with State Machines - slides*. `http://www.slideshare.net/akisdeligia/psharp-pldi`, Retrieved May 2, 2016. 2016.

[3] P. Deligiannis et al. "Asynchronous Programming, Analysis and Testing with State Machines". In: *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation*. PLDI 2015. Portland, OR, USA: ACM, 2015, pp. 154–164. ISBN: 978-1-4503-3468-6. DOI: `10.1145/2737924.2737996`. URL: `http://doi.acm.org/10.1145/2737924.2737996`.

[4] A. Desai et al. "P: safe asynchronous event-driven programming". In: *PLDI*. ACM, 2013, pp. 321–332.

[5] *Distributed Systems Abstract Model*. `http://formalverification.cs.utah.edu/ds2`, Retrieved Jan 31, 2016. 2016.

[6] G. J. Holzmann. "Logic Verification of ANSI-C Code with SPIN". In: *SPIN*. Vol. 1885. Lecture Notes in Computer Science. Springer, 2000, pp. 131–147.

[7] *P# - Asynchronous Programming, Analysis and Testing with State Machines*. `https://www.youtube.com/watch?v=jj0wUM4Rl-w`, Retrieved May 2, 2016. 2016.

4 Matthew Flatt Questions

See Marketplace [6]. Marketplace builds on Erlang-style actors with additional language constructs for managing subsets of actors.

4.1 In what ways do the goals of Marketplace overlap with the goals of DS2 (for example, possibly, turning programming patterns into language constructs)?

They overlap in several ways [3, 5, 4] [6, 7]. A list of what overlaps follow:

1. The use of Actors model [1] as a corner stone for constructing distributed system that strictly do not share data except through communication (using messages and/or events).
2. Structuring distributed systems into sub/nested ones. While DS2 takes the notion of a distributed system with multiple behaviors based on roles they want to perform in a distributed system, the network calculus organizes groups and layers of those communicating processes and what roles they want to participate in.
3. The use of subsets/patterns of actors to address, and to manage delivery and reaction to messages/events to for groups of these actors using a broadcast semantics.
4. Indeed as mentioned in the question, DS2 tries to force the developer into providing strategies (patterns) in order to synthesize code to take care of data consistency and/or coordination forced by that pattern (e.g. a primary backup strategy, paxos to converge to a common state,...etc). Market place uses the patterns *publish/subscribe*, *broadcast*, and *elevated roles* (through the addition of more responsibilities to actors the as they participate in more/higher layers).
5. Both try to raise the abstraction for developers to be more productive and to enable easier formal and informal reasoning about their systems.
6. Both have semantics similar to that of publish/subscribe semantics. For example when DS2 needs to address a subset of communicating and known to exist agents, it addresses them in either a regular-expression to match a set or directly provide a set of agents and/or predicates in order to model some kind of behavior and/or delivery pattern [3, 5], e.g. who is interested in receiving which messages, who would react to which messages and in what occasion/role/behavior, and scoping the messages to affect which level/parts of distributed system, . . . etc. Marketplace [6] has an intuitive approach that is implementable by all languages and have basic support for few requirements, and paves the way to add support for pattern matching on events, messages, and subscription status and level.
7. DS2 tries to synthesize code for taking care of coordination and consistency between actors/agents, marketplace has that by-design especially for coordination and membership management (presence and absence of actors in a certain layer/role) [6, 7].

4.2 Would it make sense to extend DS2 to support Marketplace-like subsets of actors?

Yes, it makes sense. An explanation of all details will be presented and discussed in the next section, § 4.3. Implementing the network semantics in DS2 runtime would definitely simplify the process of synthesizing the remaining code for data consistency patterns/algorithms. This is because the Network Semantics (NC)[6, 7] takes care of the portion of the code responsible for managing membership and presence of processes and crash/disconnect fault tolerance, and provides a higher level *route-by-content* compared to *route-by-address* semantics [6, 7]. That, in turn, removes a whole layer of complexity from the synthesis and filters a lot of fault sources from our model checking efforts. Knowing about Marketplace, and its network calculus is definitely a *pleasant surprise*.

4.3 If so, what would that extension look like?

To answer the question, I will need to first explain our model, how things fit together and a subset of DS2 syntax. After that, we show the changes to the syntax to support the added network semantics [6, 7].

Overview. We have *DS2 model* which is the generic distributed systems analysis infrastructure that is based on Akka [2]. In addition, we have *DS2 language* which is the higher level language for synthesizing distributed systems, from *higher level intent* stated as rules. An implementation of DS2 language is non existent yet and all we have for it is a subset of its specification, with operational semantics explained in English [5, 4, 3]. The extension can be done starting with the model then going up to the language. I will sketch how I would change the model. The flow of the answer will be to: present our *reduced* model details and its two communication primitives, how the extension to it would be, then will show *some* of the language details (syntax, English semantics,...etc), and finally how I would change the language (an example). Note that when we synthesize the final distributed system, it will be in Akka, but first we need to populate the model before we can come up with Akka implementation (because we need to do further analysis). I had to reduce the model to keep the page limit and not complicate the discussion in relation to Marketplace's network calculus(NC) [6]. There is a more detailed model in the technical reports section of DS2 website [3].

DS2 model details. **Distributed System** is a tuple $\langle \mathcal{A} \rangle$ with the set of agents in a distributed system $\mathcal{A}$. This is analogous to a configuration in Network Semantics [6]. We will first elaborate the primitives in our model, and then return to the components of the distributed system (i.e. the agents).
Message is a tuple $\langle s, p_0, \ldots, p_n \rangle$ where s is the sender agent, and $p_0, \ldots, p_n$ is the payload of the message. The set of messages is symbolized by $\mathcal{M}$.
Action is essentially the sequence of statements to execute. It has reference to which message invoked it, and which destination agent of that message, to act on that agent's local state if needed.
Timed Action is an action associated with two values of time. It is the tuple $\langle t1, t2, \gamma \rangle$ where the action γ is to execute every $t1$ time with tolerance of $t2$ amount of time, usually used to model *heart beat*. The tolerance time $t2$ is there because it cannot be predicted how long a message in flight is delayed, and hence it allows some delay-tolerance.
Behavior its a tuple $\langle nm, r \rangle$ stating a map from message to actions $r : \mathcal{M} \rightarrow \Gamma$, modeling *reactions* to messages received. nm is the name of the behavior. Just like Agha's actor model [1] behaviors are swappable using become(behavior-name). The state, however, does not change with swapping behaviors. Behaviors know how to act on an actor's state (now called agent local state) once activated since they have a reference to it.
Agent is a communicating autonomous process. It is a tuple $\langle q, \mathcal{R}, \tau, \mathcal{L}, \mathcal{B} \rangle$ where q is its receive queue, $\mathcal{R}$ is the reactions (the primary behavior), timed actions set $\tau = \{\langle t1, t2, \gamma \rangle\}$, $\mathcal{L}$ is the local state of an agent $\mathcal{L} : Ids \rightarrow Vals$. $\mathcal{B}$ is the enabled behaviors set (analogous to subscriptions in NC [6, 7], which indicates other responsibilities assigned to this agent over its primary responsibilities in $\mathcal{R}$). We called it agent, instead of actor, to distinguish between the model and implementation.

Communication primitives and supported operations. While our model has two communication primitives, **ask** (which returns a handle/future to block on) and **send**. I will focus the discussion on the **send** primitive. In short, **send** is a point-to-point asynchronous message send, from source agent (set in the message s field) to destination agent incoming messages queue q. In order to elevate our model to that of NC, we will assume that we provide a **broadcast** primitive. Its signature would be $broadcast(\alpha_{src} \in \mathcal{A}, m \in \mathcal{M}, b, l)$ where:

- α_{src} is the broadcasting agent.
- m is the message to be broadcasted
- b is the behaviors name intended to process this message.
- l is the behavior level, at which the sender (indicated inside the message s field) is operating.

We also have an agent creation primitive and it is almost identical to the one in NC [6, 7]. It creates an agent, assigns it responsibilities inherited from its creator agent, and its queue is empty, then it is added to the agents of the same layer/distributed-system [7], and finally the created agent announces its existence (part of its init operation). No layer can create an agent at another layer except after: (1) Changing its current reaction to be of that layer (2) It creates that agent (3) Finally, the Creator or created agent can move to any layer accordingly by changing its primary behavior [8] to match the destination layer behavior.

[7] I call a nested distributed system a layer or vice versa
[8] Behavior of a layer, means the service that layer provides

How everything fits together. Initially, all agents queues are empty. When an agent is started, at least one will receive the special *Start* message, and hence the distributed system is started. The agent, then, de-queues that message, searches its reactions map for the action to execute (i.e. $\mathcal{R}(msgRecvd)$, and executes it. Note that an action executed can do anything from sending messages, creating other agents, changing the agent's state, ...etc. This is exactly the same as in NC [6] except that if an agent is created, it inherits everything from its creator (the parent agent) without the queue content and without local state. Although, no parenthood is here like in Akka supervision [2]. This is done by all agents till they either exhaust all their queues, and then there is nothing is left, or they keep doing this indefinitely. Meanwhile, timed actions of each agent keep firing every $t1$, we can safely ignore $t2$ in this discussion.

DS2 model modifications. From the model, we can see that the distributed system configuration (call it C) lacks the subscriptions status. If we are to extend it so it can handle the NC semantics [6], we would have extended it in the following way: **Distributed System** will be $\langle \mathcal{A}, d, name \rangle$ where:

- $\mathcal{A}$ the set of agents comprising this distributed system/layer/service.
- d is the distributed system nesting level at which this distributed system resides i.e. 0 if this is the root distributed system/layer, 1 if this is nested inside the root, 2 if it is nested inside the nested inside the root; and so on. Multiple distributed systems can be nested inside any one.
- *name* the name of the service this layer provides, must be a a unique behavior name. It specifies that all agents part of this layer or moving into it, must make their primary behavior $\mathcal{R}$ to be that behavior. Also, all subscribers must have a behavior in their enabled set $\mathcal{B}$ that matches it.

Note that there is no other root in a distributed system, it has to be only one root. Otherwise distributed systems all at the root level will be, again, reduced to a "soup of actors". **Agent** will be extended to have an additional cached routing table ρ, updates it on every message received.. It is a mutable map $\rho : \mathbb{N} \times Ids \rightarrow Agent^*$, stating which agents are still alive in which level. That is, it stores information about what agent/actor is operating at which (level,behavior-name) pair. **Behavior** will be extended to be $\langle nm, r, o \rangle$ where every thing is the same as before, except the o which means the observation level. Observation level encodes *when* to use (or ignore) this behavior on a received message from certain layer. Its a predicate over positive integers. Basically, if it matches the level (i.e. $l \in o$) then it has to be used, otherwise the behavior is not considered to handle the message. **Messages** will be extended to $\langle s, l, d, p_0, \ldots, p_n \rangle$ where everything stays the same except for:

- l indicates the level from which the sender sent (read broadcast from now on) this message.
- d indicates the *observation* level at which this should be handled. A predicate over positive integers.

In addition, we will have two more primitives that an agent executes if it wants to move across layers of a distributed system. We have **ascend** which calls **become** to replace the primary behavior of that agent $\mathcal{R}$, to be of that destination layer's behavior , indicated by *name*. After an agent ascends, it notifies everyone (using **broadcast**) to update their routing table. The reverse of that operation is *descend* and it does the opposite job of *ascend*. However, the notification message is sent with d equal or less ($d \leq l_{old}$) to that of the old layer, so that agents of that layer and lower layers are notified to update their routing table. Notification messages for routing information are handled in all behaviors (a synthesized entry in that behavior). When an agent in a certain layer/distributed-system broadcasts a message, the message fields are set (i.e. s to the sender agent, l to the level at which that agent is operating, and d to indicate the level at which this message is to be handled). In the opposite direction, an agent that received that message, would first update its routing table accordingly, handle the message, and then update the same message fields: s will be updated to the receiver, l to the level at which it is operating when it received+processed the message, and d will be updated to hold the older value of l. Then it will broadcast it back as an acknowledgment if and only if the behavior observation level o corresponds to the old l. Otherwise, i.e. if $o \cap l = \epsilon$ (older l), then the message is ignored (i.e. dropped). One last important thing, now each distributed system is represented by a set of Agents, that keep track of agents in the same layer or lower(i.e. their nested) layers (these are the natives to that layer which are *publishers*). Agents on *nested* layers are interested in getting messages from and sending acknowledgments to *enclosing/containing* layers' agents. Any agent can create more agents or ascend/descend to/from layers (if coded in the behavior corresponding to that layer). When an agent ascends to a layer, it broadcasts a notification to all agents in that layer downward (covering all sub tree).

When it descends, it does the same but also notifies the layer from which it descended from. Due to that, notified agents will adjust their routing table ρ, which keeps track of each layer's subscriptions in that subtree[9]. In other words, an agent routing table ρ keeps track of the subtree of nested distributed systems it belongs to, including its own layer, and higher layers up to the root layer. This is maybe not the most efficient implementation of the network semantics, but it does the job.

Some DS2 language details and example syntax with semantics. In this part, I will try to give a quick example that explains the syntax and semantics of a subset DS2 language. The reason behind this is to show how things used to be (before I know about the networks semantics) and next I show how I modify them.

```
machine: ds1, 1555  // communicating on port 1555
//the old style indicating which distributed system
//this one is nested in, left out when this one is root
world: world_name // Each label (e.g. election) states a behavior
agents: (6, computation), (1,master), (3, replication), (2, election)
// SAYHELLO is a message with payload "Mo", computation behavior
reactions[computation]:(SAYHELLO(name = "Mo"),{print("Hi",name)}, ...
// regular scala function calling out rule "send"
def ack(m:Message,a: Agent) = {m.s = this; m.l = locals(level) ; send[a][m]}
reactions[master]:(SUBSCRIBE, {updateSubscriptions(); ack(sender)} ), ...
...
machine // end of a distributed system declaration
// another distributed system nested in ds1 this one
machine: ds2, 1556
// lives inside ds1's world
world: ds1
// one computation node, and 2 replication
agents: (1, computation), (2, replication)
reactions[computation]:(SAYHELLO(name = "Mo"),{print("whatever")}, ...
...
machine // end of a distributed system declaration
```

Listing 1: example DS2 distributed system with old syntax (pre NC [6])

Listing 1 shows a simple example in which a distributed system is declared and another one nested inside of it. The listing shows the old-style DS2 before added network semantics [6]. The `world` declaration for `ds1` is totally ignored since there is no machine called `world_name`, in which `ds1` can be nested in. Hence, it becomes the *root* distributed system in which all other collaborating systems should be nested. If there is another distributed system declared at the same level of `ds2`, then they both are considered to have the same level (layer). Except that by declaring two different distributed systems, we can scope certain behaviors, for example by having a mutually exclusive set of behaviors with `ds2` but overlapping behaviors with say `ds1` or other distributed systems.

```
machine: ds1, 1555  // communicating on port 1555
//the old style indicating which distributed system
//this one is nested in, left out when this one is root
world: world_name // Each label (e.g. election) states a behavior
agents: (6, computation), (1,master), (3, replication), (2, election)
// SAYHELLO is a message with payload "Mo", computation behavior
// observation level 1 means observe communaction from the scope
// of level 1, even from other neighboring dustributed systems
reactions[computation][1]:(SAYHELLO(name = "Mo"),{print("Hi,"+name)}, ...
// regular scala function calling out rule "broadcast"
// "master" says all master-layer is notified (instead of just a single
```

[9] A subtree of a layer means: that layer, and all its nested layers, down to the leaf layers

```
// agent at a time or specifying a pattern of those agents-addresses)
// Also, note the '_' will lead to all layers-masters to try to handle the
// message. If we wanted a layer and lower ones to handle it, we would specify
// that layer depth 'd' and message will carry that info to destination agents
    ↪ ,
// and if they have "master" behavior with same observation level, they would
// handle it, and acknowledge it too.
def ack(m:Message,a: Agent) = {m.sender = this; m.level = this.level ;
    ↪ broadcast[master][m][_]}
reactions[master][_]:(SUBSCRIBE, {updateSubscriptions(); ack(sender)} ), ...
...
machine // end of a distributed system declaration
// another distributed system nested in ds1 this one
machine: ds2, 1556
// lives inside ds1's world
world: ds1
// one computation node, and 2 replication
agents: (1, computation), (2, replication)
// handle whatever received from level 3 and down to 0, no negative levels
reactions[computation][_<=3]:(SAYHELLO(name = "Mo"),{print("whatever")}, ...
...
machine // end of a distributed system declaration
```

Listing 2: example DS2 distributed system with new syntax (post NC [6])

Listing 2 shows the same example after adding the networks semantics. The comments explain the semantics in normal English. We still don't have formal semantics for the language part of DS2 nor the complete syntax ready, yet. This is why the scarcity of examples. Also, since the model is largely reduced, I had to ignore all those rules that refer to other details of the extended model e.g. on-join, on-rejoin, on-demise, ...etc [5]. Notice that the change is as the following:

- *send* rules were replaced with *broadcast* rules.
- *reactions* rules were amended to have some more parameters to synthesize the behavior and the level at which it should observe.
- Also, I amended the language syntax to have two more rules, namely *ascend* and *descend*, they don't take parameters as they are executed inside an agent/actor and hence will affect that agent/actor.

Note that extensions to specifying even a *pattern* and *subsets* of behavior names on reactions is also supported.

Concluding remarks. DS2 will ultimately generate code based on Akka-provided facilities such as *routers* and *routees* [2], we can model publish/subscribe with an additional number of actors. Using an elastic cluster of actors would enable DS2 to provide more reliable, and higher throughput *service* to subscribers actors. Given the flexibility of both the network calculus meta-model, our model, DS2 language syntax, and Akka [2] implementation of the actor model, all changes are feasible without major changes.

References

[1] G. Agha. "Actors: A Model of Concurrent Computation in Distributed Systems". PhD thesis. MIT, 1985.

[2] *Akka*. http://akka.io/, Retrieved May 3, 2016.

[3] *Distributed Systems Abstract Model*. http://formalverification.cs.utah.edu/ds2, Retrieved Jan 31, 2016. 2016.

[4] *DS2 Hardware rules/spec*. http://proof.cs.utah.edu:8080/index.php/Mohammed_ds_hw_rules, Retrieved May 3, 2016.

[5] *DS2 Software rules/spec*. http://proof.cs.utah.edu:8080/index.php/Mohammed_ds_sw_rules, Retrieved May 3, 2016.

[6] T. Garnock-Jones, S. Tobin-Hochstadt, and M. Felleisen. "The Network as a Language Construct". In: *ESOP*. Vol. 8410. Lecture Notes in Computer Science. Springer, 2014, pp. 473–492.

[7] *The Network as a Language Construct*. http://www.ccs.neu.edu/home/tonyg/esop2014/, Retrieved May 2, 2016.

NAME: Wei-Fan Chiang

1 Brief Summary of the Paper: "Operator Dependant Compensated Algorithms"

The authors explored how *fused multiply and add* operator (FMA) impacts the performance and the numerical precision on compensated algorithms. A compensated algorithm performs the same task as a certain numerical algorithm or operation. The compensated version has lower performance but higher numerical precision as if using 2X bit-width floating-point numbers in the original algorithm/operation. The authors' contributions in this paper are concluded as follows:

- By using FMA, the authors propose a new compensated algorithm for floating-point multiplication. Compare to the classic compensated multiplication (Veltkamp and Dekker [1]) which requires 17 floating-point operations, the authors' algorithm requires only 2. In this paper, the authors didn't show the comparison results between the classic and the new compensated multiplication. However, on performance, the new algorithm will be theoretically faster. On precision, the new compensated multiplication provides almost the same precision as the case of using 2X bit-width floating-point numbers.
- The authors compare the performance and precision impacts of using FMA and compensated algorithms. The authors apply FMA operation in two benchmarks: Horner algorithm and dot product. FMA operation provides higher performance (A FMA fuses two operations into one.) However, compare to compensated algorithms, FMA doesn't shows improvement on precision in practice. In fact, compensated algorithm show higher precision. (This comparison is based on using the same floating-point bit-width.)
- The authors implemented two compensated versions of Horner algorithm and dot product. (Each benchmark has two different implementations.) The two versions handle the case "multiply-then-add" differently. One version uses compensated FMA. The another version uses a combination of compensated multiplication and compensated addition. The authors found that using FMA yields lower performance. However, it doesn't provide higher precision than the combination operation version. The next section contains more details of this comparison.

2 Details of Compensated Algorithms

2.1 Error-free Transformations of Compensated Algorithms/Operations

Error-free Transformation (EFT) is a property of compensated algorithms/operations. A compensated algorithm/operation computes an EFT if it takes floating-point numbers as arguments and produces a list of floating-point numbers as output. The **exact** summation of the output numbers is the result calculated in exact arithmetic. Also, the first number in the list is the result calculated in finite precision arithmetic. For example, let CSUM(a,b) denotes the Knuth's compensated addition [3] which computes an EFT for addition. CSUM(a,b) returns two numbers, [x, y], which satisfy the following condition:

$$(x = a \oplus b) \wedge (a + b = x + y)$$

($\oplus$ is the addition operation in floating-point arithmetic. '+' is the exact addition operation.)

	CSUM(a,b)	CMUL(a,b)	CFMA(a,b,c)
# of floating-point operations	6	2	17

Table 1: The Number of Floating-point Operations Required by the Three Compensated Operations

The authors' compensated multiplication, CMUL(a,b), computes an EFT for multiplication.

$$[x, y] = \text{CMUL(a,b)}\ s.t.$$
$$(x = a \otimes b) \wedge (x + y = a \times b)$$

The authors' approach of using FMA to implement CMUL(a,b) is shown as follows:

```
[x,y] = CMUL(a,b) {
    $x = a \otimes b$
    $y$ = FMA(a,b,-x)  }
```

(The function, FMA(a,b,c), denotes the "hardware FMA.")

The compensated FMA [2] used in the paper, CFMA(a,b,c), computes an EFT for FMA.

$$[x, y, z] = \text{CFMA(a,b,c)}\ s.t.$$
$$(x = \text{FMA(a,b,c)}) \wedge ((x + y + z) = (a \times b + c))$$

Table 1 shows the number of operations used in the above three compensated operations.

2.2 Compensated Horner Algorithm

The Horner algorithm evaluates a polynomial (with floating-point coefficients) at a floating-point value. Let p is a polynomial: $\sum_{i=0}^{n} a_i * x^i$. The following shows the Horner algorithm evaluates a polynomial at a floating-point value x:

```
$r$ = Horner(p,x) {
    $r = a_n$
    for i = n-1 to 0
        $r = (r \otimes x) \oplus a_i$    // A FMA can replace here. }
```

The authors firstly construct EFT for the Horner algorithm. They tried two possible EFTs: compensated FMA version and compensated mul+add version. The compensated FMA version is shown as follows:

```
$[r, p_1, p_2]$ = HornerEFT_FMA(p,x) {
    $r = a_n$
    for i = n-1 to 0 {
        $[r, \alpha, \beta]$ = CFMA(r,x,$a_i$)
        Set $\alpha$ to $p_1$'s i$^{th}$ coefficient.
        Set $\beta$ to $p_2$'s i$^{th}$ coefficient. }}
```

2

The compensated mul+add version is shown as follows:

```
$[r, p_1, p_2]$ = HornerEFT_mul_add(p,x) {
    $r = a_n$
    for i = n-1 to 0 {
        $[\gamma, \alpha]$ = CMUL(r,x)
        $[r, \beta]$ = CSUM($\gamma$,$a_i$)
        Set $\alpha$ to $p_1$'s i$^{th}$ coefficient.
        Set $\beta$ to $p_2$'s i$^{th}$ coefficient. }}
```

Note that both EFTs of Horner algorithm return a floating-point number (r) followed with two polynomials (p_1 and p_2). These returned values are used in calculating the final answer. The compensated Horner algorithm, CompHorner(p,x), calculates the final answer as shown follows:

$$\text{CompHorner(p,x)} = r \oplus \text{Horner}(p_1\text{,x}) \oplus \text{Horner}(p_2\text{,x}) = r + \text{Horner}(p_1 \oplus p_2\text{,x})$$

$(p_1 \oplus p_2)$ is a polynomial.

In this paper, the authors compare the performance and precision between HornerEFT_FMA and HornerEFT_mul_add. They found that HornerEFT_FMA doesn't show higher precision than HornerEFT_mul_add. However, its performance is lower than HornerEFT_mul_add. (Table 1 implies the reason.)

2.3 Compensated Dot Product

The authors also implemented two versions of compensated dot product: by using CFMA and CMUL+CSUM. Similar to the case in Horner algorithm, the CFMA version doesn't show higher precision but has slower performance.

3 Using Compensated Algorithm in Auto-tuning

The authors compared the performance between the 64-bit compensated Horner algorithm and the double-double (128-bit) non-compensated version. The two versions show the similar precision but the 64-bit compensated version has higher performance. Based on this result, it seems that programmers could improve performance but still keep the same precision by using compensated algorithms. However, double-double (128-bit) floating-point arithmetic should has very slow performance. It is because double-double operations are not supported by hardware. On the other hand, 64-bit arithmetic is supported. Therefore, I created my own Horner algorithm and dot product benchmarks. In this section, I report the performance and precision on three versions of both benchmarks. The three versions are described as follows:

- *(horner/dot)-fma.32* is a 32-bit version of using FMA but not using compensated algorithms.
- *(horner/dot)-comp.32* is a 32-bit version of using compensated algorithms. For "multiply-then-add" situation, it uses combination of compensated multiplication and compensated addition.
- *(horner/dot)-fma.64* is a 64-bit version of using FMA but not using compensated algorithms.
- The exact arithmetic is approximated by the 128-bit versions.

The choice of these three benchmarks is based on a potential scenario of program tuning: A programmer has an implementation on 64-bit precision. However, using 64-bit floating-point numbers may be expensive on power consumption or performance. Thus the programmer considers using 32-bit floating-point number. He/She also considers using compensated algorithm to improve precision.

Some details of my benchmark implementation and setting are shown as follows:

- I currently can not find out a way to force using hardware FMA. Therefore, to approximate the precision provided by FMA, I used 128-bit floating-point arithmetic. For a 32-bit FMA_{32}, my implementation is shown as follows:

$$\text{FMA}_{32}(a, b, c) = (float)((\text{\_\_float128 } a) \oplus (\text{\_\_float128 } b) \oplus (\text{\_\_float128 } c))$$

  Hardware FMA operation calculates the multiplication and addition in exact arithmetic, then rounds only once to get the final answer. The exact arithmetic in FMA is approached by 128-bit arithmetic. The rounding is approached by type casting from 128-bit to 32-bit floating-point number.
- Each of my Horner algorithm benchmark evaluates a fixed size polynomial with 1001 random coefficients at a random value. Each random value is in $[-1.0, 1.0]$. This is different from the authors' setting on Horner algorithm. The authors evaluated the polynomial $(x - 1)^n$ with fixed x to 1.333. They changed n to test performance and precision on different polynomials.
- Each of my dot product benchmark generates two random fixed size vectors (1000 elements each) and computes the dot product of them. Each element is in $[-1.0, 1.0]$.
- The performance score of each benchmark is measured in its elapse time (10^5 runs) divided by the elapse time of *(horner/dot)-fma.64*. For example, suppose the time of 10^5 runs of *horner-comp.32* is 15 seconds and the time of *horner-fma.64* is 10 seconds, the performance score of *horner-comp.32* is 1.5. Thus, the higher the performance score, the lower the performance.
- The precision is measured in relative error with 10^{-8} as "padding." The authors also calculate relative error as the precision metric. However, they didn't use padding. Instead, they cut all errors greater than one to one.

3.1 Experimental Results and Discussions

Table 2 shows the performance and precision results of Horner algorithm benchmarks. Table 3 shows the performance and precision results of dot product benchmarks. From these two tables, we can observe the following things:

- The performance of the three versions of both benchmarks are very close. The experiments are run on a multi-core machine (multiple Intel Xeon CPUs). It could provide good hardware support on 64-bit floating-point arithmetic. Of course, the performance results could be changed if real hardware FMA applied.
- I measured the precision by two methods: unguided random testing (URT) and binary guided random testing (BGRT). On both benchmarks, they all report *(horner/dot)-fma.64* as the most precise version. The 32-bit compensated versions are less precise than the 64-bit versions. However, they are more precise than the 32-bit versions.
- One interesting thing in our results is that the precision of the compensated versions seems to be "stable!" By changing the precision measurement algorithm from URT to BGRT, both the 32-bit and 64-bit non-compensated versions show higher errors. By using BGRT, the highest

		horner-fma.32	horner-comp.32	horner-fma.64
Exp1	performance	1.0014	1.0818	1
	precision (URT)	1.7056e-03	5.9422e-08	4.7789e-13
	precision (BGRT)	7.3591e-01	5.9449e-08	3.7831e-09
Exp2	performance	1.0008	1.0777	1
	precision (URT)	4.3984e-04	5.9446e-08	5.5271e-13
	precision (BGRT)	1.1143e+00	5.9550e-08	3.8291e-08
Exp3	performance	0.9952	1.0751	1
	precision (URT)	5.8907e-04	5.9516e-08	5.1078e-12
	precision (BGRT)	2.4850e+00	5.9418e-08	5.4220e-09
Exp4	performance	0.9915	1.0839	1
	precision (URT)	9.4963e-04	5.9524e-08	1.8255e-12
	precision (BGRT)	3.1392e+00	5.9301e-08	9.3075e-09

Table 2: Performance and Precision Results of Horner Algorithm

		dot-fma.32	dot-comp.32	dot-fma.64
Exp1	performance	1.0280	0.9919	1
	precision (URT)	2.5636e-02	5.9368e-08	3.5511e-11
	precision (BGRT)	4.3908e+02	5.9560e-08	1.6403e-08
Exp2	performance	1.0322	1.0035	1
	precision (URT)	2.7600e-02	5.9454e-08	3.3922e-11
	precision (BGRT)	3.4483e+01	5.9524e-08	5.5738e-08
Exp3	performance	1.0290	0.9988	1
	precision (URT)	5.4019e-02	5.9455e-08	1.9420e-11
	precision (BGRT)	8.2904e+01	5.9539e-08	4.9305e-07
Exp4	performance	1.0182	0.9925	1
	precision (URT)	9.5832e-01	1.8044e-07	9.6020e-10
	precision (BGRT)	8.6204e+01	5.9567e-08	4.3663e-10

Table 3: Performance and Precision Results of Dot Product

errors of 64-bit non-compensated version are reported to be close to 32-bit compensated version. On the other hand, the precision results reported by both URT and BGRT don't show too much difference. Our results could give an empirical conclusion: it is possible that compensated algorithms could eliminate many "corner inputs" that cause extraordinary high errors.

Therefore, it is worthy to investigate more on compensated algorithms to check whether they provide "stability" to numerical computations. If it is true, it could be a main motivation of using compensated algorithm in auto-tuning.

References

[1] Dekker, Theodorus Jozef. "A floating-point technique for extending the available precision." Numerische Mathematik 18.3 (1971): 224-242.

[2] Boldo, Sylvie, and J-M. Muller. "Some functions computable with a fused-mac." Computer Arithmetic, 2005.

[3] Knuth, Donald E. "The Art of Computer Programming, Vol. 2: Seminumerical Algorithms" Addison-Wesley, Reading, MA, USA, third edition, (1998)

NAME: Wei-Fan Chiang

1 Summary of "Autotuning Multigrid with PetaBricks"

This paper proposes an auto-tuning method that synthesizes multigrid programs with considering both performance and accuracy. A multigrid program (e.g. a 2D Poisson's equation solver) could be a combination of multiple underlying algorithms such as sequential, iterative, and recursive algorithms. The program recursively divides an input data into partitions, and invokes different underlying algorithms to deal with the partitions. Such algorithm selection is based on some parameters such as partition size. Since the underlying algorithms provide different performance and accuracy on different parameters, synthesizing the optimal program is searching the optimal cutoffs (thresholds of switching underlying algorithms). Previous work only consider performance as the only metric in tuning (or synthesizing) such combinative algorithms. However, this paper consider both performance and accuracy as metrics.

The methodology proposed in this paper is a dynamic programming based synthesis. This approach is based on the following assumption: finding the "optimal" (combinative) algorithms for solving smaller-size problem is independent from finding the optimal algorithms for larger-size problem. Thus, the method starts from finding the optimal algorithms under small input sizes. (An algorithm C_1 is optimal if the method doesn't find another algorithm C_2 such that C_2 dominates C_1 on both performance and accuracy.) With synthesis results of small input sizes, the method proceeds to search the optimal algorithms under a larger input size. When synthesizing a candidate algorithm which recursively calls other algorithms to handle data partitions, the method tries out all optimal algorithms previous found.

While increasing the input size, the number of optimal algorithms enumerated by the method could be dramatically increased. The paper proposes two approaches to prune candidates. The first approach is dividing the whole accuracy scale into (disjointed) ranges. For example, suppose the accuracy scale is from 0 to 100 ($[0, 100)$), it can be divided into four ranges: $[0, 25)$, $[25, 50)$, $[50, 75)$, and $[75, 100)$. For all optimal algorithms whose accuracies fall in the same range, only the one with the highest performance will be chosen to synthesize other algorithms. The second approach is to restrict the scales of some tuning (synthesizing) parameters. For example, from several (underlying) iterative algorithms, the authors selected only one of them (SOR: Red-Black Successive Over Relaxation) in the synthesis process. The number of iterations performed by the iterative algorithm (SOR) can also be fixed to a constant. Such restrictions on tuning parameters could be given by (offline) experimental results.

The authors implemented the auto-tuner (synthesizer) for a parallel programming language, PetaBricks. To write a PetaBricks program, programmers need to explicitly specify *transforms* and *rules*. Each *transform* converts an input or an intermediate result to an output or another intermediate result. The whole program is a series of *transforms* from an input to an output. Each *transform* contains several *rules* which specify different implementations of data processing. PetaBricks compiler can reason the dependencies among *transforms* and generate code in two modes. The first mode is opening the *rule* choices (algorithm choices) and parameters to external configuration files. The second mode is to hard-wire a configuration into the code that improves the performance. The algorithm synthesis method is starting from small input size (as described in previous paragraphs) and seeded with single-algorithm implementations (applying only one underlying algorithm to whole computation). The generated program will has the highest performance among optimals which meet the accuracy requirement.

The experimental results show that the methodology proposed in this paper could successfully generates programs (of a 2D Poisson's equation solver) that have higher performance than single-algorithm implementations (with satisfying the accuracy requirement). Compare to two other reference tuning methods, the generated programs also have higher performance.

2 Summary of "Language and Compiler Support for Auto-Tuning Variable-Accuracy Algorithms"

This paper proposes a genetic algorithm for automatic tuning (synthesizing) of variable-accuracy algorithms. The paper defines "variable-accuracy" algorithms as those who could earn performance by paying some degree of accuracy as price. Such algorithms could be approximation algorithms for NP-hard problems or iterative algorithms which iteratively compute results until *convergence criteria* met. Multigrid programs, which recursively employ underlying algorithms for computing partial results, are also variable-accuracy algorithms. To trade performance with accuracy (or vice versa), a variable-accuracy algorithm could has many parameters for tuning performance/accuracy, such as the number of iterations or algorithm choice of handling sub-problems. These parameters build an "abstraction boundary" between the library writers and the library users. Library writers, who design the variable-accuracy algorithms, may expose many parameters to library users for handling various accuracy requirements. However, library users may not know the impacts on accuracy made by the parameters. The genetic tuning algorithm is designed to bring down the "abstraction boundary." The authors implemented the genetic tuning/synthesis algorithm in a parallel programming language compiler (PetaBricks compiler) with support of language extensions (in PetaBricks). The proposed tuning algorithm can help library writers find the optimal configurations that achieve high performance under specific accuracy requirements. For library users, the automatic tuning can hide the complexity of setting parameters.

For language extensions, the authors extended PetaBricks (described in the previous section) to allow more flexibility in tuning/synthesis. In this paper, four major extensions are proposed. The first extension is allowing programmer-specified accuracy metric. PetaBricks programmers can write their *transforms* that calculate accuracy scores based on inputs and outputs. The second extension is supporting programmer-specified parameters in programs. The tuning method will automatically find the values of these parameters in optimal programs. The third extension is providing a loop-statement whose number of iterations is dynamically decided during the tuning process. The forth extension is a runtime accuracy check statement that allows programmers to specify locations of accuracy check. This also allows programmers to specify handlers for the case that accuracy check fails.

The genetic tuning/synthesizing method proposed in this paper is implemented on top of PetaBricks compiler. Each tuned program is mapped to a *choice configuration file* created by PetaBricks compiler. The file contains all parameter values including programmer-defined variables and decision tree. A decision tree records algorithm choice on each choice site (for handling sub-problems). During the tuning process, each configuration (an algorithm/program) is randomly chosen as a "parent" to generate "children configurations." This generation of children configurations is done by a mutator function. The mutator function randomly changes parameters of the parent configuration based on their categories. For decision tree, the mutator could randomly remove or insert a algorithm choice cite with a cutoff value to recursively handle sub-problems. For large scale parameters like cutoff value, their values are sampled from a log-normal distribution. The intuition of using log-normal distribution is that it enlarges the impacts of small changes on small values than large values. For small scale parameters like algorithm choice, their values are sampled from a normal distribution. The whole genetic tuning method keeps a set of optimal algorithms (a population) and performs a two-level loop: the outer and the inner loop. The outer loop controls the input size. The method starts from generating optimal algorithms under small input sizes. The outer loop increases input size on every iteration. Before entering the inner loop, all algorithms in the population are tested and preserved if it is optimal. The inner loop repeats random mutations and creates new algorithms to the population. To prevent search space explosion, each inner loop iteration preserves the best K (a user-specified parameter) algorithm in the population.

The genetic tuning method requires comparison among algorithms. The comparison is done by providing trained inputs. The number of inputs to provide is dynamically decided by the tuning method. The tuning method uses a probability metric to indicate that the two algorithms are the same or not. Once the two algorithms are indicated to be distinguishable, not more inputs will be provided for comparison. Otherwise, more inputs will be provided until a threshold (a user-specified parameter) reached.

To evaluate the genetic tuning method, the authors generated a set of benchmarks. Their experimental results show that the tuning method can generate high performance programs for all benchmarks which also meet various accuracy requirements.

3 Discussions

3.1 Potential Impact on Accuracy and The Needs of Improved Accuracy Check

The two papers propose techniques that assist programmers to sacrifice accuracy for performance improvement. Both papers use "concrete testing" to measure accuracy. I refer "concrete testing" as providing a set of concrete inputs to the program and observe on outputs. However, they don't describe the source of "trained" inputs. (The term "trained" is used in the papers. I assume that the "trained" inputs are actually random or arbitrarily specified.) Therefore, it is possible that the tuned programs meet accuracy requirements in the synthesis but (frequently) fail in practice.

My current work, described in our latest paper, is a search based methodology for finding inputs that cause high numerical errors. Therefore, *by using our method of input generation, programs generate by the two techniques may be more accurate.* Here are points to clarify for the this statement:

- A program is more accurate means it is less likely to fail the accuracy requirements in practice.
- The generated programs may have lower performance.
- The terms "accuracy" and "numerical precision" don't have the same meaning here. The "accuracy" means the "distance" between the (current) computed solution and the optimal solution. Such distance could be known by computing residual. The accuracy could be decided by the algorithm of approaching the optimal solution and the number of iterations we take in the algorithm. On the other hand, the "numerical precision" means the difference between the result calculated in floating-point arithmetic and the result calculated in real number arithmetic. The cause of the numerical imprecision is mainly from rounding operations during floating-point arithmetic. Numerical precision could be improved by using higher bit-width

floating-point numbers or applying compensation methods (like Kahan summation or Knuth summation).

However, I still believe that our search based input generation can help improve accuracy. It is because our method finds high numerical errors by exploring "input distributions." An input distribution depicts how input values scatter in the input domain. We found that many inputs of causing high/low numerical errors could possibly have similar distributions. I believe (but not formally or empirically proved yet) that many inputs which cause low accuracy could also have similar distributions. It is because our method has shown effectiveness on finding high numerical error with different metrics.

3.2 Potential Work Needed for Applying Search Based Input Generation

My proposed approach of integrating our input generation method to the two tuning methods is that, for each candidate algorithm, we search sufficient amount of inputs and provide the top K (user-specified) inputs needed in the tuning. Apparently, there are some performance issues to solve. First, the amount of inputs we explored could be much more than the amount needed by the tuning methods. Second, we may need to re-generate a whole new set of inputs for each candidate algorithm found by the tuning methods. It is because different algorithms may require different input distributions to trigger low accuracy. My proposed solutions for the performance issues are listed as follows:

- We need to "reuse" the generated inputs. First of all, we need to invent a mapping methodology that links an algorithm, with its input size, to a set of inputs. An input must be allowed to appear in other algorithms' input sets or be part of another input.
- In a variable-accuracy algorithm, an input could be divided into partitions (disjointed sections) for computing partial results. Therefore, we can firstly generate inputs for algorithms which take small input sizes. For large inputs, we could generate them by aggregating small inputs. However, it is possible that computations on partial results could also affect accuracy. Consequently, inputs generated from aggregations may not effectively trigger high inaccuracy after several levels of aggregations. Therefore, for algorithms taking large input sizes, we can set a probability to generate inputs from search.
- Storing actual inputs could require lots of space. An idea to "compact" inputs could be storing their "distributions." A distribution can be stored as a conjunction of constraints. A distribution represents a set of inputs. For example, consider two inputs I_1 and I_2:

  $$I_1 = \{v_1 = 1.0,\ v_2 = -0.8,\ v_3 = 0.499\} \quad I_2 = \{v_1 = 0.9,\ v_2 = -0.7,\ v_3 = 0.509\}$$

  We could conclude the set of inputs, $\{I_1,\ I_2\}$, has distribution

  $$(v_1 > 0) \wedge (v_2 < 0) \wedge (v_3 - 0.5 \leq 0.01)$$

  We may borrow the template based approach of (loop) invariant synthesis to discover distributions. Once we store distributions as input sets, we need to re-sample inputs from the distributions. Naive sampling from distributions is a try-and-error approach. (Randomly enumerate inputs and prune invalids.) Efficient sampling from distributions is to be investigated.
- Static analysis could also help reduce the space needed for input reusing here. For a program/algorithm, we could analyze the *transform* which specifies the accuracy metric. (The

second paper allows programmer-specified accuracy metrics.) We can identify the specific subset of input variables that affect the accuracy calculation. This can be done by programing slicing or information flow analysis. Then only those identified input variables need to be store. Other variables can have random values.

1 Central Contribution of the Paper "A Sound Floating-point Polyhedra Abstract Domain

The paper proposes a sound floating-point implementation of polyhedra abstract domain, an effective numerical abstract domain. This could improve the performance of abstract interpretation for numerical programs. It is because previous implementations of polyhedra domain use rational numbers to represent numerical values and involve rational number arithmetic which greatly degrade analysis performance. The authors improve performance by taking the advantage of (relatively) faster floating-point arithmetic. However, due to rounding operations, naively replacing rational numbers with floating-point numbers may cause unsound analysis results. (For a numerical variable, its polyhedra domain is sound if the domain contains all the variables' plausible values.) To keep soundness, the authors propose a sound floating-point Fourier-Motzkin elimination and employ rigorous linear programming (LP). These two techniques are the core subroutines of polyhedra domain operations. The key ideas of preserving soundness in these two techniques are described in the following two paragraphs.

Fourier-Motzkin elimination is mostly used in removing (or havoc) a variable in a domain. When coefficients of the input inequality system of Fourier-Motzkin elimination are floating-point numbers, to keep soundness, the coefficients in the result system are represented by intervals. (Inequalities with interval coefficients are called linear interval form.) An interval coefficient preserves all possible values with considering rounding uncertainty during the computation. For each inequality in linear interval form, the authors employed a linearization technique to find a linear non-interval inequality to over-approximate it. Thus, the result inequality system of floating-point Fourier-Motzkin elimination can be composed by all linearizations of linear interval inequalities.

Linear programming is mostly used in entailment check. It checks whether a domain is contained by another domain. Entailment check is required by operations like redundancy removal, emptiness test, and widening. To keep soundness, the rigorous LP considers the worst-case rounding error and finds the lowest/highest plausible bound when the objective function targets to minimum/maximum. Consequently, rigorous LP provides conservative answers to entailment checks. For example, considering two domains P_1 and P_2, a rigorous LP answers positive to $P_1 \models P_2$ means positive. A rigorous LP answers negative means "not sure" (or "maybe"). Such conservative behavior can over-approximate polyhedra domain operations. (e.g. Fewer inequalities will be chosen in the widening operations.)

To improve analysis precision, the authors also propose some tightening techniques. They are briefly described as follows. Bounds of variables/coefficients can be tightened by LP and bound propagation. Widening operation can be tightened by envelop. In linearization (for over-approximating linear interval inequalities), coefficients can be biased to integers. In the case of sacrificing soundness, rigorous LP can be replaced by standard LP solver.

2 Potential Sources of Analysis Imprecision

2.1 General Cases

As the authors mentioned in the paper, there are two general sources of imprecision. The first source is in linearization. In the inequality shown in Definition 2,

$$\zeta(\varphi, X) =^{def} \sum_k d_k \times x_k \leq c \ \oplus_{+\infty} \bigoplus_k {}_{+\infty}(max\{b_k \ominus_{+\infty} d_k,\ d_k \ominus_{+\infty} a_k\} \otimes_{+\infty} \mid x_k \mid)$$

$\mid x_k \mid$ is decided by the bounding box of (vector) x. As shown in *Example 1* in the paper, when x_k in the bounding box is $[-10, 5]$, $\mid x_k \mid$ is 10 which is a source of over-approximation. (The right-hand size of the above inequality will be enlarged.) Consequently, an inequality which contains more values will be introduced. The second source is in rigorous LP. In the case of finding the minimum value of an objective function, rigorous LP returns a conservative result which is lower than the real minimum. Therefore, the results of some entailment checks, which should be positive, become "not sure." Consequently, as mentioned in the previous section, the polyhedra domains generated by some operations will contain fewer inequalities, which means more values.

2.2 The Pathological Case in Linearization

There is a pathological case in floating-point Fourier-Motzkin elimination (in the linearization phase). This pathological case only happens in some specific environments.

The main reason of having this pathological case is that floating-point summation can be calculated differently by applying different associativities. Consider a mini floating-point format which allow only four digits in significant. The following shows an example of adding four mini floating-point numbers (A, B, C, and D). The floating-point numbers are shown in binary.

$$A = 1.0001 * 2^{-4} \quad B = 1.0010 * 2^{-4} \quad C = 1.0011 * 2^{-2} \quad D = 1.0100 * 2^{3}$$

$$\begin{array}{lcl} Exact & : & A + B + C + D = 1.01001101111 * 2^3 \\ Associativity1 & : & ((A \oplus_{+\infty} B) \oplus_{+\infty} C) \oplus_{+\infty} D = 1.0101 * 2^3 \\ Associativity2 & : & ((D \oplus_{+\infty} A) \oplus_{+\infty} B) \oplus_{+\infty} C = 1.0111 * 2^3 \end{array}$$

We can observe the two associativities shown above yield to different results.

Now, again, consider the inequality shown in Definition 2, we name its right-hand side as f:

$$f = c \ \oplus_{+\infty} \bigoplus_k {}_{+\infty} \ (max\{b_k \ominus_{+\infty} d_k,\ d_k \ominus_{+\infty} a_k\} \otimes_{+\infty} \mid x_k \mid)$$

We use m_k as a short name of a part of f:

$$m_k = (max\{b_k \ominus_{+\infty} d_k,\ d_k \ominus_{+\infty} a_k\} \otimes_{+\infty} \mid x_k \mid)$$

Therefore, f could be written as follows: (with an explicit specification on associativity)

$$f = c \ \oplus_{+\infty} \ ((\ldots(((m_1 \oplus_{+\infty} m_2) \oplus_{+\infty} m_3) \oplus_{+\infty} m_4) \oplus_{+\infty} \ldots m_k)$$

The scale of f decides the precision of the original inequality. Fortunately, f is expected to be small wrt all feasible associativities of adding c and all m_ks. The intuition is shown as follows:

- Consider Equation 3 in the paper, a_k of Definition 2 is $(a_k^+ \oslash_{-\infty} a_i^+) \oplus_{-\infty} (a_k^- \oslash_{-\infty} a_i^-)$, and b_k of Definition 2 is $(a_k^+ \oslash_{+\infty} a_i^+) \oplus_{+\infty} (a_k^- \oslash_{+\infty} a_i^-)$. Therefore, in Definition 2, a_k is close to b_k. Thus, if $\mid x_k \mid$ is not too large, we can expect a small m_k.
- If c is sufficiently larger than $(\bigoplus_{k\ +\infty} m_k)$, f could be very close to the exact result.
- An example can be shown by the "Associativity 1" in the second paragraph of this section. Suppose we have three m_k terms: m_1, m_2, and m_3. Consider

$$f = c \oplus_{+\infty} ((m_1 \oplus_{+\infty} m_2) \oplus_{+\infty} m_3)$$

let $c = D$, $m_1 = A$, $m_2 = B$, and $m_3 = C$, f is very close to the exact result.

However, if m_k is large (because of large $\mid x_k \mid$) or c is close to $(\bigoplus_{k\ +\infty} m_k)$, the inequality could dramatically lose precision. An example can be shown by the "Associativity 2" in the second paragraph. We can observe the different by mapping $c = C$, $m_1 = D$, $m_2 = A$, and $m_3 = B$. Now f, $c \oplus_{+\infty} ((m_1 \oplus_{+\infty} m_2) \oplus_{+\infty} m_3)$, is far from the exact result. Even though the difference between the two associativities is just 2 units in the last place $(10111 - 10101)$, the exponent part (2^3) could magnify the difference.

In other words, the inequality in Definition 2 does not address this pathological case and seem to use a deterministic approach to calculate its right-hand side. This could result in a pathological case of high precision loss in analysis. On the other hand, there also exists a pathological case (associativity) that causes low precision loss.

Note that the summary is calculated in "rounding to infinite" mode. So every feasible associativity is guaranteed to compute the summation greater than the exact result, which preserves soundness.

Qualifying Exam

Question 1

Pascal Grosset

University of Utah

1 In your research statement, you discuss several alternative solutions for compositing algorithms for volume rendering of combustion codes using AMR (radix-k, binary swap, direct send). An underlying issue is how the approach will evolve as socket-level architectures scale to large numbers of cores. One way to evaluate different implementation strategies is to develop performance models of the alternatives, and then do some validation on the target architectures to verify the models.

For this question, you are to design performance models that capture the relevant characteristics of two very different compositing algorithms (e.g., direct send and radix-k).

The two algorithms I will compare are the ones suggested: Direct Send and Radix-k.

Image compositing usually has 2 stages: compositing and gathering. In the compositing stage, each process starts with one part of the final image that it has been generated though volume rendering or a similar rendering process. The processes then exchange sections of the image they have until at the end of the stage, each process is authoritative on one section of the final image. Then follows a gathering stage where each process send their section to one process which puts the sections it receives in the correct position in the final image. The gathering stage is the same for both algorithms.

1.1 (a) For each, summarize the algorithm, and describe the amount of computation and communication relative to input data set (0.5 page each).

1. Direct Send

There are two versions of direct send. In the first version, all the processes send their image to the display process [4] and the latter assembles them to produce the final image on its own. There is no gathering stage in this version. In the second, more popular version, each process is responsible for one section of the final image. If there are n processes in the system, each process is responsible for $(1/n)^{th}$ of the final image. All processes send the sections that they are not responsible for to the relevant node. Figure 1 shows an example of that. Then there is a gathering stage where one process receives all the image sections and puts together the complete image. I will consider the second version as this is the one that is most often cited [1],[6].

In terms of communication, for a system with n processes, the amount of messages that needs to be sent/received per process is $n-1$ (the -1 is because a process does not send to itself). Let the final image have p pixels. The number of pixels per message sent is p/n and the total number of pixels that a process will send/receive:

$$(n-1)\frac{p}{n} \tag{1}$$

In terms of computation, a process will need to blend each pixel from $n-1$ images it received. So the number of blending operations that it needs to perform is:

$$(n-1)\frac{p}{n} \tag{2}$$

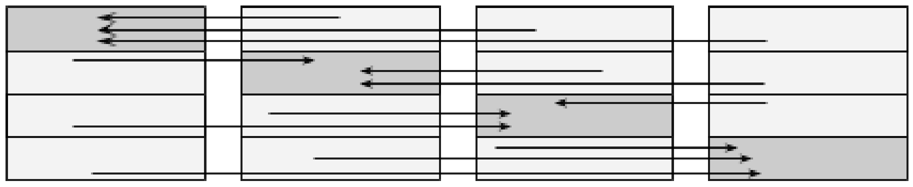


Fig. 1. Each square represents a process and the dark gray rectangle show the section that each process is responsible for. The arrow shows the movement of data

2. Radix-k

This is a more complex algorithm. Each process is still responsible for one section of the image but the way the exchanges take place that is different. The number of processes n is factored in r factors so that k is a vector where

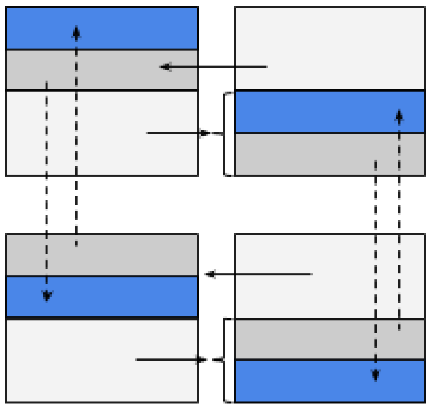


Fig. 2. The gray rectangle shows the section that each process is responsible for in the first round and the blue rectangle show each section that a process is responsible for in the second round. In this example, r=2, k=[2,2].

$k = [k_1, k_2, ..., k_r]$. For example if $n = 80$ and $r = 3$, $k = [5, 4, 4]$. The processes are arranged into groups of size k_i and exchange information. At the end of a round, each process is authoritative on one section of the image in its group. In the next round, all the the processes with the same authoritative partition are arranged in groups of size k_{i+1} and exchange information. This goes on for r rounds until each process is the only one authoritative on one section of the image. Figure 2 shows an example of how this works. These sections are then gathered to generate the final image. Radix-k is described in detail in [5].

In terms of communication, there are r rounds and in each round, the number of messages a node receives in a specific round i is $k_i - 1$. So the total number of messages sent is $\sum_{i=1}^{r} k_i - 1$. The size of the messages decreases per round. If the final image has p pixels, in the first round the size of messages exchanged is p/k_1. Then it's $(p/k_1)/k_2$ and so on. In the i^{th} turn, the message size is $\prod_{j=1}^{i} p/k_j$. So the total number of pixels sent/received per process is:

$$\sum_{i=1}^{r}[(k_i - 1)(\prod_{j=1}^{i} p/k_j)] \tag{3}$$

In terms of computation, for each round i the size to blend is $\prod_{j=1}^{i} p/k_j$ which is done $k_i - 1$ times. For r rounds, the total amount of blending is:

$$\sum_{i=1}^{r}[(k_i - 1)(\prod_{j=1}^{i} p/k_j)] \tag{4}$$

For both radix-k and direct send, the number of pixels sent is the number of pixels that need to be composited which is why for both algorithm, the communication and computing is same in terms of pixels. At the end of the day, both algorithms have to send the same number of pixels but radix-k does that in fewer messages. The gathering stage is the same for both algorithms; they need to send their section to the process that will output the final image.

In terms of communication, each process sends 1 message which has p/n pixels. The total number of messages that the final process will receive is $n-1$ and so the total size of message in the system is:

$$(n-1)(\frac{p}{n}) \tag{5}$$

For computation, only the output process has work to do. The work it does is copying each section of the image it receives to the right place. I am using n instead of $n-1$ this time as the section of the image that the output process has might need to be placed in the correct location too.

$$n(\frac{p}{n}) \tag{6}$$

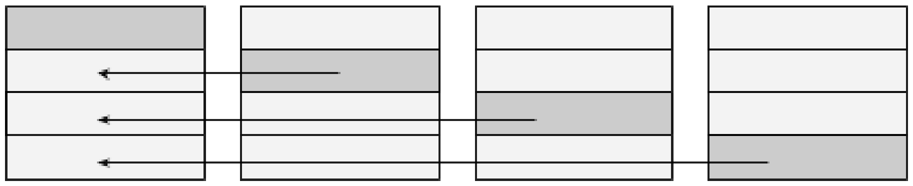


Fig. 3. All the processes send the section they are authoritative on to the display process.

1.2 (b) Describe how the implementation might vary by architecture and scale at the socket level (0.5 page each): (i) conventional multi-core cluster; (ii) GPU cluster; (iii) many-core cluster of Xeon Phis

(i) Conventional multi-core cluster

For both of these algorithms, it is possible to run these algorithms on a per-node basis where a process is mapped to one node or a per-core basis (where one node has several cores) where each process is mapped to one core. Assuming modern Intel CPUs are available on the cluster, the implementation will use vectorization leveraging the SIMD architecture and threads to use all the cores and exploit hyper threading if the latter is available. With vectorization, on a Sandy Bridge architecture for example, this means that 16 single floating point operations can be done per core in parallel. Also when doing compositing for volume rendering, the order in which to blend the images is crucial. If the images do not arrive in the correct order to be blended, they will have to be buffered while waiting for processing.

For a per core implementation where each process is mapped to one core, there will be a lot of messages in the system and this can very quickly become the bottleneck. Cores on the same node will communicate using MPI which is slower than using shared memory. Threads will only be available if hyper-threading is enabled though vectorization will be available.

For a per node implementation, we have one MPI process per node and each node can use threads and vectorization to maximize the use of all the cores. Binding the threads to the cores will also probably be advantageous. In terms of MPI communication, there will be less messages in the whole system. Blending will generally be faster than on a per core basis.

For Direct Send, on a per core implementation, there will be a lot more messages in the system. Scaling up the job will massively increase the number of messages in the system as even processes on the same node will communicate through MPI instead of using shared memory. With more messages, the order in which the messages will arrive will vary a lot and some cores might have to wait for a long time to be able to get the image sections they need to blend them. On a per node basis, there will be fewer messages in the whole system. Communication will be less of a bottleneck. Also, blending will most likely be faster as threads (with vectorization) can be used which will share the same shared memory. The biggest gain here will come from having less messages in the whole system and so adding more process will scale better.

For Radix-k, on a per core basis, we will have more round than on a per node basis. Scaling will increase the number of messages in the both cases but the number of messages will be less in a per core than a per node. Since the number of messages is fewer in radix-k than direct send, radix-k will scale better.

Studies [3] have also confirmed that operating at the node level is better than operating at the level of a core.

(ii) GPU Cluster

I will assume that GPU cluster has one GPU per node. The CPU will handle mostly communication while the computation will be taken care of by the GPU. On receiving an image to be composited with the one it has, the CPU will send that image to GPU where it will be converted into a texture so that it can be easily blended with OpenGL operations. Using a texture and OpenGL might be preferable to things like CUDA or OpenCL in this case as OpenGL has inbuilt functions for doing bending. So in this case, the implementation for both algorithms will require code for the CPU and code for the GPU as well. Moreover, now there will need to be information transfer from the system memory to graphics card memory.

For Direct Send, each process will operate at the node level. Each node will take care of 1 portion of the image and will receive that section from each other node in the system. That portion of the image can reside on the GPU for the whole duration of the compositing operation. The image received will have to be sent over the bus to the graphics card which might be costly. Scaling will increase the number of messages in the system and with direct send using all-to-all communication, performance will suffer.

For Radix-k, unlike for direct send, there is not a part of the image that can reside on the GPU throughout the duration on the operation. For example if we have 4 processes and they are grouped into groups of size 2, in the first round, $p/2$ pixels will be sent while in the second round $p/4$ pixels will have to be sent where p is the total number of pixels in the whole image. This means that for each round, we will need to retrieve from the GPU memory some part of the image. This will take some time. Scaling up the number of processes will increase the number of messages in the system but also the number of transfer from the GPU to the CPU and this will have a negative impact on the performance.

(iii) many-core cluster of Xeon Phis

The xeon phi currently act like a coprocessor attached to a node. There are usually three modes of execution possible on the xeon phi clusters: Offload, Native and Symmetric. In the Offload mode, the MPI is on the cpu of the node and work is offloaded to the xeon phi card. In the native mode, the Xeon Phi has an MPI rank and can execute and communicate with the other xeon phis and finally in the symmetric mode, both the xeon phi and the node's cpu has an MPI rank and talk. Each MPI rank will start with an image which will contribute to the final image.

In the offload mode, the xeon phi acts like a GPU. All communication must go through the node's CPU. Openmp and vectorization can be used to accelerate the blending operation though directives that will offload that computation to the Phi. The implementation will have to change a bit to accommodate for that. For both direct send and radix-k, the number of processes will be one per node of the cluster. The CPU in both cases can be used to buffer messages that will be coming out of order and once two images next to each other in the composting order are available, the CPU will send them to the xeon phi to be blended. Radix-k will have the same issues as using a GPU where each time part of the image will have to be retrieved over the bus after a round to be sent to the other nodes in the next round. For direct send, a part of the image can just sit on the xeon phi for the whole duration of the compositing operation. Scaling will have the same issues as with a GPU.

In the native mode, the CPU of the node is not used. The xeon phi card will handle all the communication as well as the processing. This is like the per node approach in (i). However, there is usually no direct bus from the xeon phi to the outside world, communication will still have to go through the CPU and this might be slower than using the CPU directly if the latter is for example an intel sandy bridge. This is also because computation in compositing is usually super fast. Scaling up will involve having more messages in the system and will also imply having more communication to and from the xeon phi. That might have slowdown the system.

In the symmetric mode both the cpu node and the xeon phi card have are MPI processes and each will start with a part of the image. However the xeon phi is usually much more powerful than the cpu on the node but on the other hand, the xeon phi will pay a bigger price for communication. So load imbalances can easily happen here and can be unpredictable. Scaling up will increase the number of messages and for the xeon phis, the number of communication along the bus to and from the cards. Also the implementation might have to be different to suit the xeon phi and the cpu as they may have different different register size and so memory alignment criteria would be different.

1.3 (c) Present a performance model that will predict the performance for each of your two algorithms. Consider that load imbalance must somehow be captured in the model. (1.5 pages each)

The first section of the answer describes the time that must be spent for tasks common to both direct send and radix-k. In the second part, I will apply the model to diect send and radix-k.

As mentioned before, compositing takes place in two stages: compositing and gathering. I'll start with compositing. Compositing can be broken down into 3 components: the time spent computing, time spent communicating and time spent idle. Thus the total time for each process can be expressed as:

$$T_{Compositing} = T_{Computing} + T_{Communicating} + T_{Idle}$$

where:
- computing: is time spent blending pixels of two images.
- communicating: is time spent sending and receiving pixels,
- idle: is time spent waiting for images arrive,

For the communicating time, this is normally made up of two further components, an initial startup/latency + time to transfer one byte multiplied by the length of the message L in bytes.

$$T_{Communicating} = T_{lag} + T_{comm-per-byte} * L$$

For both algorithms, I will assume a hybrid implementation where nodes communicate through MPI and cores inside a node share global memory. This is the predominant approach now as it has been shown to be more efficient by Howison et al. [3].

For computing, most of the time is spent blending pixels together. The front-to-back blending equation that we commonly use to assemble images is:

$$C_i = (1 - A_{i-1})C_i + C_{i-1} \tag{7}$$

where C_i and A_i are the accumulated opacities.

So the amount of computation required per channel is 1 subtraction $(1 - A_{i-1})$, 1 multiplication $(previous * C_i\ )$, 1 addition $(previous + C_{i-1})$ and 1 assignment. The $(1 - A_{i-1})$ is common across all channels and is maybe optimized. So it's either 16 operations or 13 operations probably depending if the compiler can optimize it or not: an intel compiler probably will but a blend operation on the graphics card by OpenGL might not. So the computing time per pixel is the time to spent to perform these 16 or 13 operations and the overall computing time for an image is the number of pixels p multiplied by the blend time where blend-time is either time for 16 or 13 operations. $T_{time-blend}$ should also take into account the time for memory fetches as well.

$$T_{time-blend} = time\,to\,execute\,16\,ops$$

$$T_{Computing} = T_{time-blend} * p$$

I have grouped the operations together because some architecture like the SIMD can perform several operations at the same time. The Intel Sandy Bridge can do 16 single floating point operations in parallel or 8 double operations in parallel.

The idle time is when a process is not communicating and not doing any computation. The usually happens when the process is waiting for data to come to resume computation which is because the right information is not there yet for computation to start. In compositing, the order of blending matters. If a process is waiting for let's say 3 images, and it currently has images 1 and 3 but not 2, it cannot do any computation and has to wait. This is usually the result of the messages arriving out of order because it took more time to compute the image for 2 than for 3 (a classic example of load imbalance) or it's the network chose to send image 3 before image 2. Also, direct send uses all-to-all communication while Radix-k uses point to point communication. When there are many processes in the system, it is conceivable that the all-to-all communication will slow down much more than the point-to-point communication.

Quantifying load imbalance is hard as it depends on the order in which messages will be received and on the dataset, the view position and transfer function; a small tweak in the transfer function could make a whole region of the dataset blank and so while some processes will have a lot to compute, others will have nothing.

$$T_{Idle} = T_{Compositing} - (T_{Computing} + T_{Communicating})$$

where T_{Idle} has two components:

$$T_{Idle} = T_{Idle-comm} + T_{Idle-comp}$$

where $T_{Idle-comm}$ would represent the time idle because of messages not arriving in the right order and $T_{Idle-comp}$ would represent the time idle because of the dataset and viewing parameters.

The best way to get an estimate for T_{Idle} would be to run the compositing algorithms a few times simulating some cases like translucent dataset, completely full dataset, sparse image and record some timings for those. Then we should be able to express the idle time as a fraction x of the total time.

$$T_{Idle} = x * (T_{Computing} + T_{Communicating})$$

Compositing papers that try to come up with a performance model like [2] often ignore the idle time as it is hard to model.

Once compositing has been done by each process, the display process needs to gather the images from each process. This would be an MPI-Gather from all the processes. So it's $n-1$ (assuming the display process is one of the compositing nodes) messages of size p/n each for a total of $(n-1) * \frac{p}{n}$ pixels sent over the network and the display node would have to integrate $(n-1) * \frac{p}{n}$ messages.

$$T_{Gather} = T_{Gather-Communicating} + T_{Integrate-pixels}$$

$$T_{Gather-Communicating} = (n-1) * (T_{lag} + T_{comm-per-byte} * L)$$

$$T_{IntegratingPixels} = n * (T_{copying-pixel} * p)$$

I am also assuming that though the display process has its section of the pixel it will still need to move it to the right location for display. So all the pixels in the final images p have to be moved. The $T_{copying-pixel}$ is made up of time it takes to do one assignment operation.

$$T_{Total} = T_{Compositing} + T_{Gather}$$

This is the same for both direct send and radix k. Also there is hardly any lag in the gather process as as soon as an image is received it can be processed. Any lag here would be because some processes are not done yet with their compositing work.

Each pixel has 4 channels (red, green, blue & alpha). Floating point is the datatype commonly used to store each channel but this can vary. To make it more general, let's assume that the data type used is of size b bytes. I will now apply this model to direct send and radix-k using the equations in part (a).

1. Direct Send

We have previous computed the computing and communication terms in terms of pixels for Direct Send in equations 1 and 2 from section (a).

Compositing:

$$T_{Compositing} = T_{Computing} + T_{Communicating} + T_{Idle}$$

$$T_{Communicating} = (n-1) * T_{lag} + T_{comm-per-byte} * [(n-1)\frac{p}{n} * 4 * b]$$

$$T_{Computing} = T_{time-blend} * [(n-1)\frac{p}{n}]$$

$$T_{Idle} = x * (T_{Computing} + T_{Communicating})$$

2. Radix-k

For Radix-k, we will use the equations 3 and 4 from section (a).

$$T_{Communicating} = [\sum_{i=1}^{r}(k_i-1)]*T_{lag}+T_{comm-per-byte}*[\sum_{i=1}^{r}[(k_i-1)(\prod_{j=1}^{i} p/k_j)]]*4*b]$$

$$T_{Computing} = T_{time-blend} * [\sum_{i=1}^{r}[(k_i-1)(\prod_{j=1}^{i} p/k_j)]]$$

$$T_{Idle} = x * (T_{Computing} + T_{Communicating})$$

Gather

This is the same for both and can be modeled as follows:

$$T_{Gather} = T_{Gather-Communicating} + T_{Integrate-pixels}$$

$$T_{Gather-Communicating} = (n-1)*(T_{lag}+T_{comm-per-byte}*(p/(n-1)*n)*4*b)$$

$$T_{IntegratingPixels} = n * (T_{copying-pixel} * p)$$

The total time would hence be estimated as:

$$T_{Total} = T_{Compositing} + T_{Gather}$$

and the value we would have to plugin the are:
- n: the number of processes
- p: size of the final image in pixel
- b: how many bytes are being used for each channel

and the times to be measured are:
- $T_{comm-per-byte}$
- $T_{time-blend}$ which is the time for doing 16 operation on the targeted platform.

Once we have values for these, we can see how much they differ from the total time and use that difference to get an estimate for x, x being how much should we account for load imbalance. Then it should be a pretty decent model of the performance of radix-k and direct-send.

References

1. E. W. Bethel, H. Childs, and C. Hansen. *High Performance Visualization: Enabling Extreme-Scale Scientific Insight.* Chapman & Hall/CRC, 1st edition, 2012.
2. W. Fang, G. Sun, P. Zheng, T. He, and G. Chen. Efficient pipelining parallel methods for image compositing in sort-last rendering. In *Proceedings of the 2010 IFIP International Conference on Network and Parallel Computing*, NPC'10, pages 289–298, Berlin, Heidelberg, 2010. Springer-Verlag.
3. M. Howison, E. W. Bethel, and H. Childs. Mpi-hybrid parallelism for volume rendering on large, multi-core systems. In *Proceedings of the 10th Eurographics Conference on Parallel Graphics and Visualization*, EG PGV'10, pages 1–10, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.
4. K. Moreland. Icet users' guide and reference. Technical report, Sandia National Lab, January 2011.
5. T. Peterka, D. Goodwell, R. Ross, H.-W. Shen, and R. Takur. A configurable algorithm for parallel image compositing applications. In *SC '09: Proceedings of ACM Supercomputing 2009*, pages 1–10, 2009.
6. H. Yu, C. Wang, and K.-L. Ma. Massively parallel volume rendering using 2-3 swap image compositing. In *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, SC '08, pages 48:1–48:11, Piscataway, NJ, USA, 2008. IEEE Press.

Qualifying Exam

Question 2

Pascal Grosset

University of Utah

1 Consider a n-dimensional rectilinear grid with a set of real values assigned to is vertices. A scalar field f(x) is defined by tensor product of linear interpolation along the edges (for n=2 this is a 2D grid with bilinear interpolation).

For a given real value w, a level set L(w) is the inverse image f-1 (w) of the isovalue w. Consider the following questions from a mathematical and algorithmic perspective providing formal arguments and/or counter examples as answers. In particular, pay attention to the formal definition of level set and all the special cases that may arise depending on the shape of the domain, the configuration of the function f(w) and the choice of isovalue w.

1.1 Is L(w) always manifold?

It is most of the time a manifold but not always. Consider for example a scalar field on a 2D grid. This would look like a terrain shown in figure 1 where the isocontour lines would be the level sets. Most of the isocontour lines are 1-manifold but some of them hit the boundary which would make them manifolds with boundary which is not really a manifold. Also at saddles, two circles of contour lines can touch and the resulting isocontour line is a figure eight, a lemniscate, instead of a circle. The intersection would make it not a manifold.

1.2 What is the dimension of L(w)?

It should usually be n-1. For a 2D grid, n=2, the level set is in the form of lines which are 1 dimensional. For a 3D grid of scalar values, n=3, the level set

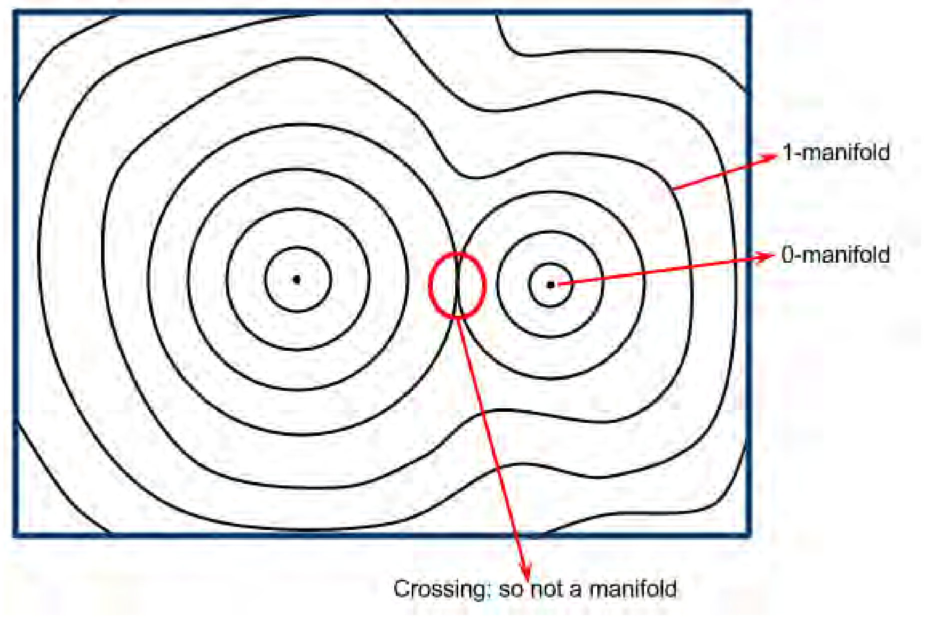


Fig. 1. Iscontour lines on grid show what a level set would look like for a 2D scalar field

is a surface, 2-dimensional, like what a marching cubes algorithm for example would extract. However, maximas and minimas will be different. They will be a vertex with 0-manifold. Figure 1 shows an example of that.

1.3 Is the dimension of L(w) uniform?

No. For example in a 2D grid that represents a terrain like in Figure 1, most of the level sets are circular in shape (1-manifold) but at a maximum, the level set is just a point (0-manifold). So the dimension is not uniform.

1.4 Provide an optimal algorithm for computing a single isosurface without preprocessing.

In 3D there is the marching cubes algorithms [5], marching squares for 2D, which has been extended by Bhaniramka et al. [1] to go n dimensions. For each hypercube in the n-dimensional grid, process each edge and check whether the scalar value w lies between the values on the two ends of the edge. That is, check if at one end of the edge the scalar value is greater than w and for the other end it's less than w. If that happens, this indicates a crossing of that edge by the isosurface for w. Linear interpolation can be used along the edge to determine the position of the crossing. Once we have the set of vertices, we can use a lookup table to see which configuration the set of points we have found match with and triangulate for it. Lorenen and Cline [5] have such a lookup table for 3 dimensional cases. Bhaniramka et al. [1] have a technique for generating lookup tables in any dimension or they can construct the isosurface on the fly. Both approaches are similar except that the lookup table approach might require lists of storage. Let U be the set of points that indicate where the isosurface crosses the cube. For each hypercube determine set of vertices V_h^+ whose value are greater or equal to w. Canonically construct a convex hull for V_h^+ and remove the simplices that lie on the hypercube and the remaining are the isosurface. As mentioned before, if we would like to contract a lookup table, we would repeat the same procedure but on all possible combinations of edge crossings.

1.5 Dividing the work between pre-processing and isosurface computation:

a. provide an optimal algorithm for computing an isosurface?

One of the ways that the marching cubes can be optimized is by preprocessing the hypercubes to know which ones contain which scalar values. This would speed up the marching cubes algorithm as instead of having to go through all the hypercubes, we could only go through those that will intersect with the isosurface to recreate the latter.

Since each hypercube will have a maximum and minimum scalar value, we need to find structure to represent that. Interval trees as proposed by Edelsbrunner [3] is an ideal candidate for storing intervals and it was used by Cignoni et al. [2]. The algorithm is as follows:

1. Sort all the scalar values at the vertices of the grid $(x_1, x_2, ..., x_n)$
2. Pick the median value m
3. This gives 3 sets of intervals, those less than m, those greater than m and those who span m
4. Each node has 2 pointers: the left one pointing to intervals less than m and a right one pointing to intervals greater than m
5. Each node has 2 lists: a list of intervals ordered by minimum vale and a list of intervals ordered by maximum value
6. For each of the left and right interval, we recursively pick the median value and repeat the steps from 2 to 5

For a scalar value s, we recursively query the tree, to find the node contain the scalar value s and for each hypercube. Once this is found, we only need to reconstruct the isosurface in the cube using marching cubes in 3D or the algorithm proposed by Bhaniramka et al. [1].

b. argue that the algorithm is optimal.

Given that the interval tree is a balanced binary tree, the maximum depth for n levels is $log\, n$. This means that we will step through at most $log\, n$ nodes before finding the right interval. $log\, n$ is as good as it gets when it comes to searching algorithms. So this is the optimal algorithm.

c. what is the cost of the preprocessing?

The cost of finding where to insert an element in a balanced tree is $log\, n$. Given that we have to do that for n hypercubes, the cost is $O(n\, log\, n)$.

1.6 If the purpose of the computation is to render the isosurface from a particular viewpoint. Is the algorithm provided to answer parts 4 and 5 still optimal in terms of big O complexity? If not how can it be improved?

Both algorithms provided in parts 4 and 5 will generate a complete isosurface no matter where the user is looking from. The algorithm in part 4 has O(n) complexity while the one in part 5 had O(log n) where n is the number of cells. However, only some of the cells processed will be visible when we take into account that the front part of the isosurface will conceal the back part. Livnat et al. [4] mention a saving of up to 93% in terms of isosurface size to show only the visible isosurface. So an algorithm is needed that would take into account the viewpoint, do visibility detection and thus avoid processing the whole isosurface could result in quite a big saving. How much will be saved depends more on the dataset and viewpoint that the algorithm.

The algorithm proposed by Livnat et al. [4] would work for that. Instead of storing the isosurface in an interval tree, an octree, as proposed by Wilhems and Van Gelder [6], can be used to store the data as it will allow us to traverse the data from front-to-back. At each node in the octree, the minimum and maximum value of the subtree is also stored. This will allow us to avoid exploring regions of the dataset that do not contain the relevant scalar value. In Wilhems and Van Gelder work, the lowest level of the octree would point to 8 cells which would contain the data.

The octree is traversed from front-to-back and for each triangle that is extracted, by looking up in table like for marching cube, we project it to a virtual screen that acts as an occlusion mask. When traversing nodes, we project them on the screen to determine if it will be visible or not. If they are not, all of their children won't be visible as well. A hierarchical frame buffer is used to help with usability testing. Each time a triangle is extracted, the hierarchical frame buffer is updated in a bottom-up fashion. For each new cell that we need to traverse, we project it into the frame buffer and compared with the hierarchical frame buffer to determine its visibility. When a cell is determined to be visible, marching cubes is used for triangulation.

References

1. P. Bhaniramka, R. Wenger, and R. Crawfis. Isosurface construction in any dimension using convex hulls. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):130–141, Mar. 2004.
2. P. Cignoni, P. Marino, C. Montani, E. Puppo, and R. Scopigno. Speeding up isosurface extraction using interval trees. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):158–170, Apr. 1997.
3. H. Edelsbrunner. Dynamic data structures for orthogonal intersection queries. Technical Report F59, Inst. Informationsverarb., Tech. Univ. Graz, Graz, Austria, 1980.
4. Y. Livnat and C. Hansen. View dependent isosurface extraction. In *Proceedings of the Conference on Visualization '98*, VIS '98, pages 175–180, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.
5. W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 163–169, New York, NY, USA, 1987. ACM.
6. J. Wilhelms and A. Van Gelder. Octrees for faster isosurface generation. *ACM Trans. Graph.*, 11(3):201–227, July 1992.

Qualifying Exam

Question 3

Pascal Grosset

University of Utah

1 This question concerns physical, mathematical, and perceptual issues of Depth of Field.

1.1 a) Definite Depth of Field, Depth of Focus, Blur, Circle of Confusion.

Most optical systems like the human eye and cameras (except for a pinhole camera) have a lens which has certain properties like a focal length and its behavior is defined by the thin lens equation:

$$\frac{1}{f} = \frac{1}{s} + \frac{1}{z_f} \tag{1}$$

where f is the focal length of the lens, s is the distance from the center of the lens to the image plane and z_f is the distance from the center of the lens to an object in the scene. That relation is shown in Figure 1(a)

Ideally, we want the lens to focus light rays from one point in the scene to one point on the image plane. When that happens the objects are said to be in focus but for a lens with a specific focal length and the image plane fixed at a certain distance, only objects within a specific distance - the depth of field region - in the scene will be in focus and appear sharp on the image plane. Figure 1(b) shows the distance from the lens where that happens. For objects that are not in focus, instead of getting mapped to a single point on the image plane, they get mapped to a circular region called the circle of confusion which can be computed from this equation:

$$c(z) = A\frac{|z - z_f|}{z} \tag{2}$$

where $c(z)$ is the diameter of the circle of confusion, A is the aperture of the lens.

The definitions:

- Depth of field : the region in a scene where the objects located in that region will appear in focus (sharp) on the image plane.
- Depth of focus : the region behind the lens where the image plane can be moved so that a point from the scene appears to be in focus. An object is in focus when the diameter of the circle of confusion is less than the pixel diameter.
- Blur : when instead of a point from scene being focussed to one point on the image plane, a point in the scene is focussed on a region and appears fuzzy, it is said to be blurred.
- Circle of confusion : the circular region on the image plane to which a point from the scene is mapped to when it is not in focus (blurred).

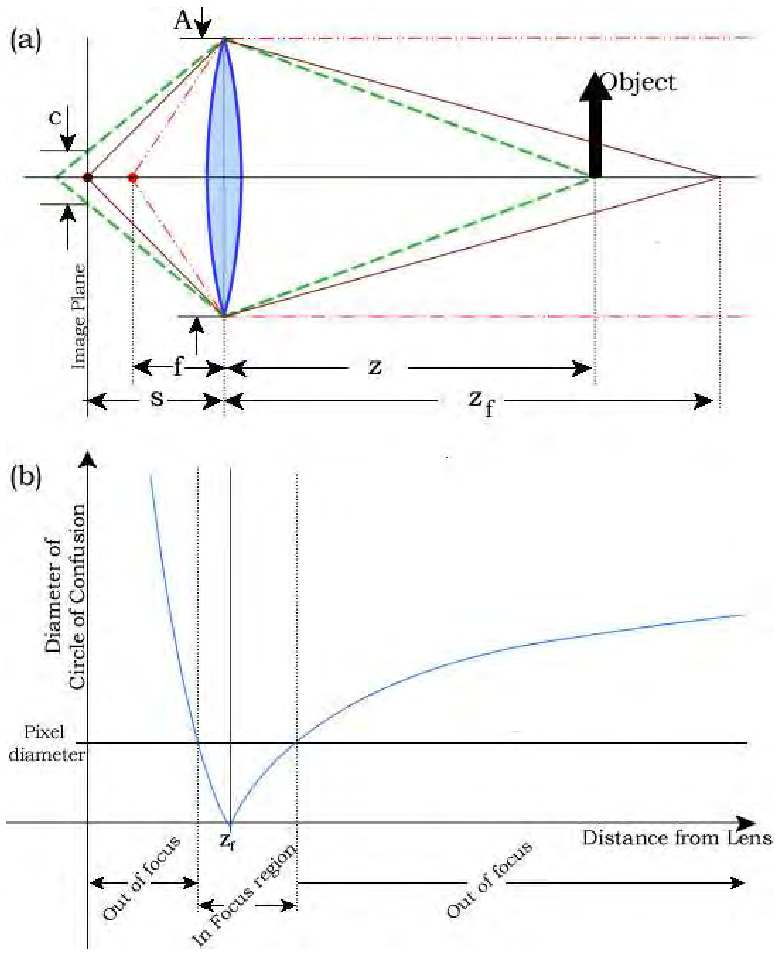


Fig. 1. (a) Lens Setup, (b) Circle of Confusion. Taken from [1]

1.2 For a camera lens and for a given image, the near and far depth of field (DOF) equations are:

$$NearBoundary = \frac{sf^2}{f^2 + Nc(s-f)} \tag{3}$$

$$FarBoundary = \frac{sf^2}{f^2 - Nc(s-f)} \tag{4}$$

where $s =$ subject distance, $f =$ focal length, $N =$ aperture, $c =$ minimum circle of confusion

Thus depth of field can be described by three parameters: focal length of the lens, size of the aperture, and the camera-subject distance. If you were to create a computational version of DOF, what would the corresponding computational parameters be? Create a diagram illustrating these concepts for a viewer looking at a screen (image plane) of a 3D visualization.

This is just a rewrite of equations 3 and 4 to use the same terms that I was using in part 1: A for aperture and z for subject distance.

$$NearBoundary = \frac{zf^2}{f^2 + Ac(z-f)} \tag{5}$$

$$FarBoundary = \frac{zf^2}{f^2 - Ac(z-f)} \tag{6}$$

We can also combine equations 2 and 1 to to get a better insight on the effects of focal length and size of aperture.

$$c(z) = A\frac{\left|z - (\frac{1}{f} - \frac{1}{s})\right|}{z} \tag{7}$$

The focal length of a lens quantifies the lens' ability to bend light. So a shorter focal length means a wider view while a longer focal length means a narrower view. Figure 2 shows an example of this. With a narrower view, there is also less blurring as the diameter of the circle of confusion tends to be smaller. So computationally, the focal length behaves like the field of view and there is an equation that maps the focal length f to the field of view fov for a vertical distance x, the blue line between the two black rays in figure 2b.

$$fov = 2 * arctan(\frac{2 * x}{f}) \tag{8}$$

The perspective project matrix [1] is as below:

[1]Taken from: https://www.safaribooksonline.com/library/view/webgl-programming-guide/9780133364903/appc.html

$$\begin{bmatrix} \frac{1}{aspect*tan(fov/2)} & 0 & 0 & 0 \\ 0 & \frac{1}{tan(fov/2)} & 0 & 0 \\ 0 & 0 & -\frac{Z_{far}+Z_{near}}{Z_{far}-Z_{near}} & -\frac{2(Z_{near}*Z_{far})}{Z_{far}-Z_{near}} \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

So using the projection matrix, we can extract the field of view used in a 3D scene (when specifying perspective projection we also specify aspect ratio and field of view) and compute the focal length using equation 8 and use the latter in the equations 3 and 4. If the projection is orthographic, the user will have to specify a value for the focal length.

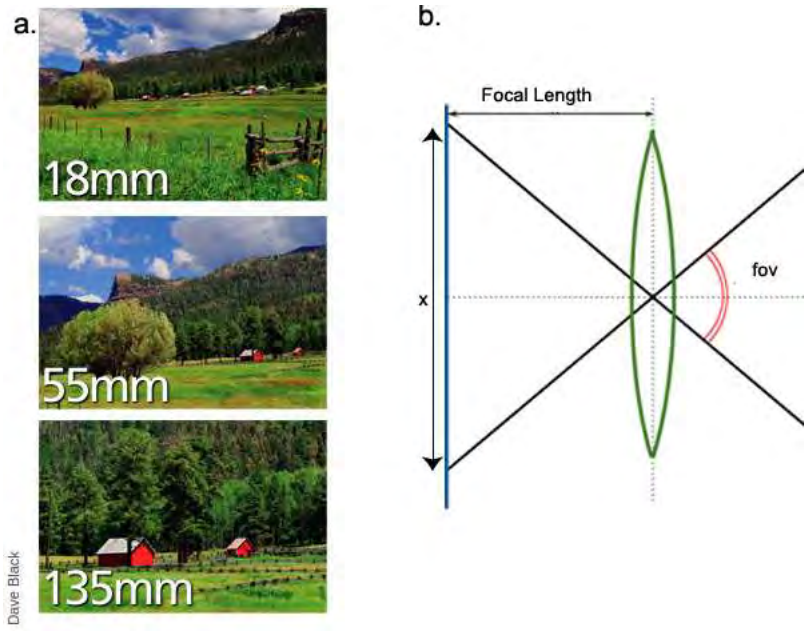


Fig. 2. a. This figure shows how a lens with different focal length captures the same scene. The 18mm focal length provides a much wider view of the scene compared to the 135 mm focal length. Taken from the Nikon website: http://www.nikonusa.com/en/Learn-And-Explore/Article/g3cu6o2o/understanding-focal-length.html b. This show the relation between focal length and field of view

Aperture defines how much light we allow through the lens. From equation 7, we can see that aperture affects how much blurring we have for depth of field. From equations 5 and 6, we can also see that it impacts the position of the near and far boundary of the depth of field. Since the aperture is also strongly linked to the size of the circle of confusion, this is a parameter that the user will have to set. Schott et al. [2] asked the user to enter an angle alpha which would control the aperture. From the angle value, the Aperture was computed as: $A(z_f) = 2tan(\frac{\alpha}{2})z_f$. The same approach can be used.

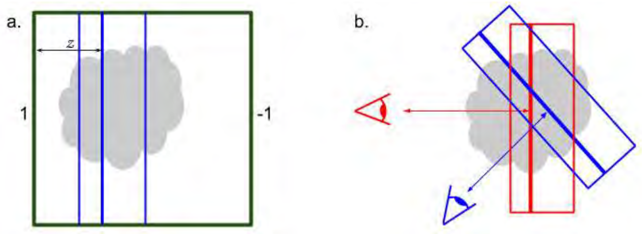


Fig. 3. a. Subject distance in normalized device coordinates b. Subject distance with Near and Far DOF boundary

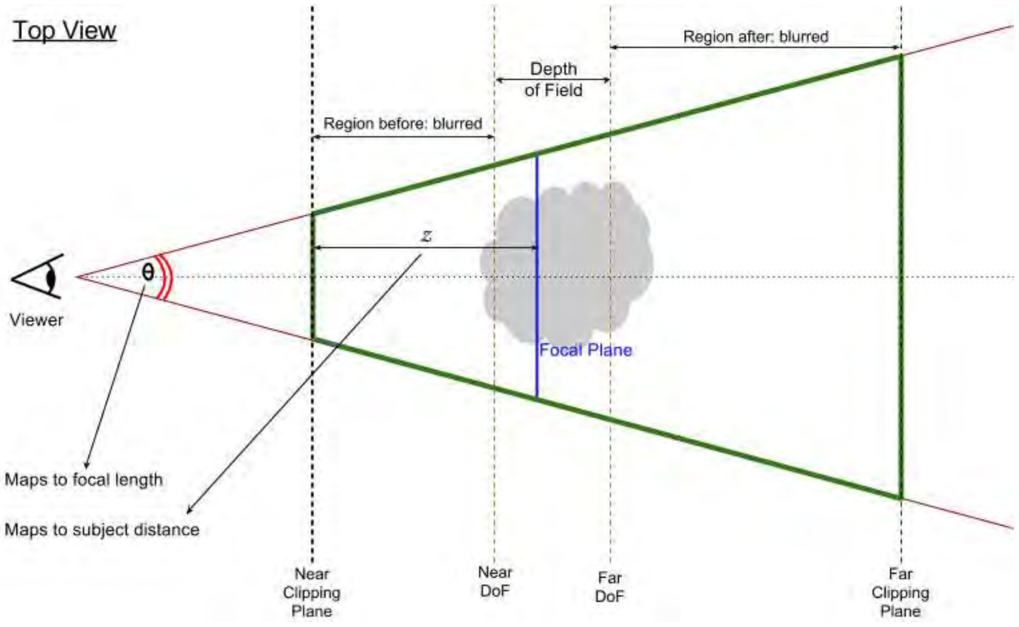


Fig. 4. Viewer's perspective of a 3D scene. z is the subject distance and θ is the field of view which would map to focal distance.

Subject distance is just the distance of the subject that the camera is focussing on. So computationally, the subject distance would be a fixed distance from the camera to where the user would like to focus on irrelevant of the angle at which the viewer is looking at an object in the scene. However, since in a 3D scene, we only see as from the near clipping plane, figure 4, z should be only from the near clipping plane else it would be very confusing to a user. This can even further be simplified to a fraction between 0 and 1, see figure 3, to indicate at which range inside the clipping volume the focus should be at. In Figure 3a., z would be set to about $\frac{1}{3}$.

So to summarize, the focal length is derived from the field of view parameter, the subject distance is manipulated by the user and for the aperture, we ask the user to enter a value. Figure 4 shows an overall view of this setup. θ that represents the field of view would map to the focal length for equations 3 and 4. The fraction into which the subject is in between the near and far clipping plane would match to the subject distance and the aperture would have to be entered separately as an additional parameter.

References

1. A. Grosset, M. Schott, G.-P. Bonneau, and C. Hansen. Evaluation of depth of field for depth perception in dvr. In *Visualization Symposium (PacificVis), 2013 IEEE Pacific*, pages 81–88, Feb 2013.
2. M. Schott, A. V. P. Grosset, T. Martin, V. Pegoraro, S. T. Smith, and C. D. Hansen. Depth of field effects for interactive direct volume rendering. In *Proceedings of the 13th Eurographics / IEEE - VGTC Conference on Visualization*, EuroVis'11, pages 941–950, Aire-la-Ville, Switzerland, Switzerland, 2011. Eurographics Association.

Qualifying Exam

Question 4

Pascal Grosset

University of Utah

1 Compare and contrast the following papers on AMR volume rendering:

1. Stephane Marchesin and Guillaume Colin de Verdiere High-Quality, Semi-Analytical Volume Rendering for AMR Data IEEE TVCG, VOL. 15, NO. 6, 2009
2. Weber, G.H., Kreylos, O., Ligocki, T.J., Shalf, J.M., Hagen, H., Hamann, B., Joy, K.I. and Ma, K.-L. (2001), High-quality volume rendering of adaptive mesh refinement data (pdf), in: Ertl, T., Girod, B., Greiner, G., Niemann, H. and Seidel, H.-P., eds., Vision, Modeling, and Visualization 2001, IOS Press, Amsterdam, The Netherlands, pp. 121-128.
3. Joachim E. Vollrath, Tobias Schafhitzel, and Thomas Ertl Employing Complex GPU Data Structures for the Interactive Visualization of Adaptive Mesh Refinement Data Volume Graphics, 2006, pp. 55-58
4. Kaehler, R., Prohaska, S. , Hutanu, A., Hege, H.-C. Visualization of time-dependent remote adaptive mesh refinement data IEEE Visualization 2005, pp. 175 - 182

Since these papers present different approaches to solving AMR, some are for CPUs, some for GPUs, some focus on cell centered data, others on vertex centered, I will start by presenting a summary of the paper and detail their contribution and then comment on how the resemble or differ from each other.

1. Semi-Analytical Volume Rendering for AMR Data

This paper presents an analytical approach for volume rendering vertex-centered data which can also applied to AMR data. A k-d tree is then used to store the different sub volumes on a CPU. These are then sent to a GPU for rendering.

To sample the scalar value at a specific location $s(x, y, z)$ inside an hexahedral cell, we most of the time use trilinear interpolation to interpolate among the different vertices of a vertex centered dataset. In this paper, the authors use a parametric representation of trilinear interpolation and the result is a third degree polynomial $s(t)$. During volume rendering, rays are sent in a volume and sampling is done at regular intervals along that ray. To optimally sample along the ray and make sure not to miss any relevant features, the authors propose to sample at the entry point and exit point of the cell as well as the local maximum or minimum in that cell. That extrema is obtained through the derivative of $s(t)$ which is a quadratic. For the transfer function, they use a formulation similar to the pre-integrated classification of Engel and Ertl [2]. For volume shading, the gradient is needed and for that they compute the partial derivatives of $s(x, y, z)$ and come up with a parametric representation of that $d(t)$. By integrating $s(t)$ and $d(t)$ in the volume rendering equation, they produce a complete analytical representation of the volume rendering equation.

To use this in AMR data, they need to take care of the discontinuities that exist at the edge between a fine and a coarse cell. At the boundary of AMR levels, a cell can have for one face one neighbor and for the opposite face, four neighbors. This will break the trilinear interpolation at this boundary cell. So that cell is split into small cells so that trilinear interpolation can be applied and thus an analytical solution would work.

The AMR patches are then stored in a k-d tree on the CPU. Each leaf of the k-d tree is repeatedly split until a leaf contains cells belonging to the same AMR level. In practice, each left can just store an AMR patch. Once this is done, the volume is traversed from front to back and whenever a patch needs to be rendered, it is sent to the GPU for fast rendering.

2. High-quality volume rendering of adaptive mesh refinement data.

In this paper, Weber et al. tackle the issues in rendering cell based AMR data. They present a system where a stitch mesh is used to fill in the gaps between different refinement levels and use cell projection to render the data.

In volume rendering, we usually deal with data found at the vertices of a grid but with cell centered data, the data is at the center of a cell. To get past this issue, the paper considers the dual representation which would center the grid at the center of each cell. However, using the dual representation shortens a grid on each side. For example, for a grid with 5 x 5 vertices, the dual a grid of size 4 x 4 and is located inside the 5 x 5 grid. This creates gaps between grids. Moreover with AMR data, where we have a fine grid covering a coarse grid the gaps becomes even more noticeable.

To deal with that, Weber at al. propose that a stitch mesh is constructed that interpolates the values between the dual grids at the gaps. Since the grid points between a fine a coarse mesh are not aligned, the stitch mesh is made up of pyramidal, deformed triangle prisms and deformed hexahedral cell depending on the position of the grid points. This approach is explained in details in [4]. For gaps between grids of the same level, the cells are just regular cubes. When doing volume rendering, we want to sample values inside a cell. For a cube it is simple but for the deformed cells, the papers explains how to sample values for each shape.

To do volume rendering, they use cell projection presented by Ma and Crockett [3]. Rays are traced through the volume and each cell determine the rays that pass through it. Each pixel collects the ray segments from the cells that will contribute to the color value of that pixel. These segments are stored in a queue based on the depth information of that pixel. To get the scalar value from the volume for these segments, interpolation is done with the cell the ray segment maps to and the emission absorption model is used to get the color value.

To finally render the AMR data, each level of coarseness is cell projected separately. A bottom up approach would involve rendering the finest level first followed by coarser levels.

3. Employing Complex GPU Data Structures for the Interactive Visualization of Adaptive Mesh Refinement Data

In this paper, the whole of the AMR data is placed on the GPU in either an octree texture of an adaptive page table data structure. Rendering is then trivial for the adaptive page table or for data at the same level for an octree. Data at different levels require an additional interpolation step for the octree.

For the octree representation, an octree texture is constructed over the whole 3D grid. Each of the 8 nodes in the octree is represented by a texel which indicates whether that node points to the actual data, is empty or is an internal left node. For volume rendering, we will send rays and sample along the ray. To obtain the sample at one sampling location, we will need to do trilinear interpolation and retrieve 8 samples. If all the samples are at the same AMR level, interpolation is trivial. If some of the samples are at one refinement level and at another level, we need to do some more work. If the exact location falls in a coarse cell and some of the samples returned are in a fine cell, the latter can be average to a coarser level to allow for trilinear interpolation. If it is the opposite case, the coarse cells can be interpolated to get finer cells which can be easily trilinearly interpolated.

Adaptive page table is another data structure than can be represented using textures on the GPU. It maps a virtual domain to the physical memory on the graphics card. The AMR dataset is mapped into pages and each page is recursively subdivided until it matches the size of the smallest cell in the AMR dataset. Sometimes it leads to oversampled cells which increase the storage requirements but it speeds up access and thus rendering. Also since there is no guarantee that pages adjacent in the virtual domain will be adjacent in the graphics memory, there is an extra layer of ghost cells padded to physical pages which increases memory requirement. Once these are set though rendering is fast.

4. Visualization of time-dependent remote adaptive mesh refinement data

This paper presents a system where time-dependent AMR data is stored remotely on a cluster but visualization takes place on the local graphics hardware. So the data must be streamed from the cluster to the local PC. The main focus of this paper though is how to visualize intermediate time steps of an AMR simulation, time steps where some of the fine level have data while the coarse level might not as one simulation step for the coarse level has completely stepped over these smaller intermediate time steps.

The first section of this paper explains the mechanics of an AMR simulation. How initially we start with a coarse grid and need to refine regions of the grid based on error estimators. Grids with a coarse level of refinement move forward a big time step during the simulation while grids with finer level of refinements move ahead in smaller intermediate time steps and so the finer grid needs several iterations in time in order to catch up with the coarse grid. If the user wants to visualize data at one of these intermediate time steps, we have a problem as that intermediate time step is undefined in the coarse level grid. To solve this issue, they do the following. Firstly, all the sub grids at the same refinement levels are merged. In doing so, information about the boundaries of the sub grids are lost. This poses a problem as this makes the new merged sub grid hard to store. A solution would be to store the sub grid as an unstructured mesh but this not very convenient. So the merged grid is broken down into many small sub grids which are then clustered together (to reduce the number of sub grids) using [1]. Once this is done, we have a nice intermediate AMR grid with different levels. We must fill these grids and their cells with values. If the cell existed at that refinement level for a time step before and after, it's a matter of using an interpolation scheme. In the paper they use continuous C^0 piecewise linear interpolation or the continuous C^0 cubic hermit interpolation. If the level of refinement did not exist previously, we interpolate data from the next coarsest level of refinement before doing interpolation. Finally, the data on the coarser levels are updated by data the finer grid as the latter is more accurate. All of this is done remotely on the server and the grid with data can now be streamed to be rendered on the local computer. In this paper, this is done using Remote Procedure Call and the SOAP web protocol.

As can be seen from the summaries of these four papers, they focus on quite different things. The focus of the first paper is to find an analytical approach to volume rendering and they show that it can be used for AMR.

The second paper focusses on rendering cell based data and present ways to get past issues linked with cell-based AMR. The third paper only want to see if more complex data structures like octrees and adaptive page table would work on the GPU for representing AMR data and the focus of the last paper is rendering intermediate time steps in an AMR simulation. Nevertheless, we can group the discussion on how they address some important aspects of volume rendering.

1. Dataset: Cell Centered vs Vertex Centered

The work of Weber et al. (paper 2) focus on cell centered data. So they create a dual mesh and have to deal with the gaps resulting from the creation of dual meshes. Marchesin et al. (paper 1) deal with exclusively vertex centered data where the issue of gaps do not come up at all. The others are quite vague about it but since they do not mention anything about stitching, they are probably using vertex centered data.

2. Rendering

Weber et al. (paper 2) is a purely CPU based rendering paper while all the other leverage the GPU for rendering. Because of the obnoxious cells created by their stitching approach, they use cell projection for volume rendering which is more common in unstructured grid rendering. The others use GPU for rendering. Marchesin et al. (paper 1) use code their analytical rendering formulation in the shader program. For Joachim et al. (paper 3), they use use both an octree and a page table for GPU rendering. Kaehler et al. (paper 4) only mention that rendering is on the GPU.

3. Data Organization

Using a data structure to organize AMR data is quite common. Marchesin et al. (paper 1) use k-d tree for storing the arm meshes. Joachim et al. (paper 3) experiment with both an octree and an adaptive page table while the others remain vague about it.

As mentioned before, these papers have different focus and each devised interesting approaches to solve their problem.

1.1 For the Uintah AMR data structures, would any of these methods suffice and if not, what changes/new techniques would be required and why.

The uintah dataset has both cell-centered and vertex centered. So some the approach by Weber et al. (paper 2) would work for cell-centered rendering. Vertex-centered rendering is usually trivial and we could also use the work by Marchesin et al. (paper 1) for that. However, one of the reason why a different approach is required is because of the goal of the uintah project. In terms of rendering, the GPU rendering would work fine but the focus of the uintah project is to be able to render remotely on the server which is at the opposite end of paper 4 which does rendering on the GPU. Finally using a kd-tree is a good idea which is already being done for visualizing data in Uintah. However, it is the goal of the project that is generating demands for new techniques.

The goal of the project is to allow the rendering of massively large datasets that will become more and more common as we advance in the exascale era of computing. Also, what we presume is going to happen with exascale systems is that they will be less focussed on GPUs but will rather move in the direction of processors with hundreds and thousands of cores like the upcoming Intel Knight Landing processor. Three of the four papers focussed on using GPU rendering which will not be relevant. CPU rendering through ray casting is trivial though and the same approach being used on the GPU can be used on the CPU. The main issue is with scaling up the number of nodes. This will drastically increase communication during the compositing phase. None of the papers mention compositing as they all render all the data on the same machine. Good compositing algorithms like radix-k and binary swap do exist but they were created at a time where the focus was on balancing the workload. Computing power is starting to be cheap with the new generation of CPUs. What we must try to decrease is communication cost. Also, these existing compositing algorithms assume that each node will have one image and break down if we have several images. With AMR, one node might have more than one image. A fine patch might overlap coarse AMR patches that exist on two or more nodes. The fine AMR patch can also cause the coarse patches to be concave. So we must find new compositing algorithms that minimize communication in general and work for AMR as well.

References

1. M. Berger and I. Rigoutsos. An algorithm for point clustering and grid generation. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(5):1278–1286, Sep 1991.
2. K. Engel and T. Ertl. Interactive high-quality volume rendering with flexible consumer graphics hardware, 2002.
3. K.-L. Ma and T. Crockett. A scalable parallel cell-projection volume rendering algorithm for three-dimensional unstructured data. In *Parallel Rendering, 1997. PRS 97. Proceedings. IEEE Symposium on*, pages 95–104, 119–20, Oct 1997.
4. G. H. Weber, O. Kreylos, T. J. Ligocki, J. M. Shalf, H. Hagen, B. Hamann, and K. I. Joy. Extraction of crack-free isosurfaces from adaptive mesh refinement data. pages 25–34. Springer Verlag, 2001.

1 Overview of Nelson-Oppen Theory Combination

Nelson-Oppen combination is a method of combining decision procedures of different theories to solve formulas which are mixed with expressions in different theories. Since every decision procedure only solves formulas in one particular theory, a formula mixed with multiple theories can not be solved by a single decision procedure. Nelson-Oppen combination is not a decision procedure but a methodology of collaborating different solvers to handle formulas in combination theory.

High Level View of Nelson-Oppen: To solve a formula φ (in a conjunction form) which contains expressions in two different theories, T_1 and T_2 (with their signatures $\sum_1$ and $\sum_2$), we can introduce variables to split closures in φ which contain multiple symbols from T_1 and T_2. We stop splitting closures until each of them belongs to only one theory. Then we can partition the closures into two sub-formulas (F_1 and F_2) based the theories. Suppose F_1 belongs to theory T_1 and F_2 belongs to theory T_2, we solve them separately by their decision procedures. If F_1 or F_2 is unsatisfiable (UNSAT), φ is UNSAT. Otherwise, from F_1 or F_2, some equality relationships between variables of φ may be inferred. In the case that F_1 can infer an equality between two variables but such relationship cannot be inferred from F_2, F_1 passes this "information" to F_2. Then we solve F_1 and F_2 again. If both F_1 and F_2 are satisfiable (SAT) and have no more "unique" information, φ is SAT.

Information Exchange Between Decision Procedures: The information inferred from a single-theory formula (says F_1 mentioned in the previous paragraph) is a disjunction of equalities between variables. However, if T_1 is a convex theory, F_1 must be able to infer a single equality between two variables. (This is based on the definition of convex theory. The single equality is one of the terms in the disjunction.) In this case, the information flows from F_1 to F_2 is a single equality. We should prioritize the exchange of single equality. For a non-convex theory, it may generate a disjunction of equalities. (Let's assume T_2 is non-convex.) There may be a case that each equality in the disjunction can not be individually inferred from F_2. In this case, a non-convex theory involved, Nelson-Oppen method works in a recursive mode. It needs to consider each equality separately. Let's $a = b \vee c = d$ is a disjunction inferred from F_2. We may add $(a = b)$ to F_1 and F_2 and continue. If no more information can be inferred at the end and both F_1 and F_2 are UNSAT, we need to roll back and try $(c = d)$ instead. If we exhaust every possible exchange of equality and each of the exchanges results in UNSAT, the whole formula is UNSAT. If any SAT found (means all sub-formulas are SAT) and no more information could be generated, the whole formula is SAT. It means that we can find an interpretation, a witness, to satisfy the formula.

2 Restrictions of Nelson-Oppen Theory Combination

For a formula φ which contains theories $T_1, \ldots, T_n$ with their signatures $\sum_1, \ldots, \sum_n$, it can be solved by Nelson-Oppen combination if the following restrictions are met:

- $T_1, \ldots, T_n$ are quantifier-free first-order theories with equality.
- Every theory in $T_1, \ldots, T_n$ has its decision procedure.
- $\forall i, j \; s.t. \; (1 \leq i, j \leq n) \wedge (i \neq j)$. $\sum_i \cap \sum_j = \{=\}$. This restriction is for dividing φ into single-theory sub-formulas.
- $T_1, \ldots, T_n$ are theories over infinite domains.

3 An Illustrative Example

We show how Nelson-Oppen combination works on the formula

$$\varphi = (1 \leq x) \wedge (x \leq 2) \wedge (f(x) \neq f(1)) \wedge (f(x) \neq f(2))$$

which contains uninterpreted functions and linear integer arithmetic. (Suppose that we have decision procedures for these two theories.)

Step 0: Formula φ will be firstly divided in to two sub-formulas: F_1 and F_2. F_1 is in linear integer arithmetic. F_2 is in uninterpreted function theory. The following table shows the division result:

Formula Name	F_1		F_2
Formula	$(I_1 = 1)$ $(I_2 = 2)$ $(1 \leq x)$ $(x \leq 2)$		$(f(x) \neq f(I_1))$ $(f(x) \neq f(I_2))$
Inference and Information Flow			

We introduce two fresh variables, I_1 and I_2, to replace the constants in F_2.

Step 1: Since both F_1 and F_2 are SAT, we try to generate information. An information, a disjunction $(x = 1) \vee (x = 2)$, inferred by F_1 is shown in the following table:

Formula Name	F_1		F_2
Formula	$(I_1 = 1)$ $(I_2 = 2)$ $(1 \leq x)$ $(x \leq 2)$		$(f(x) \neq f(I_1))$ $(f(x) \neq f(I_2))$
Inference and Information Flow	$(x = 1) \vee (x = 2)$		

Step 2: Since the inference generated by F_1 is a disjunction, the whole process is split: F_1 could pass either $(x = 1)$ or $(x = 2)$ to F_2. The equality passed to F_2 will also be considered in F_1's decision procedure. In this step, we consider the case of passing $(x = 1)$ first.

Formula Name	F_1		F_2
Formula	$(I_1 = 1)$ $(I_2 = 2)$ $(1 \leq x)$ $(x \leq 2)$		$(f(x) \neq f(I_1))$ $(f(x) \neq f(I_2))$
Inference and Information Flow	$(x = 1)$	$\Rightarrow$	

The symbol $\Rightarrow$ doesn't mean implication here. I just borrow this symbol to indicate information flow.

Step 3: Both F_1 and F_2 are SAT. However, F_1 can infer $(I_1 = x)$ now. It passes this new information to F_2 which is shown as follows.

Formula Name	F_1		F_2
Formula	$(I_1 = 1)$ $(I_2 = 2)$ $(1 \leq x)$ $(x \leq 2)$ $(x = 1)$		$(f(x) \neq f(I_1))$ $(f(x) \neq f(I_2))$ $(x = 1)$
Inference and Information Flow	$(x = I_1)$	$\Rightarrow$	

Step 4: With $(x = I_1)$, F_2 is UNSAT. Thus the procedure rolls back to step 1, and F_1 passes $(x = 2)$ to F_2 instead.

Formula Name	F_1		F_2
Formula	$(I_1 = 1)$ $(I_2 = 2)$ $(1 \leq x)$ $(x \leq 2)$ $(x = 1)$ $(x = I_1)$		$(f(x) \neq f(I_1))$ $(f(x) \neq f(I_2))$ $(x = 1)$ $(x = I_1)$ **UNSAT!!!**
Inference and Information Flow			

Step 5: We skip the details of passing $(x = 2)$ from F_1 to F_2 in step 1. The consequence is an UNSAT scenario shown in the following table. Then we will need to roll back to step 1 again. However, we already exhausted all information in the disjunction generated in step 1. It means that any possible information exchange results in UNSAT. Thus the whole formula, φ, is UNSAT.

Formula Name	F_1		F_2
Formula	$(I_1 = 1)$ $(I_2 = 2)$ $(1 \leq x)$ $(x \leq 2)$ $(x = 2)$ $(x = I_2)$		$(f(x) \neq f(I_1))$ $(f(x) \neq f(I_2))$ $(x = 2)$ $(x = I_2)$ **UNSAT!!!**
Inference and Information Flow			

Appendix P: Sample Thesis/Dissertation Abstracts

SOFT SHADOW MIP-MAPS

by

Yang Shen

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computing

School of Computing

The University of Utah

August 2016

ABSTRACT

This document introduces the Soft Shadow Mip-Maps technique, which consists of three methods for overcoming the fundamental limitations of filtering-oriented soft shadows. Filtering-oriented soft shadowing techniques filter shadow maps with varying filter sizes determined by desired penumbra widths. Different varieties of this approach have been commonly applied in interactive and real-time applications. Nonetheless, they share some fundamental limitations. First, soft shadow filter size is not always guaranteed to be the correct size for producing the right penumbra width based on the light source size. Second, filtering with large kernels for soft shadows requires a large number of samples, thereby increasing the cost of filtering. Stochastic approximations for filtering introduce noise and prefiltering leads to inaccuracies. Finally, calculating shadows based on a single blocker estimation can produce significantly inaccurate penumbra widths when the shadow penumbras of different blockers overlap.

We discuss three methods to overcome these limitations. First, we introduce a method for computing the soft shadow filter size for a receiver with a blocker distance. Then, we present a filtering scheme based on shadow mip-maps. Mipmap-based filtering uses shadow mip-maps to efficiently generate soft shadows using a constant size filter kernel for each layer, and linear interpolation between layers. Finally, we introduce an improved blocker estimation approach. With the improved blocker estimaiton, we explore the shadow contribution of every blocker by calculating the light occluded by potential blockers. Hence, the calculated penumbra areas correspond to the blockers correctly. Finally, we discuss how to select filter kernels for filtering.

These approaches successively solve issues regarding shadow penumbra width calculation apparent in prior techniques. Our result shows that we can produce correct penumbra widths, as evident in our comparisons to ray-traced soft shadows. Nonetheless, the Soft Shadow Mip-Maps technique suffers from light bleeding issues. This is because our method only calculates shadows using the geometry that is available in the shadow depth map. Therefore, the occluded geometry is not taken into consideration, which leads to light bleeding. Another limitation of our method is that using lower resolution shadow mip-map

layers limits the resolution of the shadow placement. As a result, when a blocker moves slowly, its shadow follows it with discrete steps, the size of which is determined by the corresponding mip-map layer resolution.

SCALABLE SPATIAL SCAN STATISTICS

by

Raghvendra Singh

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computing

School of Computing

The University of Utah

August 2015

ABSTRACT

We present algorithms for detecting spatial anomaly in a time efficient manner. There are many other approaches solving the same problem but are facing a serious issue of very huge computational time. We came up with some novel algorithms which help us to solve the problem in a time efficient manner for very large data sets. We tried to show, by executing experiments on both synthetic and real world data set, that the results obtained from original data set and the sampled data set are very similar and therefore we executed all our approaches on sampled data set rather than on the original data set. Thus we saved a lot of computational time by using sampled data set as an input to our approaches.

CeNet- CAPABILITY ENABLED NETWORKING: TOWARDS LEAST-PRIVILEGED NETWORKING

By

Jithu Joseph

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

School of Computing

The University of Utah

July 2015

ABSTRACT

In today's IP networks any host can send packets to any other host irrespective of whether the recipient is interested in communicating with the sender or not. The downside of this openness is that every host is vulnerable to an attack by any other host. We observe that this unrestricted network access from compromised systems is also a main reason for data ex-filtration attacks within corporate networks. We address this issue using the network version of capability based access control.

We bring the idea of *Capabilities* and Capability based access control to the domain of Networking.

Network capabilities can be passed between hosts thereby allowing a delegation-oriented security policy to be realized. We believe that this base functionality can pave the way for the realization of sophisticated security policies within an enterprise network.

Further we built a policy manager which is able to realize Role-Based Access Control (RBAC) policy based network access control using capability operations. We also look at some of the results of formal analysis of capability propagation models in the context of networks.

SEACAT: AN SDN END-TO-END CONTAINMENT ARCHITECTURE

by

Makito Kano

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

School of Computing

The University of Utah

May 2015

ABSTRACT

Healthcare organizations heavily rely on networked applications. Many applications used in healthcare settings have different security, privacy, and regulatory requirements. At the same time, users may use their devices with medical applications for non-medical related purposes. Running arbitrary applications on the same device may affect the healthcare applications in a way that violates their requirements. The ability of using the same device for multiple purposes in an enterprise network presents a challenge to healthcare IT operations. To allow the users to use the same device for both medical and non-medical related purposes while meeting the set of requirements for medical applications, we present the design and implementation of the SeaCat, an SDN End-to-end Application Containment ArchitecTure, and evaluate the system in a testbed environment. SeaCat has two major components. First is the container technology used in the client device to securely isolate any application. Second is the software-defined networking (SDN) that provides isolated secure network resource access for each application.

CONCEPT AWARE CO-OCCURRENCE AND ITS APPLICATIONS

by

Klemen Simonic

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

School of Computing

The University of Utah

2015 May

ABSTRACT

Term cooccurrence data has been extensively used in many applications ranging from information retrieval to word sense disambiguation. There are two major limitations of cooccurrence data. The first limitation is known as the data sparseness problem or the zero frequency problem: For a majority of pairs, the probability that they co-occur in even a large corpus is very small. The second limitation is that in cooccurrence data, each term is considered as a meaningless symbol, or in other words, terms do not have types, or any semantic relationships with other terms. In this paper, we introduce a novel approach to address these two limitations. We create concept aware cooccurrence data wherein each term is not a symbol, but an entry in a large scale, data driven semantic network. We show that with concepts or types, we are able to address the data sparseness problem through generalization. Furthermore, using concept co-occurrence, we show that our approach can benefit a large range of applications, including short text understanding.

Appendix Q: Partial List of School’s Graduate Students and their Employment Information

Students Name	Degree		Year Completed	First Company	First Title/Role	Present Company	Present Title/Role	Advisor That
Pascal Grossett	Ph.D.		2016	LANL	PostDoc	LANL	PostDoc	Chuck Hanse
Kevin Wall	MS		2016	Self-employed (Startup)		Self-employed (Startup)		Chuck Hanse
Mark Kim	Ph.D.		2016	ORNL	PostDoc	ORNL	PostDoc	Chuck Hanse
Manasa Prosad	MS		2015	Google				Chuck Hanse
Brad Loos	Ph.D.		2014	Bungee Games		Bungee Games		Chuck Hanse
Liang Zhou	Ph.D.		2014	University of Stuttgart	PostDoc	University of Stuttgart	PostDoc	Chuck Hanse
Yong Wan	Ph.D.		2013	University of Utah	PostDoc	University of Utah	PostDoc	Chuck Hanse
Carson Brownlee	Ph.D.		2012	TACC		Intel		Chuck Hanse
Mathias Scott	Ph.D.		2011	Nvidia		Nvidia		Chuck Hanse
Siddarth Shankar	MS		2009			CDADAPT Co		Chuck Hanse
Jianrung Shu	MS		2009	Numira Biosciences				Chuck Hanse
Daniel Kopta	Ph.D.		2015	NVIDIA	Senior OptiX Engineer	SoC at U	Assistant Prof	Cem Yuksel
Yang Shen	MS		2015	Originate	Software Engineer	Originate	Software Engineer	Cem Yuksel
Yuntao Ou	MS		2014	AMD	GPU Software Engineer	AMD	GPU Software Engineer	Cem Yuksel
Amit Prakash	MS		2016	AMD	GPU Software Engineer	AMD	GPU Software Engineer	Cem Yuksel
Wei-Fan Chiang	Ph.D.		2016	Delaware	PostDoc			Ganesh Gopa
Vishal Sharma	Ph.D.		2016	Microsoft				Ganesh Gopa
Sriram Aananthakrishnan	Ph.D.		2016	Intel Labs				Ganesh Gopa
Peng Li	Ph.D.		2015	Fujitsu Research		Samsung Research		Ganesh Gopa
Subodh Sharma	Ph.D.		2012	Oxford	PostDoc	Computer Science, IIT Delhi	Assistant Professor	Ganesh Gopa
Anh Vo	Ph.D.		2011	Microsoft				Ganesh Gopa
Sarvani Vakkalanka	Ph.D.		2010	Microsoft		VMWare		Ganesh Gopa
Guodong Li	Ph.D.		2010	Fujitsu research		Codiscope (security firm)		Ganesh Gopa
Yu Yang	Ph.D.		2009	Microsoft		Pinterest		Ganesh Gopa
Tyler Sorensen	MS	BS/MS T	2014	Imperial College	PhD student			Ganesh Gopa
Wei-Fan Chiang	MS	BS/MS T	2010	University of Utah	PhD student			Ganesh Gopa
Ben Meakin	MS	Thesis	2010	Qualcomm				Ganesh Gopa
Bruce Bolick	MS	BS/MS N	2014					Ganesh Gopa
Leif Andersen	MS	BS/MS N	2014			Northeastern Univ	PhD student	Ganesh Gopa
Joe Mayo	MS	BS/MS N	2010	Microsoft				Ganesh Gopa
Brandon Gibson	MS	BS/MS N	2010	Microsoft				Ganesh Gopa
Grant Ayers	MS	BS/MS N	2010			Stanford	PhD CS student	Ganesh Gopa
Alan Humphrey	MS	BS/MS N	2009			University of Utah	PhD student	Ganesh Gopa
Carson Jones	MS	BS/MS N	2009					Ganesh Gopa
Chris Derrick	MS	BS/MS N	2009					Ganesh Gopa
KaiQiang Wang	MS	MS Proje	2016	Google		Google		Jeff Phillips
Liang Zhang	MS	MS Proje	2015	Microsoft				Jeff Phillips
Raghvendra Singh	MS	MS Thes	2015	InsideSales		Overstock.com		Jeff Phillips
Amey Desai	MS	MS Thes	2014	UrbanEngines		Google		Jeff Phillips
Shashanka Krishnaswamy	MS	MS Proje	2013	Amazon				Jeff Phillips
Supraja Jayakumar	MS	MS Proje	2013	Cerner Systems				Jeff Phillips
Yuan Fang	MS		2013	Zillow				Jeff Phillips
Lingbing Jiang	MS		2012	Microsoft				Jeff Phillips
Jon Rafkind	Ph.D.		2013	HP	Senior software engineer	HP	Senior software enginee	Matthew Flat
Kevin Tew	Ph.D.		2012	BYU	Assistant professor	BYU	Assistant professor	Matthew Flat

Students Name	Degree	Year Completed	First Company	First Title/Role	Present Company	Present Title/Role	Advisor That
Kevin Atkinson	Ph.D.	2011	Rice U. / U. Halmstad	Researcher			Matthew Flat
Eric Eide	Ph.D.	2009	University of Utah	Research assistant professor	University of Utah	Research assistant profe	Matthew Flat
Arvind Haran	MS	2014	IBM	Software Engineer: EDA Formal V	IBM	Software Engineer: EDA	Zvonimir Rak
Prashanth Nayak	MS	2014	NetApp	Software Developer	NetApp	Software Developer	Eric Eide
Scott Kuhl	Ph.D.	2009	Michigan Tech		Michigan Tech		William Thom
Margarita Bratkova	Ph.D.	2009	ImageMovers Digital				William Thom
Tina Ziemek	Ph.D.	2010	Stupid Fun Company				William Thom
Kritina Rand	Ph.D (Psychology	2014					William Thom
Mukund Raj	MS	2013	University of Utah	Ph.D. candidate			William Thom
Xing Lin	Ph.D.	2015	NetApp Advanced Technology Group				Robert Ricci
Weibin Sun	Ph.D.	2014	Google				Robert Ricci
Anil Kumar	MS	2016	Microsoft				Robert Ricci
Anil Mallapur	MS	2015	LinkedIn				Robert Ricci
Nikhil Mishrikoti	MS	2013	Cisco Systems				Robert Ricci
Srikanth Raju	MS	2013	Coverity				Robert Ricci
Yathindra Dev Naik	MS	2013	NetApp				Robert Ricci
Srikanth Chikkulapelly	MS	2011	Amazon AWS				Robert Ricci
Raghuveer Pullankandam	MS	2011	Adobe Systems				Robert Ricci
Matt Strum	MS BS/MS	2013	Amazon Silk Browser Team				Robert Ricci
Erik Anderson	Ph.D.	2013	EGI and University of Oregon	PostDoc			Chris Johnson
Joel Daniels	Ph.D.	2012	CD-Adapaco	PostDoc			Chris Johnson
Yaniv Gur	Ph.D.	2014	IBM Research	PostDoc			Chris Johnson
Brad Hollister	Ph.D.	2016	CalPoly	PostDoc			Chris Johnson
Jens Krueger	Ph.D.	2009	University of Duisberg-Essen, Ger	PostDoc			Chris Johnson
Paul Rosen	Ph.D.	2015	University of South Florida	PostDoc			Chris Johnson
Mavin Martin	MS	2014	Workday				Elaine Cohen
Fangxiang Jiao	Ph.D.	2012	VA Hospital, SLC, UT	Senior Research Staff			Elaine Cohen
Dafang Wang	Ph.D.	2012	Johns Hopkins University - Institu	Assistant Research Scientist			Elaine Cohen
Kristi Potter	Ph.D.	2010	University of Utah	Postdoc	University of Oregon	Manager, Research Supp	Elaine Cohen
Anastasia Mironova	MS	2009	Conoco-Phillips				Elaine Cohen
William Martin	Ph.D.	2012			Google	Research Scientist	Elaine Cohen
Tobias Martin	Ph.D.	2011			Eidgenossische Technische Hochschule, Zu	PostDoc	Elaine Cohen
Suraj Musuvathy	Ph.D.	2011			Siemens Research	Research Scientist	Elaine Cohen
Joel Daniels	Ph.D.	2009	NYU	PostDoc			Elaine Cohen
Sai Deng	MS	2015			Google		Elaine Cohen
Suqin Zeng	MS	2015			Google		Elaine Cohen
Geoffrey Draper	Ph.D.	2009	BYU Hawaii	Professor-tenure track			
Mike Steffen	Ph.D.	2009	Boing Corp				Mike Kirby
Dafang Wang	Ph.D.	2012	Johns Hopkins	PostDoc	Johns Hopkins	Researchers	Mike Kirby
Hanieh Mirzaee	Ph.D.	2012	Fraunhaufer MeVIs (Research Institute, Germany)		Fraunhaufer MeVIs (Research Institute, Germany)		Mike Kirby
Blake Nelson	Ph.D.	2012	Utah State University		Utah State University		Mike Kirby
Varun Shankar	Ph.D.	2014	University of Utah	Department of Math	University of Utah	Department of Math	Mike Kirby
James King	Ph.D.	2016	Google				Mike Kirby
Anshul Joshi	Ph.D.	2016	Cubiscan	R&D Engineer	Cubiscan	R&D Engineer	Thomas C. He
Linda DuHadway	Ph.D.	2016	Weber State University	Assistant Professor	Weber State University	Assistant Professor	Thomas C. He

Students Name	Degree		Year Completed	First Company	First Title/Role	Present Company	Present Title/Role	Advisor That
Protonu Basu	Ph.D.		2016	Postdoctoral Research Fellow	Lawrence Berkeley National Labo	Postdoctoral Research Fellow	Lawrence Berkeley Natic	Mary Hall
Saurav Muralidharan	Ph.D.		2016	NVIDIA	Research Scientist	NVIDIA	Research Scientist	Mary Hall
Anand Venkat	Ph.D.		2016	Intel	Research Scientist	Intel	Research Scientist	Mary Hall
Yu-Jung Lo	MS	Thesis	2015	Pinterest	Software Engineer	Pinterest	Software Engineer	Mary Hall
Axel Rivera	MS	Thesis	2014	Intel	Compiler Developer	Intel	Compiler Developer	Mary Hall
Shreyas Ramalingam	MS	Thesis	2012	AMD	Design Engineer 2			Mary Hall
Gabe Rudy	MS	Thesis	2010	Golden Helix	VP Product Development	Golden Helix	VP Product & Engineerin	Mary Hall
Suchit Maindola	MS	Project	2012	Cisco Systems	Software Engineer	Facebook	Software Engineer	Mary Hall
Prajakta Mane (ECE)	MS	Project	2015	VT iDirect	Software Engineer	VT iDirect	Software Engineer	Mary Hall
Amit Roy	MS	Project	2016	DrChrono	Software Engineer	DrChrono	Software Engineer	Mary Hall
Gagan Sachdev	MS	Project	2011	AMD	Design Engineer 2	ARM Inc.	Staff Design Engineer	Mary Hall
Ashequl Qadir	Ph.D.		2016	Philips Research	Research Scientist			Ellen Riloff
Ruihong Qadir	Ph.D.		2014	Stanford University	PostDoc	Texas A&M University	Assistant Professor, Com	Ellen Riloff
Siddharth Patwardhan	Ph.D.		2010	IBM T.J. Watson Research Center	Research Staff Member	Apple	NLP and Machine Learni	Ellen Riloff
David Price	MS		2009	PAWAR Systems Center Pacific				Ellen Riloff
Nathan Gilbert	MS		2014	Delcam	Software Engineer	Autodesk	Senior Software Enginee	Ellen Riloff
Lalindra De Silva	MS		2016	Department of Veterans Affairs S	Senior Software Engineer			Ellen Riloff
Yupeng Zhang	MS		2015	Amazon		Amazon	SDE	Feifei Li
Oscar Marshall	MS	BS/MS	2015	Startup		Unknown	SDE	Feifei Li
Mengyang Wang	MS		2015	Microsoft		Microsoft	SDE	Feifei Li
Klemen Simonic	MS		2015	Facebook		Facebook	SDE	Feifei Li
Natalee Ann Villa	MS		2015	Adobe		Adobe	Research Scientist	Feifei Li
Mingwang Tang	Ph.D.		2014			Uber	SDE	Feifei Li
Wangchao Le	Ph.D.		2013	Microsoft		Microsoft	SDE	Feifei Li
Jeffrey Jestes	Ph.D.		2013	Cerner		Cerner	Senior Software Enginee	Feifei Li
Chi Zhang	Ph.D.		2013	Walmat Lab		Walmat Lab	RSDE	Feifei Li
Cody Hansen	MS		2013	Disney Interactive		Disney Interactive	Software Engineer	Feifei Li
Chengxu Ding	MS		2013	Epic		Epic	Software Engineer	Feifei Li
Limou Wang	MS		2013	Turn		Yahoo	SDE	Feifei Li
Namita Mahtta	MS		2013	Goldman Sachs		University of Utah	Information Technology	Feifei Li
Yu Sun	Ph.D.		2007	U. South Florida	Associate Professor			John Hollerba
Josh de Bever	Ph.D.		2015	Stanford Cancer Imaging Training	Program			John Hollerba
Babak Hejrati	Ph.D. (Mechanica		2016	Harvard University Biodesign Lab	Postdoctoral Fellow			John Hollerba
Joshua Dawson	MS		2015	US Army	Flight Instructor			Miriah Meyer