Conceptual Mutation Testing for Student Programming Misconceptions

Siddhartha Prasad
Ben Greenman
Tim Nelson
Shriram Krishnamurthi

BROWN

CRA
Computing Research Association

Understanding CS Problems

Understanding CS Problems

Q. median

Understanding CS Problems

Q. median

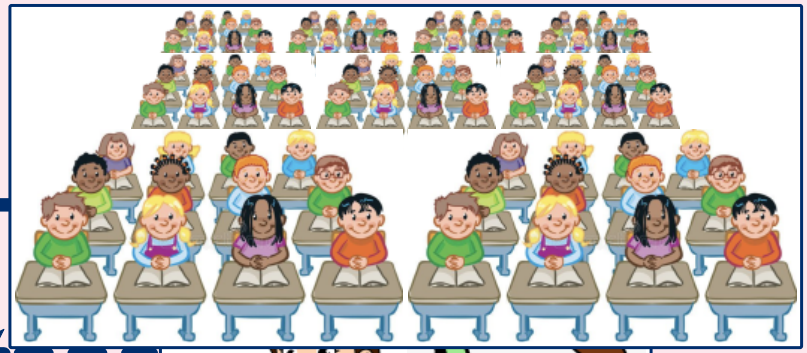
Understanding CS Problems

Q. median

feedback?

Understanding CS Problems

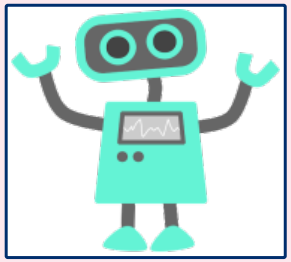
Q. median



feedback?

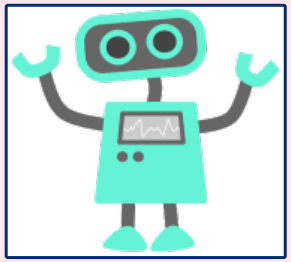
Understanding CS Problems

Q. median

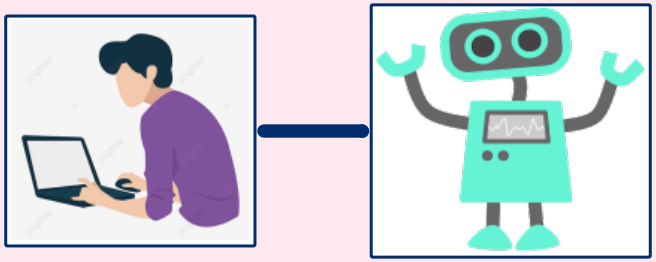


Understanding CS Problems

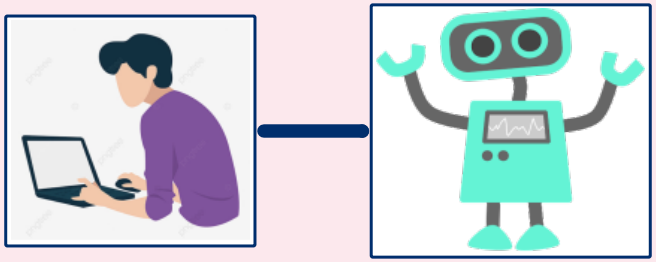
Q. median



How to communicate?

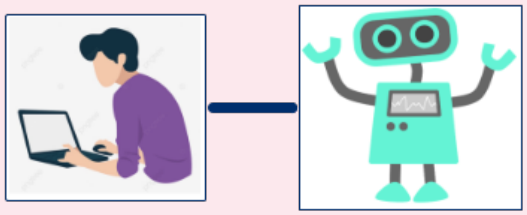


How to communicate?

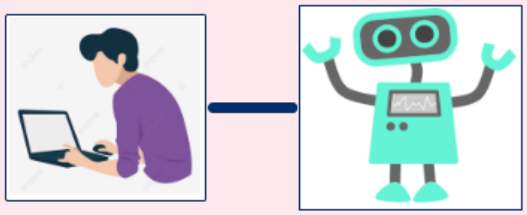


Test Cases

Q. median

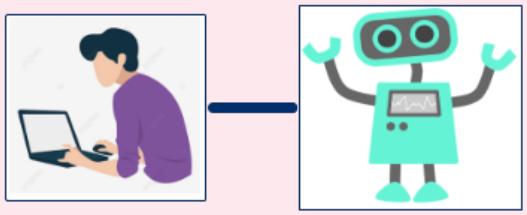


Q. median



```
median [1] is 1
median [1, 2, 3] is 3
median [3, 3, 3] is 3
```

Q. median



```
median [1] is 1
median [1, 2, 3] is 3
median [3, 3, 3] is 3
```

What's wrong with these tests?

View File median-tests.arr + Begin

```
include my-gdrive("median-code.arr")
# DO NOT CHANGE ANYTHING ABOVE THIS LINE

check:
  median([list: 1]) is 1
  median([list: 1, 2, 3]) is 3
  median([list: 3, 3, 3, 3]) is 3


end
```

View File median-tests.arr + Begin Implementation Run Stop

```
1  include my-gdrive("median-code.arr")
2  # DO NOT CHANGE ANYTHING ABOVE THIS LINE
3
4  check:
5    median([list: 1]) is 1
6    median([list: 1, 2, 3]) is 3
7    median([list: 3, 3, 3, 3]) is 3
8
9
10 end
```

median-tests.arr

INCORRECT CONSEQUENTLY, THOROUGHNESS IS UNKNOWN

These tests do not match the behavior described by the assignment:

definitions://:5:2-5:30

```
6  median([list: 1, 2, 3]) is 3
```

View File median-tests.arr + Begin

```
include my-gdrive("median-code.arr")
# DO NOT CHANGE ANYTHING ABOVE THIS LINE

check:
  median([list: 1]) is 1
  median([list: 1, 2, 3]) is 2
  median([list: 3, 3, 3, 3]) is 3

  #Shows that Median is not Mean
  median([list: 1, 1, 3]) is 1

  # Shows that Median is not Mode
  median([list: 1, 1, 3, 4, 4]) is 3
end
```

View File median-tests.arr + Begin Implementation Run Stop

```
include my-gdrive("median-code.arr")
# DO NOT CHANGE ANYTHING ABOVE THIS LINE

check:
  median([list: 1]) is 1
  median([list: 1, 2, 3]) is 2
  median([list: 3, 3, 3, 3]) is 3

  #Shows that Median is not Mean
  median([list: 1, 1, 3]) is 1

  # Shows that Median is not Mode
  median([list: 1, 1, 3, 4, 4]) is 3
end
```

median-tests.arr

VALID

These tests are valid and consistent with the assignment handout. They caught 2 of 4 sample buggy programs. Add more test cases to improve this test suite's thoroughness.

View File median-tests.arr + Begin Implementation Run Stop

```
include my-gdrive("median-code.arr")
# DO NOT CHANGE ANYTHING ABOVE THIS LINE

check:
  median([list: 1]) is 1
  median([list: 1, 2, 3]) is 2
  median([list: 3, 3, 3, 3]) is 3

  #Shows that Median is not Mean
  median([list: 1, 1, 3]) is 1

  # Shows that Median is not Mode
  median([list: 1, 1, 3, 4, 4]) is 3
end
```

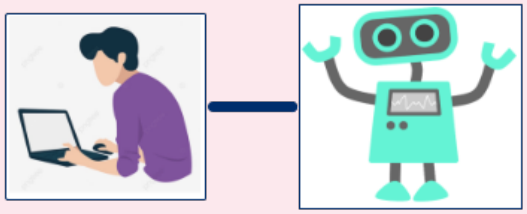
median-tests.arr

VALID

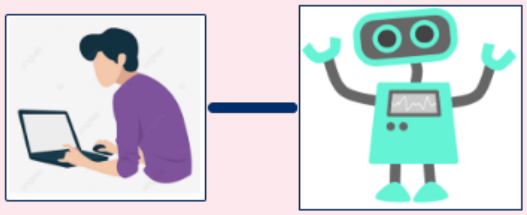
These tests are valid and consistent with the assignment handout. They caught 2 of 4 sample buggy programs. Add more test cases to improve this test suite's thoroughness.

What's wrong with these tests?

Q. median



Q. median



Tests must distinguish:

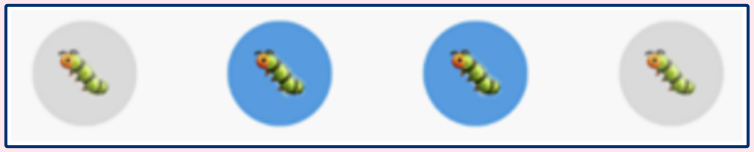
`median` vs. `mean`
`mode`
`middle ...`

Valid & Thorough

How to check thoroughess?

Buggy solutions

(mutation testing)



RQ. How to design buggies?

RQ. How to design buggies?

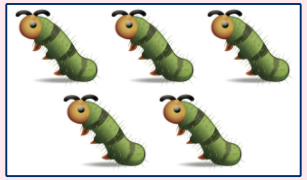
Need to discover misconceptions

Prior Work:

Expert-Driven

Today, a recipe:

Buggies from Data

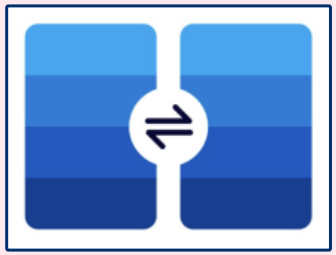


1. Design problem

1. Design problem

Running example:

Doc Diff



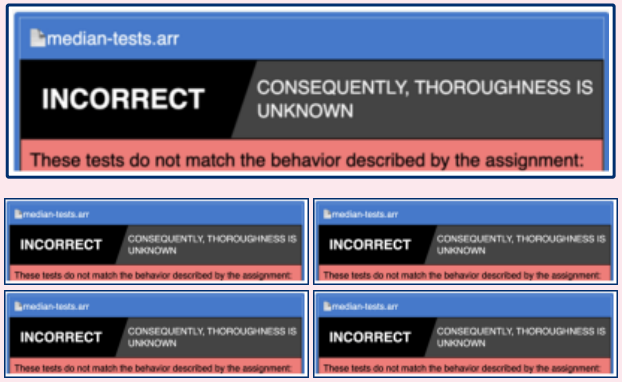
```
docdiff ['a'] ['A'] is 1
```

```
docdiff ['one', 'two'] ['one'] is 1/2
```

```
docdiff ['hello'] ['world] is 0
```

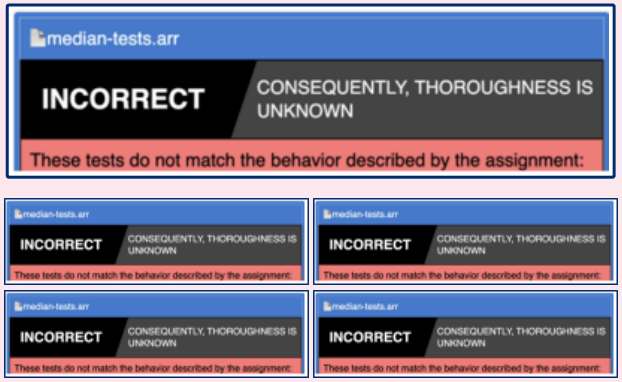
2. Collect invalid tests

2. Collect invalid tests



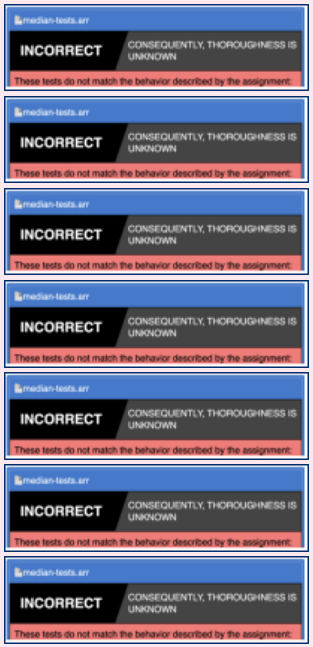
2. Collect invalid tests

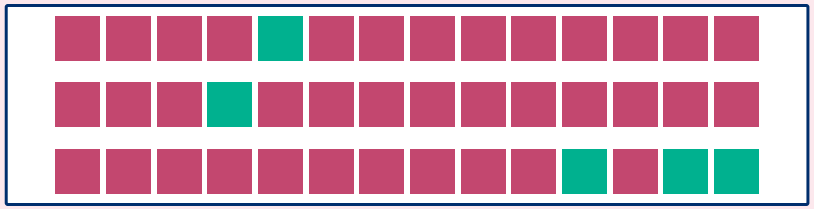
Doc Diff ==> 1,500 invalids in ~1 week



3. Cluster tests by feature vector

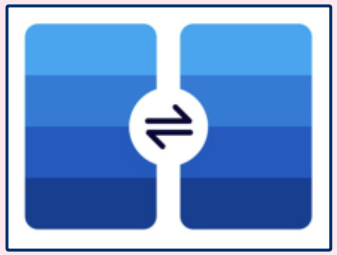
3. Cluster tests by feature vector





Feature vectors <== problem characteristics

Feature vectors <== problem characteristics



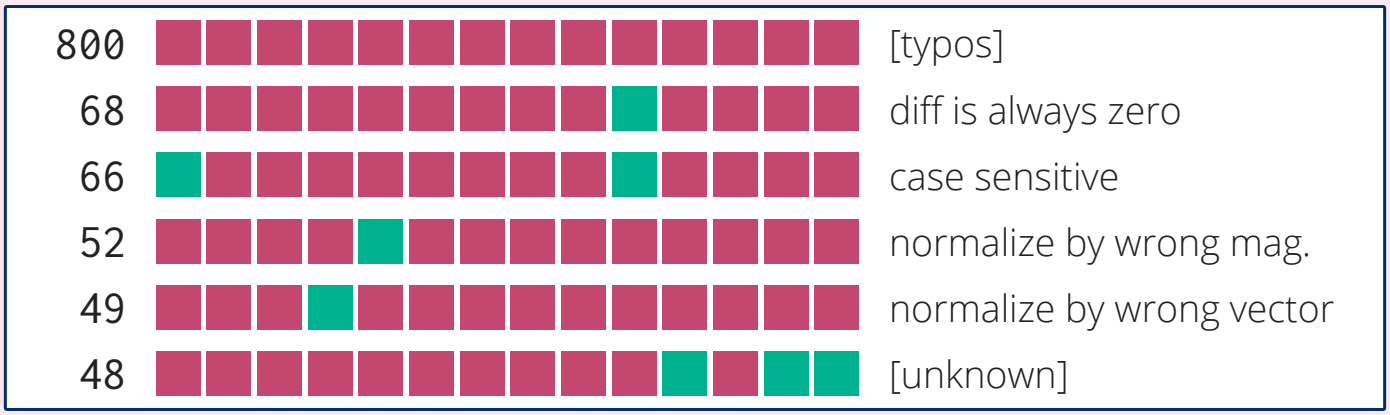
- Case-insensitive
- Words may repeat
- Diff may be a fraction

... [14 in total]

5. Sort clusters

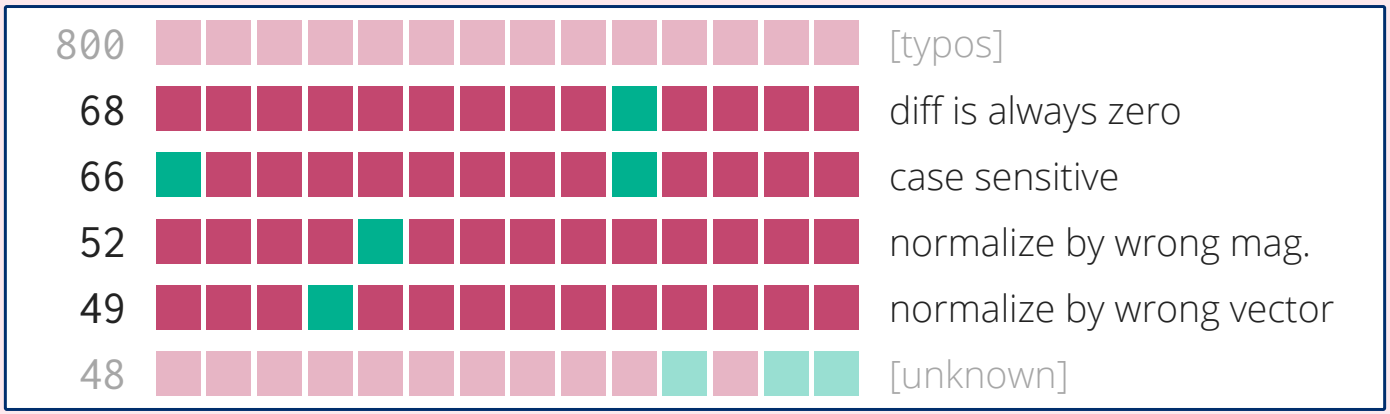
5. Sort clusters

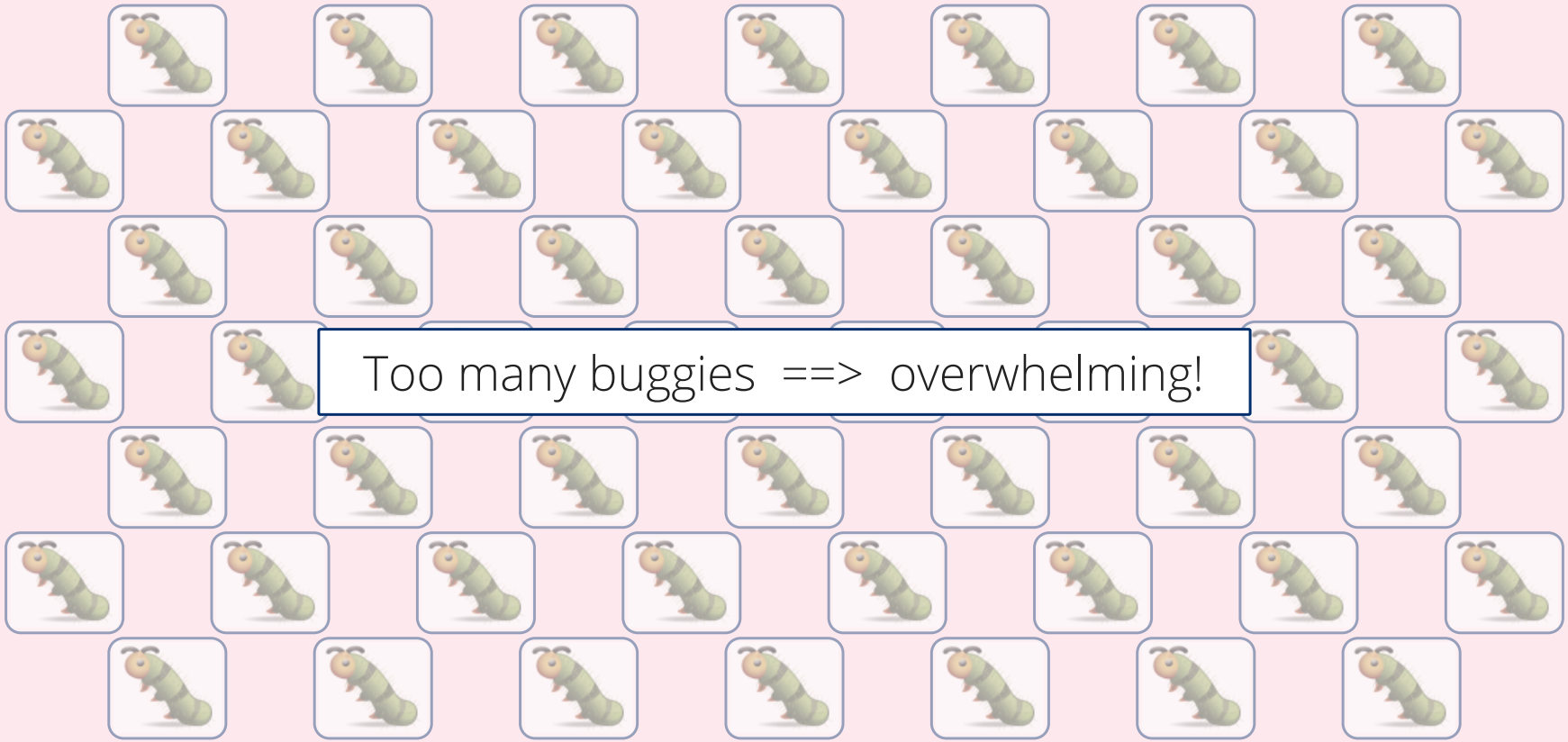
5. Sort clusters



5. Sort clusters

6. Make buggies




Too many buggies ==> overwhelming!

6. Make buggies

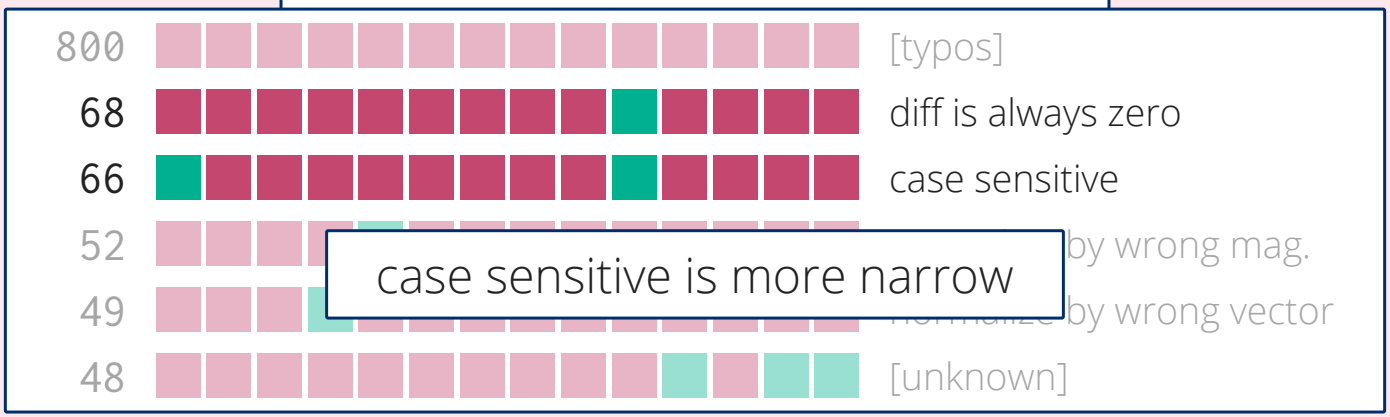
* Focus on 1-2
* Favor narrow characteristics
* Maximize subproblem coverage

6. Make buggies

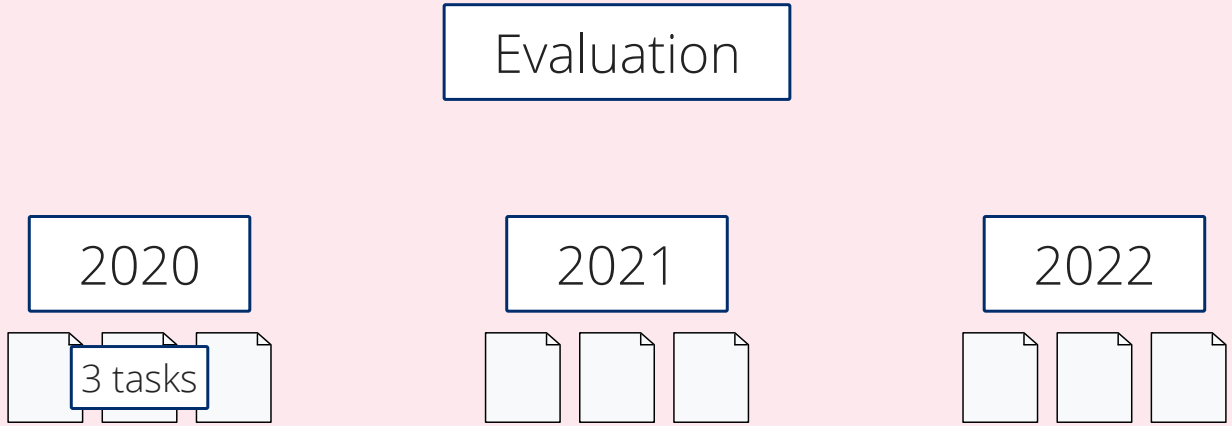
* Focus on 1-2
* Favor narrow characteristics
* Maximize subproblem coverage

6. Make buggies

* Focus on 1-2
* Favor narrow characteristics

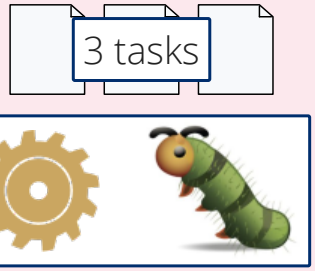


case sensitive is more narrow


Evaluation
2020
3 tasks
2021
2022

Evaluation

2020

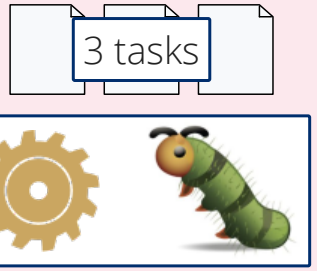


2021

2022

Evaluation

2020



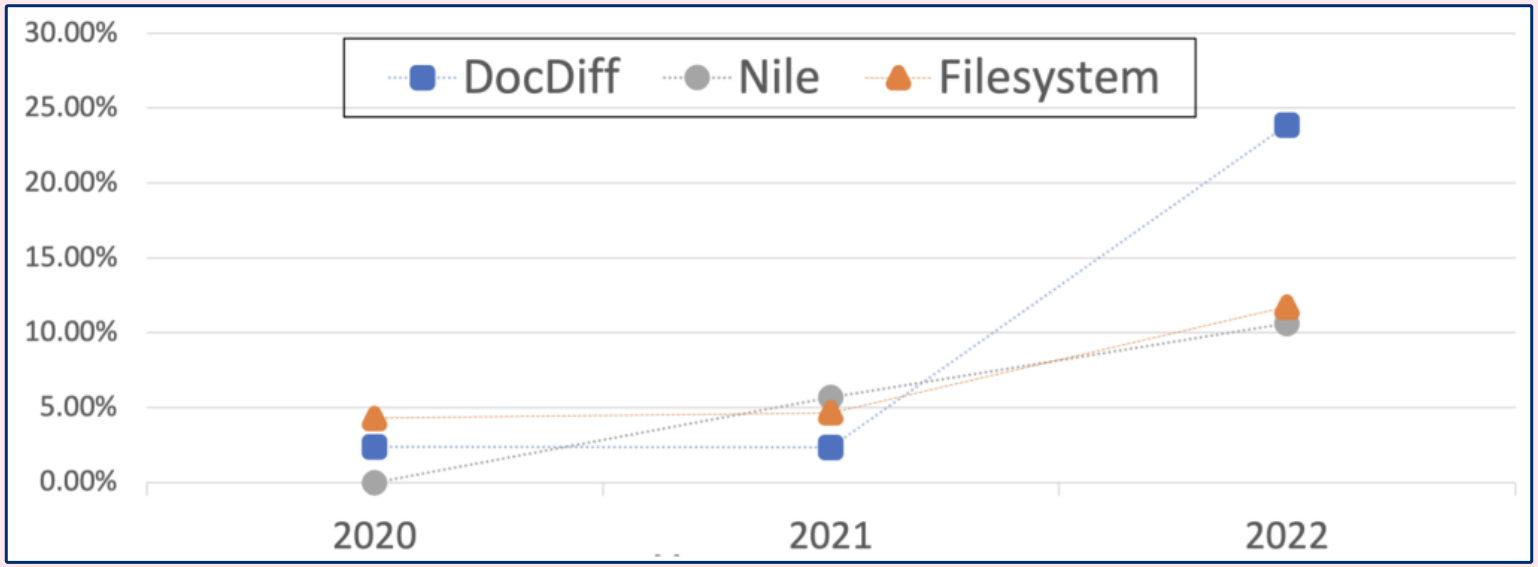
2021

2022

2020, 2021 ==> test
2022 ==> deploy

% explainable invalid tests

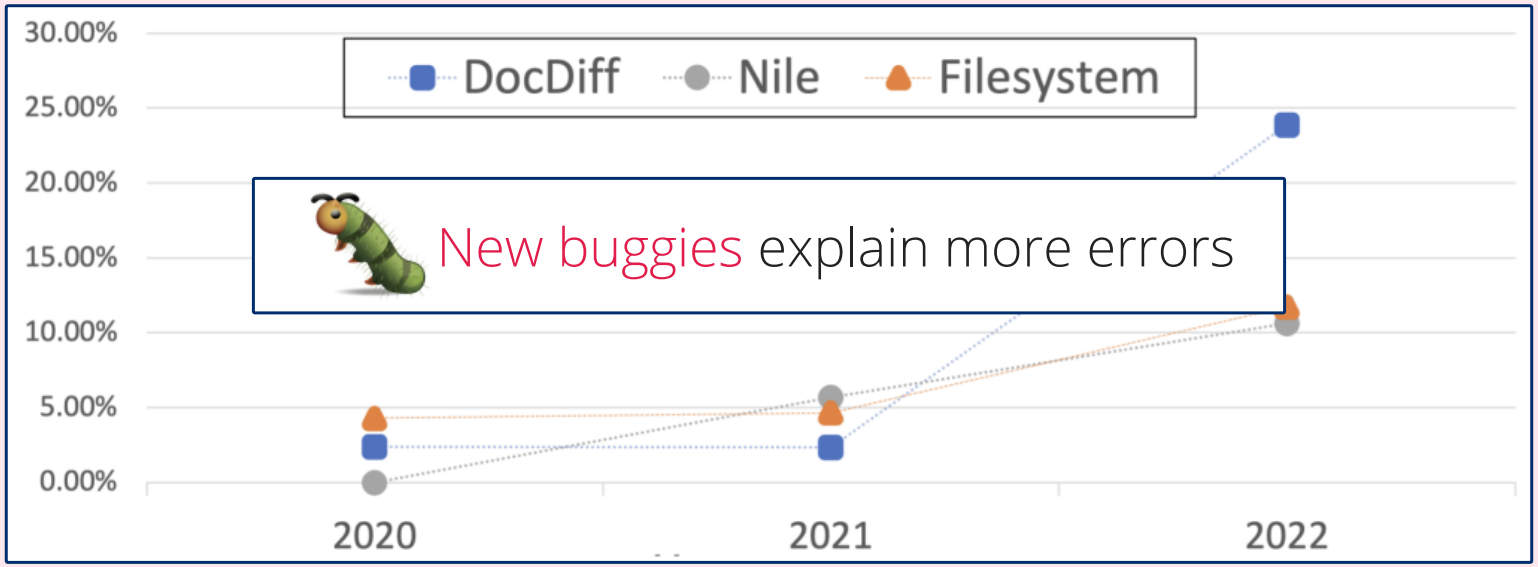
explainable = 1- or 2-



old buggies | old buggies | new buggies

% explainable invalid tests

explainable = 1- or 2-



old buggies | old buggies | new buggies

High Effect Sizes for 2022

Matchup	Problem	95% CI	p value
2022 vs 2020	DocDiff	[-0.75, -0.57]	1.35E-29
	Nile	[-0.55, -0.26]	9.07E-14
	FileSys	[-0.35, -0.21]	2.35E-10
2022 vs 2021	DocDiff	[-0.70, -0.51]	6.87E-29
	Nile	[-0.27, -0.07]	1.82E-3
	FileSys	[-0.33, -0.19]	2.32E-9
2021 vs 2020	DocDiff	[-0.07, 0.08]	4.60E-1
	Nile	[-0.39, -0.13]	1.15E-17
	FileSys	[-0.06, 0.03]	2.52E-1

Weeks of Data > Years of Tuning

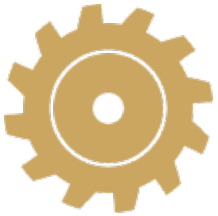
Promising approach for new problems

II

Recipe to uncover misconceptions

semi-automatic

II



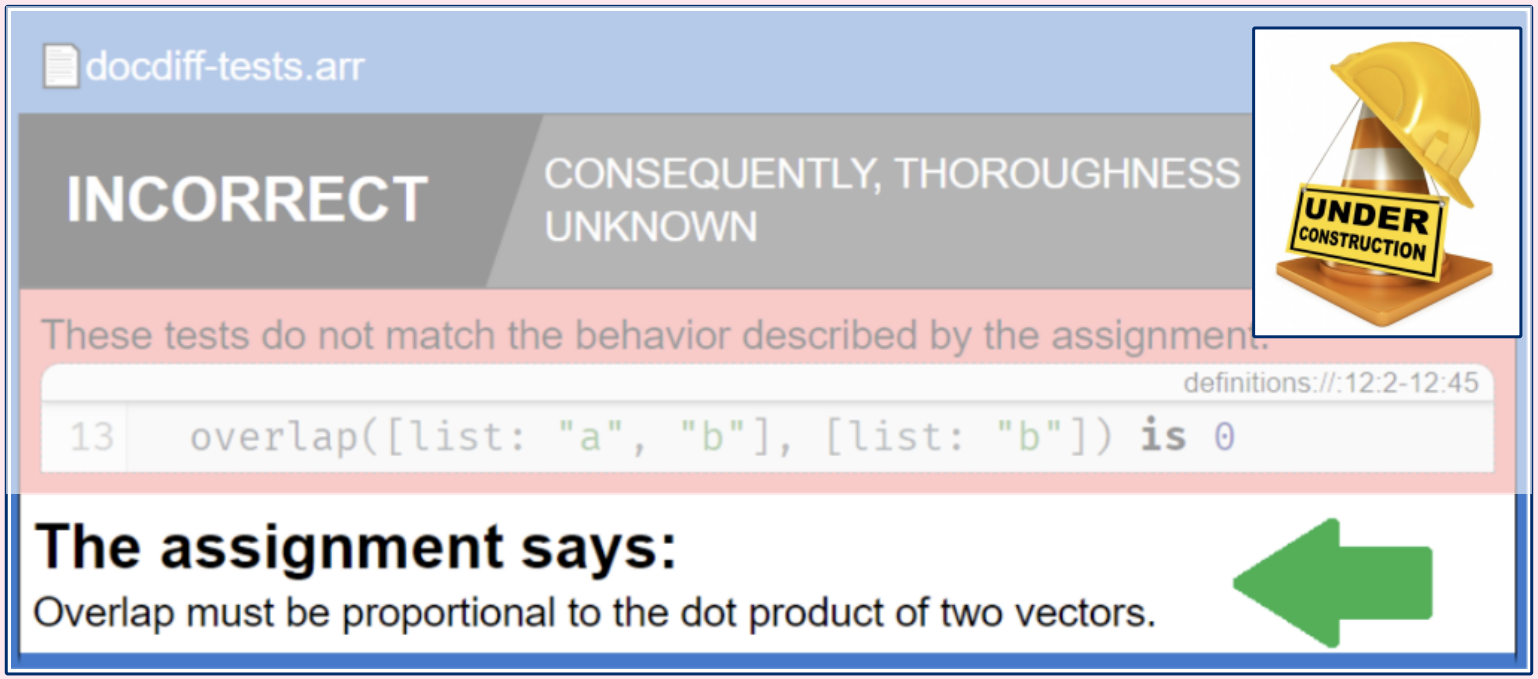
Recipe to uncover misconceptions

semi-automatic

Data ==> better teaching

What's next? Hinting

What's next? Hinting



Deep Goal:

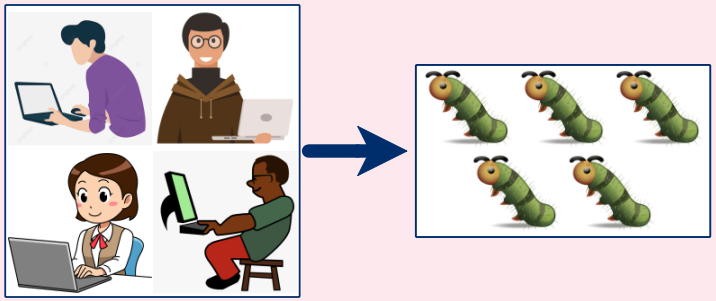
Rigorous methods for CS Ed research

THE UNIVERSITY OF UTAH

BROWN

Let's talk!

1. design problem
2. identify characteristics
3. collect invalid tests
4. cluster by feature vector
5. analyze top clusters
6. select buggies



Future

Data collection is a bottleneck

~1 semester ramp-up

+70% typos! How to reduce?

D4 / Data Druid

Table 8 Our 2022 chaffs gave 1-m/2-m outcomes significantly more often than prior chaffs. The 2021 vs. 2020 results are similar except for Nile, which used D4 in 2021.

Matchup	Assignment	p value	Z score	Effect Size [95% CI] (Cohen's D)
2022 vs 2020	DocDiff	1.35E-29	-11.24	0.66 [-0.75, -0.57]
	Nile	9.07E-14	-7.36	-0.41 [-0.55, -0.26]
	Filesystem	2.35E-10	-6.22	-0.28 [-0.35, -0.21]
2022 vs 2021	DocDiff	6.87E-29	-11.09	-0.61 [-0.70, -0.51]
	Nile	1.82E-03	-2.91	-0.17 [-0.27, -0.07]
	Filesystem	2.32E-09	-5.86	-0.26 [-0.33, -0.19]
2021 vs 2020	DocDiff	4.60E-01	0.1	0 [-0.07, 0.08]
	Nile	1.15E-17	-8.48	-0.26 [-0.39, -0.13]
	Filesystem	2.52E-01	-0.67	-0.02 [-0.06, 0.03]