CHOPSAT: Feasible Region Properties

Amelia Lessen
University of Utah

UUCS-23-002

School of Computing
University of Utah
Salt Lake City, UT 84112 USA

12 April 2023

Abstract

Given a logical sentence (or formula) from propositional calculus, the Satisfiability Problem (SAT) is to determine if there is a truth assignment to the logical variables that makes the sentence true. If so, the sentence is called satisfiable; if not, then it is unsatisfiable. The best known algorithm to solve this problem is NP-complete (requires polynomial time on a nondeterminisitic Turing machine). Recently, Henderson et al. proposed a geometric approach, called Chop-SAT, for solving SAT. The method produces a convex polytope feasible region, and the goals here are to explore properties of the feasible region, including:

- projected points found by using linear programming to obtain extreme points,
- properties of the projected points, and
- nueural net models of the feasible region.

In order to determine the success of this approach, we will:

1. Attempt to distinguish satisfiable from unsatisfiable sentences based on the maximum distance of any projected point,
2. Study the mean value of the projected points and points in the feasible region to see if there is a relation to the logical variable probabilities, and
3. Determine if a neural network approximation to a characteristic function for the feasible region is useful.

CHOPSAT: FEASIBLE REGION PROPERTIES

by

Amelia Lessen

A Senior Thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Bachelor of Science

Department of Computer Science

The University of Utah

Approved:

____
Thomas Henderson
Thesis Faculty Supervisor

____
H. James de St. Germain
Director, Undergraduate Studies

____
Mary Hall
Director, School of Computing

ABSTRACT

Given a logical sentence (or formula) from propositional calculus, the *Satisfiability Problem (SAT)* is to determine if there is a truth assignment to the logical variables that makes the sentence true. If so, the sentence is called satisfiable; if not, then it is unsatisfiable. The best known algorithm to solve this problem is NP-complete (requires polynomial time on a nondeterminisitic Turing machine) [8]. Recently, Henderson et al. proposed a geometric approach, called *Chop-SAT*, for solving SAT (see [5]). The method produces a convex polytope feasible region, and the goals here are to explore properties of the feasible region, including:

- projected points found by using linear programming to obtain extreme points,
- properties of the projected points, and
- neural net models of the feasible region.

In order to determine the success of this approach, we will:

1. Attempt to distinguish satisfiable from unsatisfiable sentences based on the maximum distance of any projected point,
2. Study the mean value of the projected points and points in the feasible region to see if there is a relation to the logical variable probabilities, and
3. Determine if a neural network approximation to a characteristic function for the feasible region is useful.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The satisfiability problem (SAT), is an NP-complete problem that represents both theoretical and practical importance in the field of computer science. From a theoretical perspective, studying a NP-complete problem may result in demonstrating that P is unequal to NP. From a practical perspective, showing P is unequal to NP would result in less time searching for more efficient algorithms to NP-complete problems that do not exist [8].

In the geometric approach to SAT, the n-dimensional hypercube, H_n, represents the solution space for a SAT problem. Parts of H_n are removed based on a geometric interpretation of the logical sentence. This results in a convex feasible region, $\mathcal{F}$, and a solution to the SAT problem is sought in $\mathcal{F}$. This formulation is being studied to determine if it offers an efficient SAT solution or good heuristics for solving SAT.

The thesis of this work is that certain properties of $\mathcal{F}$ allow better insight into the existence of a SAT solution. Moreover, due to the continuous nature of H_n, it may also be possible to estimate atom probabilities over all solutions.

We begin with an overview of the visualization tools we created for feasible region analysis. These tools act as a simple guide for ensuring the correctness of a solution. They allow visualization of the solution using projections into 3D space.

We then explore linear programming solutions which minimize or maximize the projection of the feasible region onto selected vectors. This method aims to provide insight to whether projections can accurately determine satisfiability of a given set of logical sentences.

Next, we examine properties of the feasible region. Atom probabilities may provide the ability to determine model probabilities. We test this idea through the use of projections and geometric centers. We explore the accuracy of using the mean of projected points as atom probability estimates. We next calculate the analytic centers of different feasible regions on a variety of knowledge bases. We compare these results to known atom probabilities. We also calculate the P-center on these feasible regions and compare these results to our known atom probabilities. We examine which method most closely approximates the known atom probabilities and whether these methods assist in providing solutions for probabilistic satisfiability (PSAT).

Finally, we determine if a neural net model can be trained to accurately determine satisfiability of a sentence. We test whether a neural net model can be used effectively to aid in finding a solution for SAT.

We perform these analyses on a variety of knowledge bases to examine the robustness of our results. A thorough data analysis of our results is provided in order to determine the accuracy of the findings.

CHAPTER 2

BACKGROUND

2.0.1 *NP-Completeness*

For an algorithm V, a verifier for a language A is defined as:

$$A = w \mid V \text{ accepts } w, c \text{ for some string } c$$

We measure the time of a verifier only in terms of the length of w, so a polynomial time verifier runs in polynomial time in the length of w. NP problems have polynomial time verifiers. NP-completeness refers to the discovery by Stephen Cook and Leonid Levin where certain problems in NP have an individual complexity related to that of the entire class. This means, if there exists a polynomial time algorithm for any of these problems, all problems in NP would be polynomial time solvable. This set of problems is called NP-complete [8].

2.0.2 *SAT*

Given a logical sentence (or formula) from propositional calculus, the Satisfiability Problem (SAT) is to determine if there is a truth assignment to the logical variables that makes the sentence true. If so, the sentence is called satisfiable; if not, then it is unsatisfiable. The best known algorithm to solve this problem is NP-complete (requires polynomial time on a nondeterminisitic Turing machine) [8].

2.0.3 *PSAT*

Given n logical variables (or atoms) a model (or complete conjunction) is an assignment of 0 (false) or 1 (true) to each atom. There are 2^n models. These models can be represented as n-tuples in n-dimensional space, and correspond to the corners of H_n, the n-D hypercube.

The meaning of this is that the i^{th} axis corresponds to the values which can be assigned to the i^{th} variable.

Given any point in H_n, that point can be considered as a set of probabilities for the atoms. This allows consideration of a probabilistic version of SAT called *Probabilistic SAT (PSAT)* (see [3] [7] for a detailed introduction and analysis of the complexity of the problem). PSAT is defined as follows; given a logical sentence in Conjunctive Normal Form (CNF), and a probability, p_i, associated with each conjunct, C_i, find a function, $\pi : \Omega \rightarrow [0, 1]$, where Ω is the set of all complete conjunctions, and all of the following are true:

$$0 \leq \pi(\omega_k) \leq 1$$

$$\sum_{i=0}^{2^n-1} \pi(\omega_i) = 1$$

$$p_i = \sum_{\omega_k \models C_i} \pi(\omega_k)$$

2.0.4 *Chop-SAT*

Chop-SAT is the proposed method for solving SAT (see [5]), and it produces a feasible region as follows:

- First, a knowledge base (KB) is defined as a CNF sentence, and each conjunct is given a probability (for a SAT problem, this will be 1 for each conjunct).
- Next, the *Chop-SAT* method is used to produce a set of hyperplanes such that the intersection of their non-negative half-spaces determines the feasible region.

The method follows from the observation that every conjunct is a disjunction, and the disjunction is made false by assigning a falsifying truth value to each of its literals. Then the set of models which do not satisfy this conjunct has those assignments and all possible assignments to the literals not appearing in the conjunct. This turns out to be a sub-hypercube which can be separated (chopped) from H_n by a hyperplane.

The feasible region represents the solution space for the KB. If any of the original hypercube corners are contained in the feasible region, the KB is satisfiable. Figure 2.1 displays a satisfiable KB, KBS_3. KB_3 consists of the clauses [-1 -2 -3], [-1 -2 3], [-1 2 -3], [-1 2 3], [1 -2 -3], and [1 -2 3]. Each conjunct is represented by the index of the atom and a negative value if the atom is negated. If no original corners remain, the problem is unsatisfiable. Figure 2.2 displays the maximal feasible region of unsatisfiable KB.

Another important observation to make is that every corner of H_n is $\sqrt{n}/2$ distant from the center of H_n. In fact, every satisfiable sentence results in a feasible region which contains an H_n corner; thus, some point in the feasible region is $\sqrt{n}/2$ distant from the center of H_n, and it can be determined that the sentence is satisfiable.

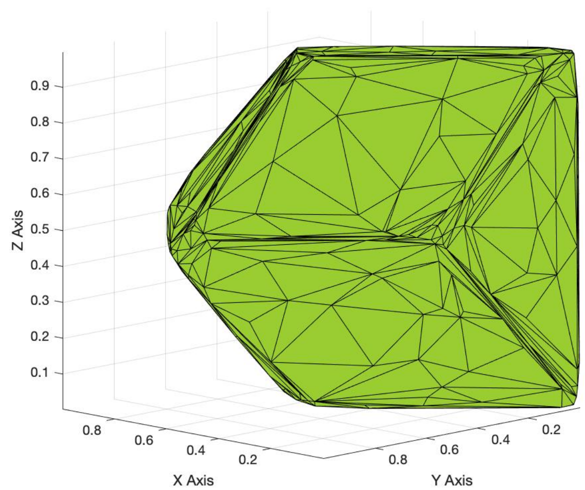

Figure 2.1: The Feasible Region for a Satisfiable Sentence in 3D.

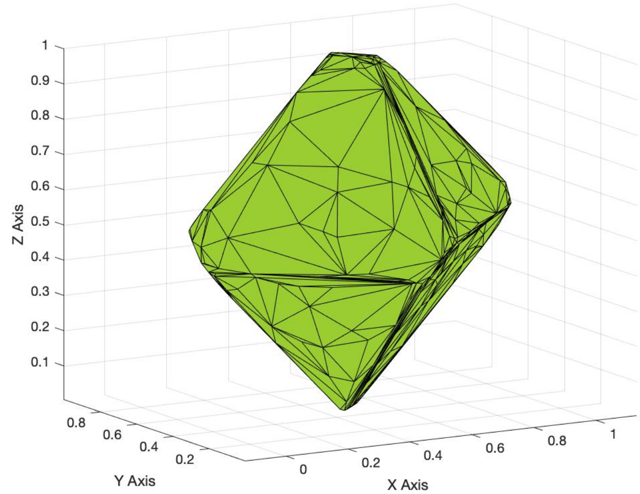

Figure 2.2: The Feasible Region for I_3, the Unsatisfiable Sentence in 3D with maximal volume.

Consider the unsatisfiable KB over n atoms defined as the conjunction of all disjunctions with n literals. This geometrically corresponds to chopping off each corner individually. The feasible region for this has a special geometry whose vertexes are the centers of all H_2 sub-hypercubes contained in H_n. The 3-D instance has an octahedron as the feasible region (see Figure 2.2). We refer to the feasible region of the n-D version of this as I_n. Also, every feasible region arising from an unsatisfiable sentence in n-D is contained in I_n.

CHAPTER 3

PROJECTIONS

Certain properties of the convex feasible region produced in ChopSAT, $\mathcal{F}$, may provide a more effective method for determining a SAT solution. The feasible region contains extreme points that, when found, could prove useful in determining the satisfiability of a sentence.

This chapter examines the usefulness of projection points in determining the satisfiability of a knowledge base. Projection points are calculated using:

$$\min_{x} f^T x \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

This corresponds to finding the point in the feasible region which projects to the minimal value on axis $f^T x$. The point x is a linprog solution.

We begin this method by calculating the projection points on fixed axes. Using the basis vectors as the fixed axes. We then calculate the linprog solution for axes in random directions. For this method, a random n-D direction vector is generated, and the linprog solution is found.

These linprog solutions are determined for both general and independent knowledge bases containing from 10 to 20 atoms. In independent knowledge bases, atoms are guaranteed to be independent. That is, for two atoms, a_1 and a_2, $P(a_1) * P(a_2) = P(a_1 \wedge a_2)$. In a general knowledge base, this equation is not guaranteed to be true.

To determine the usefulness of the projection point method, we examine how many linprog solutions must be found before the knowledge base can be declared satisfiable. We declare a knowledge base satisfiable when a linprog solution is found for which the distance from the center of a hypercube to this point is greater than a pre-determined threshold. In the maximal feasible region in an unsatisfiable KB, no point is farther than $\sqrt{n-2}/2$ from the center of H_n. This means, if a point with a distance greater than $\sqrt{n-2}/2$ is discovered, the sentence is satisfiable.

We test the projection point method on KBs with a small number of atoms (10 to 20 atoms) and KBs with a medium number of atoms (50 atoms). However, the method was found too slow to use on KBs with a large number of atoms (> 50 atoms).

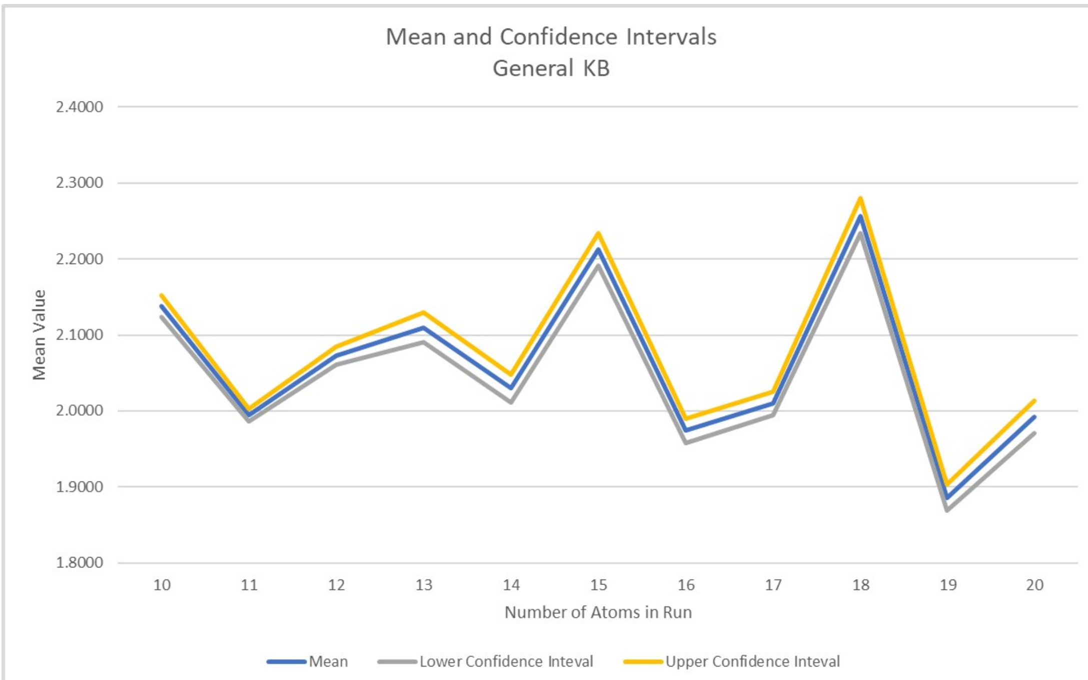

Figure 3.1: Plot of the average number of projections on set axes on general KBs, from 10 to 20 atoms.

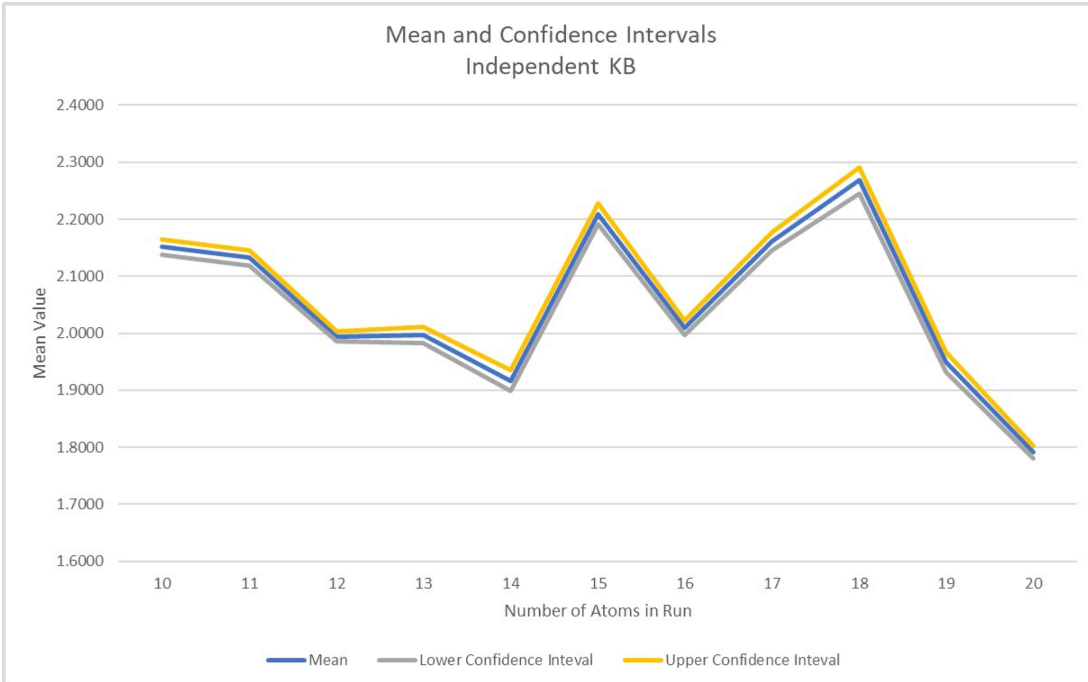

Figure 3.2: Plot of the average number of projections on set axes on independent KBs, from 10 to 20 atoms.

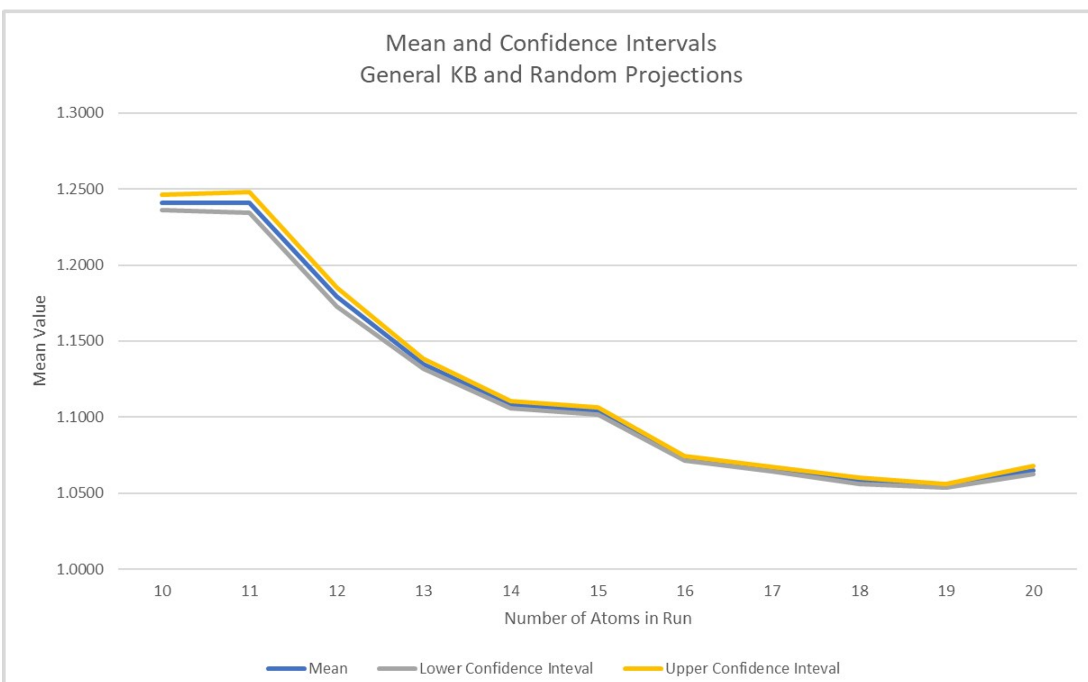

Figure 3.3: Plot of the average number of projections on random axes on general KBs, from 10 to 20 atoms.

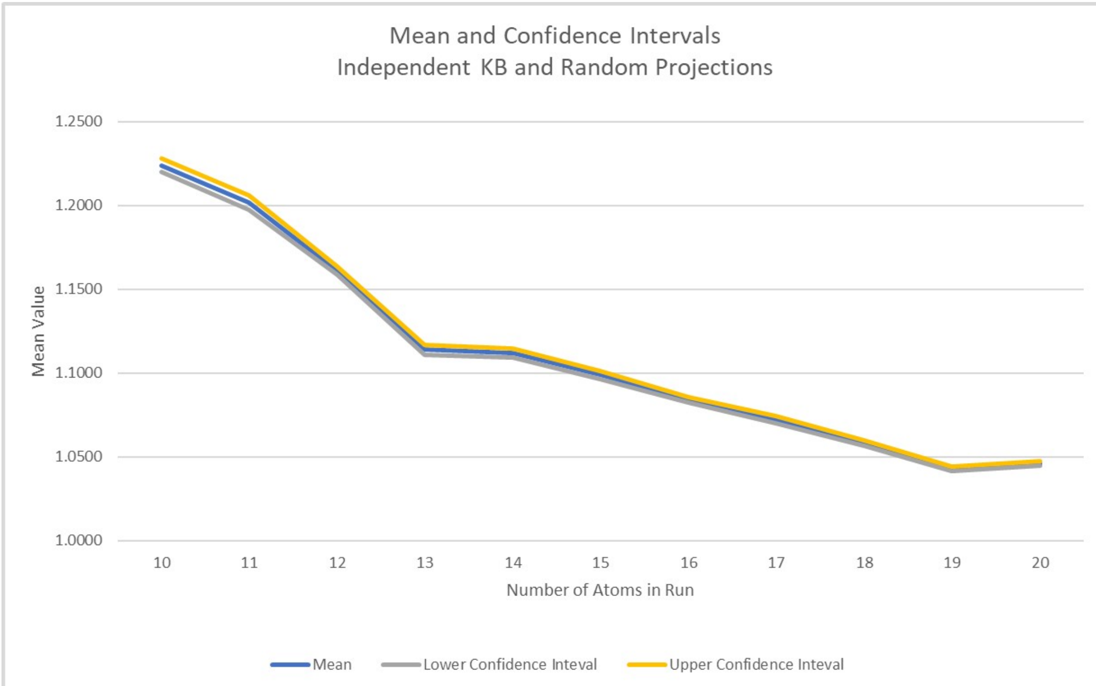

Figure 3.4: Plot of the average number of projections on random axes on independent KBs, from 10 to 20 atoms.

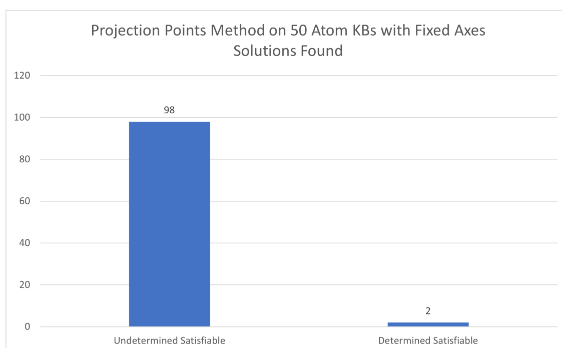

Figure 3.5: Graph of the projection point method on KBs where no solution was found vs KBs where a solution was found for 50 atom KBs on fixed axes.

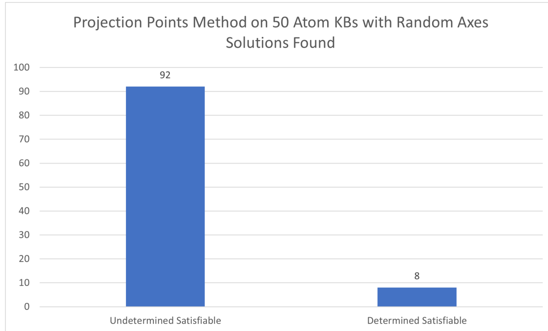

Figure 3.6: Graph of the projection point method on KBs where no solution was found vs KBs where a solution was found for 50 atom KBs on random axes.

This method worked well on knowledge bases with a small number of atoms. As shown in Figures 3.1, and 3.2, the fixed axes projection method performed very well, usually needing one or two projections to declare the knowledge base satisfiable. Figures 3.3 and 3.4 show similar results for the random axes projection method.

However, when tested on medium sized KBs, this method performed very poorly and was slow. As shown in Figures 3.5 and 3.6, the projection method was only able to successfully analyze 2 and 8, respectively, of the 100 knowledge bases.

CHAPTER 4

CENTERS

The continuous nature of H_n may allow for the atom probabilities over all solutions to be accurately estimated. In a KB with the clauses: $((a_1 \vee a_2 \vee a_3 \vee a_4) \wedge (\neg a_1 \vee \neg a_2) \wedge (\neg a_1 \vee \neg a_3) \wedge (\neg a_1 \vee \neg a_4) \wedge (\neg a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_4) \wedge (\neg a_3 \vee \neg a_4))$, the solutions are: [0, 0, 0, 1], [0, 0, 1, 0], [0, 1, 0, 0], [1, 0, 0, 0]. Each atom is true in 1/4 of the solutions, giving each atom a probability of 0.25. When this KB is analyzed to find atom probabilities, the result is [0.2500 0.2500 0.2500 0.2500].

Atom probabilities may be useful in a decision making process. If we assume atom independence, then the probabilities of the complete conjunctions can be determined. For example, in a KB with 3 atoms with the projection point probabilities of [0.3, 0.2, 0.7], then the probability for the most likely truth assignment, [0, 0, 1] is (1-0.3) * (1-0.2) * 0.7 = .392. This means that all the other truth assignments combined have an approximate probability of .608. If we look at the least likely truth assignment, [1, 1, 0], it has an estimated probability of 0.3 * 0.2 * (1 - 0.7) = 0.018. When performing probabilistic decision making, it may suffice to assign non-zero probabilities to only a small number of the most probable models. In this way, a good approximation may be found for any logical sentence of n variables.

Next, we examine methods for approximating atom probabilities. We will calculate different centers of the feasible region to determine if these centers are adequate approximations of the atom probabilities. We use three different atom probability approximation calculations in our approach.

First, we use the mean of linprog solutions as atom probability estimates. As discussed in depth in Chapter 3, linprog solutions are extreme positions in the feasible region, determined by selected axis. We use the mean value of these points and compare this value to the actual atom probabilities. In the second method, we calculate the analytic center of the feasible region and compare this point to the true atom probability vector.

The analytic center of the feasible region is defined as:

$$\overline{y} = max_{\overline{y}} \prod \overline{a_i y}$$

where $y \in \mathcal{F}$, $\overline{a_i}$ is the i^{th} hyperplane coefficient vector, and $\overline{y}$ is y with a final homogeneous coefficient of 1 [4].

Finally, we calculate the p-center of the feasible region and determine if this center can accurately approximate the atom probabilities of the given KB. The p-center of the feasible region is defined as follows: for each hyperplane in the set of linear inequalities, find a point, p, that minimizes the distance to each hyperplane from that point.

Each method was tested on one hundred satisfiable knowledge bases with twenty atoms, each with known atom probabilities. The results of each method are then compared to the known probabilities to determine if correlations or accurate approximations exist. These experiments are then repeated on one hundred satisfiable knowledge bases with fifty atoms each.

Euclidean distance is a standard measurement used to compare the closeness of two vectors, which made it an appropriate measure for determining the adequacy of a center estimation. Figures 4.1 and 4.4 show the Euclidean distance between the atom probability vectors and the linprog solution vectors. Figures 4.2 and 4.5 display the Euclidean distance between the atom probability vectors and the analytic center vectors. Figures 4.3 and 4.6 display the Euclidean distance between the atom probability vectors and the p-center vectors.

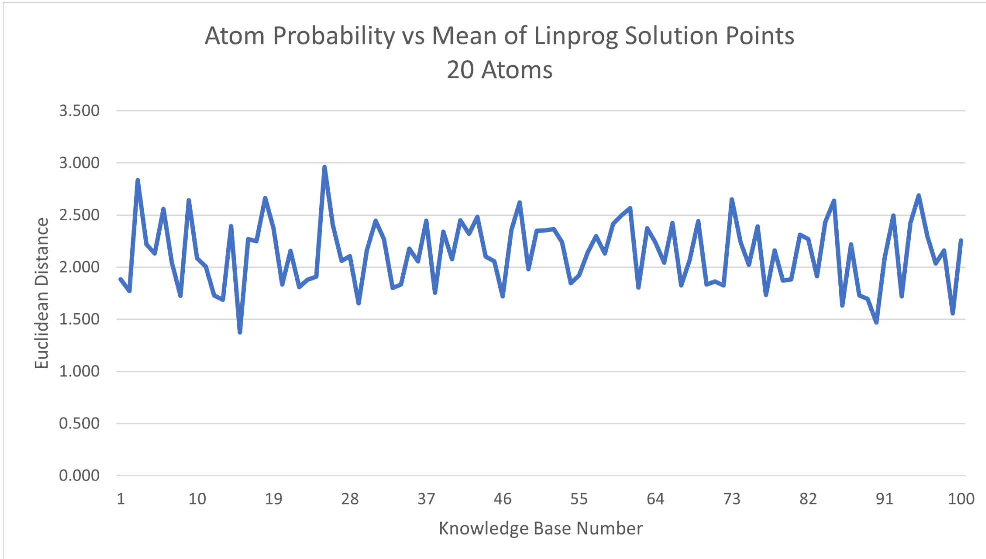

Figure 4.1: Plot of the Euclidean distance between the atom probabilities and the mean of the projection points on satisfiable, 20 atom KBs.

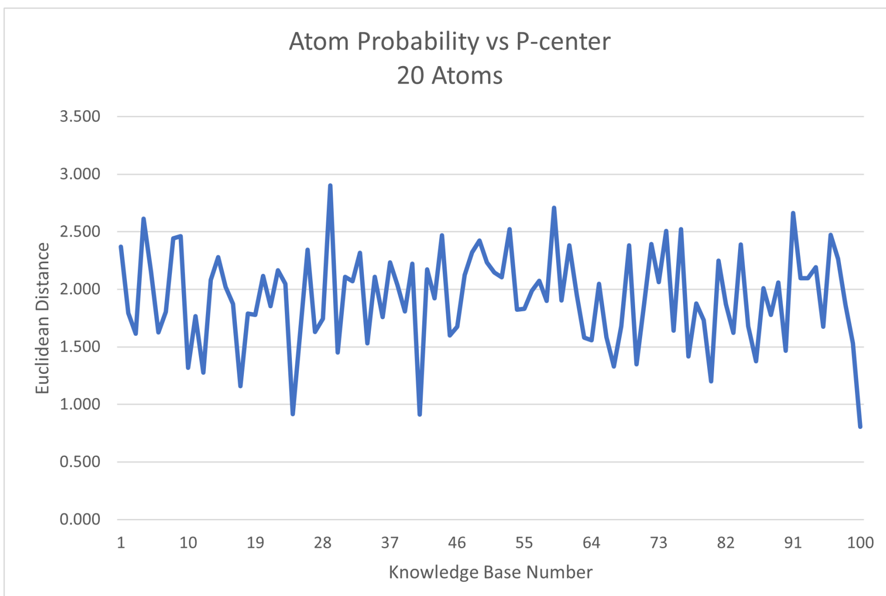

Figure 4.2: Plot of the Euclidean distance between the atom probabilities and the P-centers on satisfiable, 20 atom KBs.

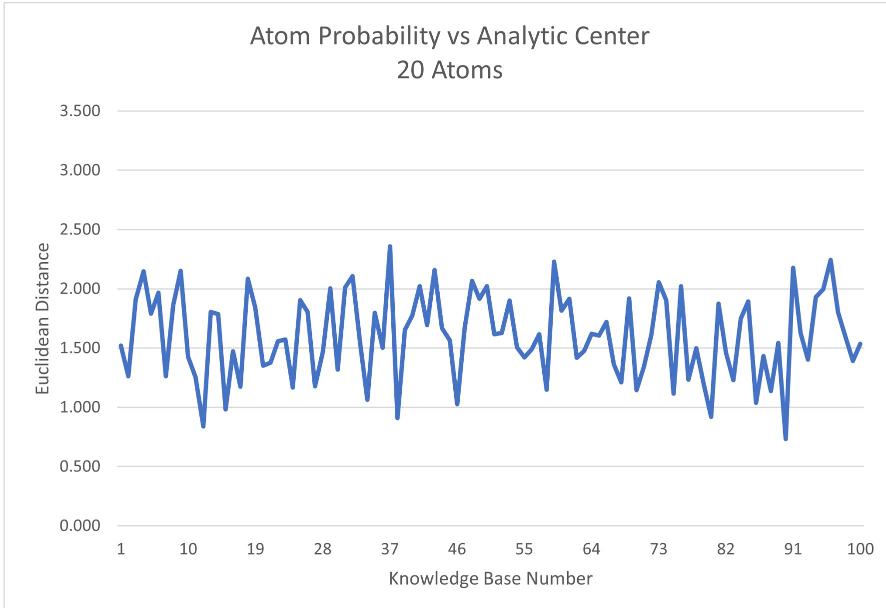

Figure 4.3: Plot of the Euclidean distance between the atom probabilities and the analytic centers on satisfiable, 20 atom KBs.

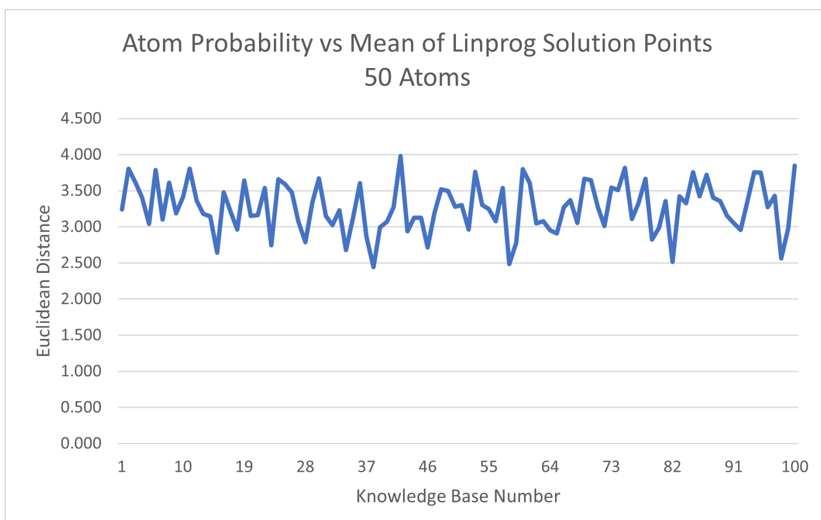

Figure 4.4: Plot of the Euclidean distance between the atom probabilities and the mean of the projection points on satisfiable, 50 atom KBs.

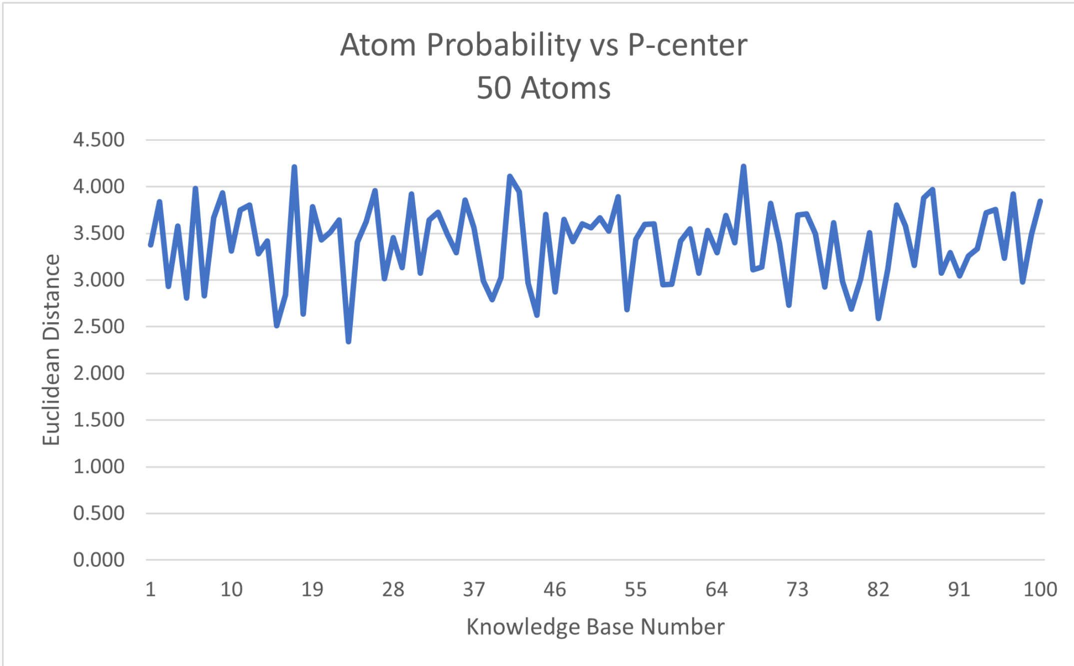

Figure 4.5: Plot of the Euclidean distance between the atom probabilities and the P-centers on satisfiable, 50 atom KBs.

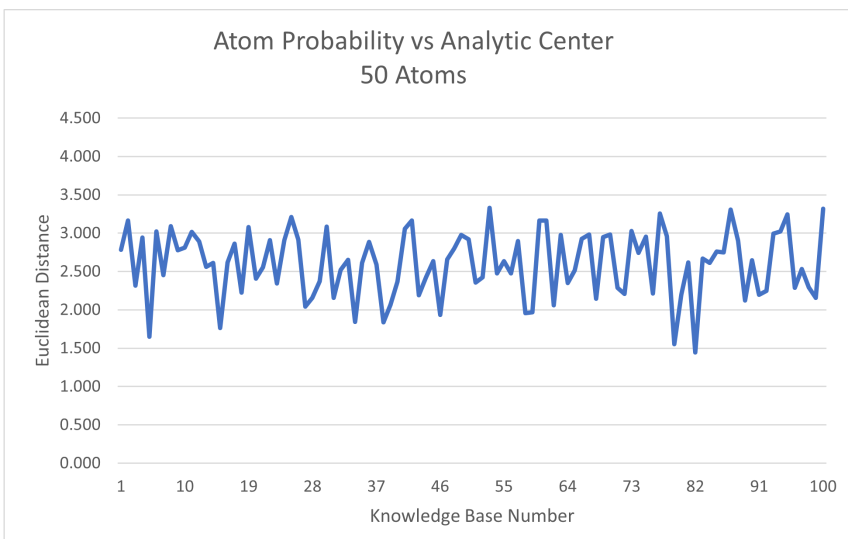

Figure 4.6: Plot of the Euclidean distance between the atom probabilities and the analytic centers on satisfiable, 50 atom KBs.

As shown in the above figures, the analytic center method generates results nearest to the true atom probabilities. In Table 4.1 below, the mean Euclidean distances for knowledge bases with 20 and 50 atoms, respectively, are given. The analytic center presents the closest estimate for atom probabilities, with P-center and projection points too inaccurate to provide a helpful heuristic.

	Number of Atoms	
Method	20	50
Analytic	1.603	2.601
P-center	1.933	3.392
Mean Linprog Solution	2.138	3.269

Table 4.1: Table of the mean Euclidean distance between the atom probabilities and each approximation method on satisfiable 20 and 50 atom KBs.

CHAPTER 5

NEURAL NET MODELS

Neural nets can be used to represent a function. Sets of inputs and outputs are used to train these nets. Linear regression assigns weights to connections to capture the function the neural net represents. This chapter examines if the structure of a neural net is itself indicative of the function it represents.

A neural net consists of layers, nodes in each layer, and connecting weights. By creating an image of these connections, the structure of the neural net can be examined in more detail. This chapter examines if this image of a neural net can be correlated to the function it represents.

We trained three sets of neural nets. The first set was trained using satisfiable knowledge bases. These neural nets represented functions to identify points as inside or outside of the feasible region for a satisfiable KB. Then, a neural net was trained using unsatisfiable KBs. The function these neural nets represented also determined whether a point in the hypercube was inside or outside of the feasible region.

To train the nets, inputs and outputs from the feasible point classifier function were gathered. A vector of the points tested was the input representation for the neural net. A vector of ones and zeros, one if the point was in the feasible region and zero if the point was not, was the associated output for the neural net. These inputs and outputs were gathered for both satisfiable and unsatisfiable knowledge bases, and then used to train their respective neural net.

An image was created of these neural nets based off of the connecting weight vectors. Figure 5.1 displays the image of a neural net representing the feasible point function for a satisfiable KB, while Figure 5.2 is that of a neural net representing the feasible point function for an unsatisfiable KB.

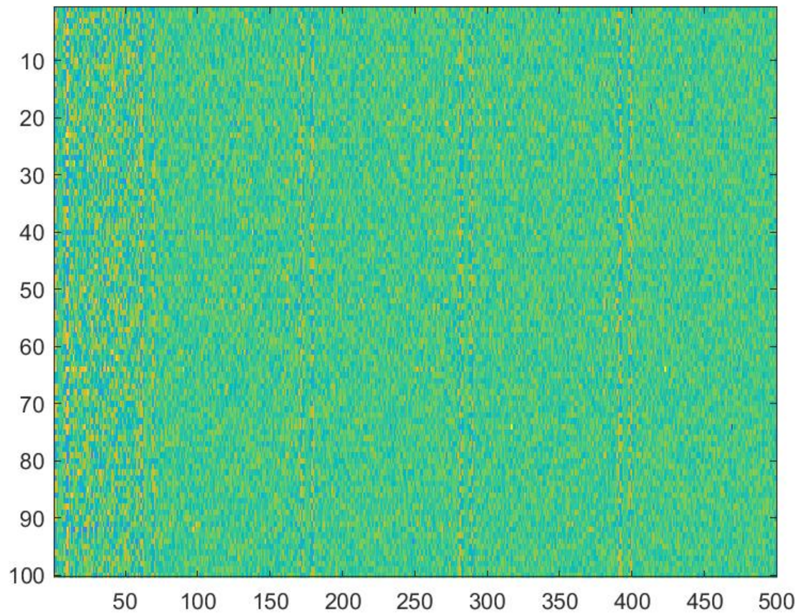

Figure 5.1: Image of Neural Net Trained With Satisfiable KB

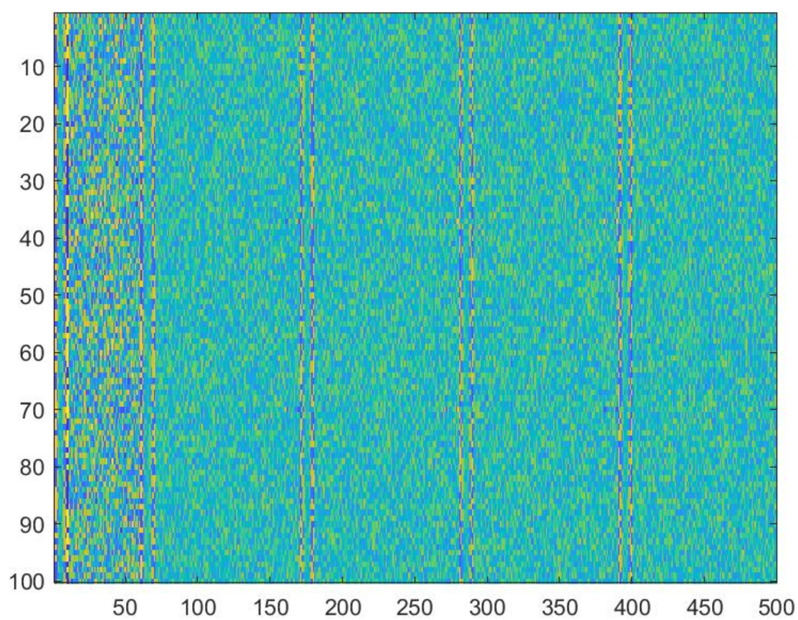

Figure 5.2: Image of Neural Net Trained With Unsatisfiable KB

These images of the first two sets of neural nets were used to train the third, and final, set of neural nets. For the input neural net images, if the neural net was trained to represent the feasible point function for a satisfiable KB, the associated input was a 1. Otherwise, the associated input was a 0.

The final neural net model trained with the neural net images classified 100 50 atom KBs as satisfiable or unsatisfiable. 0.5051 of the classifications were correct. This percentage is virtually equivalent to guessing the satisfiability of a knowledge base. Therefore, an image of the weights of a neural net modelled to represent the feasible point function for a KB does not indicate the satisfiability of the KB.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

Methods using the convex feasible region from the method ChopSAT were analyzed to see if an efficient SAT solution or useful heuristics for solving SAT could be determined. We proposed that certain properties of $\mathcal{F}$ could provide better insight into the satisfiability of a sentence. The thesis also suggested that, due to the continuous nature of H_n, accurate atom probabilities over all solutions could be estimated.

We explored linear programming solutions. These solutions minimized or maximized the projection of the feasible region onto axes in various directions. The goal of this method was to provide insight on the satisfiability of a sentence using these linear programming projection points. Although this method performed well on KBs with 20 atoms or less, it was proven ineffective for KBs with more than a small number of atoms.

We then examined methods for estimating atom probability over all solutions. This thesis explored the accuracy of using the mean of projected points as atom probability estimates. The thesis also calculated the analytic centers of feasible regions and compared those results to the atom probabilities. Finally, the P-center was calculated on these feasible regions and once again compared to the atom probabilities. We determined that the analytic center method most closely approximated the atom probabilities.

Finally, we examined the structure of neural nets to determine if the structure was indicative of sentence satisfiability. We proposed that neural nets trained from satisfiable KBs could have different image structures than those trained from unsatisfiable KBs. However, the structure of the neural net was not shown to be indicative of a SAT solution.

Many of the properties of the feasible region examined in this thesis were not proven to be effective tools for solving SAT. However, the analytic center performed well as an estimate for atom probabilities. Further study would include using the analytic center in agent decision making situations to compare its effectiveness with other methods.

References

[1] M.D. Davis, R. Sigal, and E.J. Weyuker. *Computability, Complexity, and Language*. Morgan Kaufmann, San Diego, CA, 1994.

[2] R. Gamboa and J. Cowles. A Mechanical Proof of the Cook-Levin Theorem. In *Theorem Proving in Higher Order Logics*, volume 3223 of *Lecture Notes in Computer Science*, Berlin, Germany, 2004. Springer Verlag.

[3] G. Georgakopoulos, D. Kavvadias, and C.H. Papadimitriou. Probabilistic Satisfiability. *Journal of Complexity*, 4:1–11, 1987.

[4] T. Henderson, A. Lessen, I. Rajan, and T. Nishida. Chop-SAT: A New Method for Knowledge-Based Agent Decision Making. Technical Report UUCS-21-016, University of Utah, Salt Lake City, UT, February 2023.

[5] T.C. Henderson, A. Mitiche, X. Fan, and D. Sacharny. Some Explorations in SAT. Technical Report UUCS-21-016, University of Utah, Salt Lake City, UT, July 2021.

[6] T.C. Henderson, R. Simmons, B. Serbinowski, M. Cline, D. Sacharny, X. Fan, and A. Mitiche. Probabilistic Sentence Satisfiabilty: An Approach to PSAT. *Artificial Intelligence*, 278, 2020.

[7] N. Nilsson. Probabilistic Logic. *Artificial Intelligence*, 78:71–87, 1986.

[8] M. Sipser. *Introduction to the Theory of Computation*. Cengage Learning, Independence, KY, 2012.