

Chop-SAT: A New Approach to Solving SAT and Probabilistic SAT for Agent Knowledge Bases

Thomas C. Henderson¹^a, David Sacharny², Amar Mitiche³, Xiuyi Fan⁴, Amelia Lessen¹, Ishaan Rajan¹ and Tessa Nishida¹

¹*School of Computing, University of Utah, Salt Lake City, Utah, USA*

²*Blyncsy Inc, Salt Lake City, UT, USA*

³*Department of Telecommunications, Institut National de la Recherche Scientifique, Montreal, Quebec, Canada*

⁴*Nanyang Technological University, Singapore*

{tch, sacharny}@cs.utah.edu, mitiche@emt.inrs.ca, xyfan@ntu.edu.sg, {u1149219, u1256104, tessa.nishida}@utah.edu

Keywords: Autonomous Agent Reasoning, Cutting Planes, SAT, PSAT.

Abstract: An early approach to solve the SAT problem was to convert the disjunctions directly to equations which create an integer programming problem with 0-1 solutions. We have independently developed a similar method which we call Chop-SAT based on geometric considerations. Our position is that Chop-SAT provides a wide range of geometric approaches to find SAT and probabilistic SAT (PSAT) solutions. E.g., one potentially powerful approach to determine that a SAT solution exists is to fit the maximal volume ellipsoid and explore it semi-major axis direction to find an H_n vertex in that direction.

1 INTRODUCTION

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. SAT is the original NP-complete problem and requires polynomial time on a nondeterministic Turing machine (Sipser, 2012). Recently, Henderson et al. proposed a geometric approach, called *Chop-SAT*, for solving SAT (see (Henderson et al., 2021)). The method produces a convex polytope feasible region, and the goal here is to explore properties of the feasible region which allow the solution of SAT or 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 semantics of the hypercube is that the i^{th} axis corresponds to the values which can be assigned to the i^{th} variable, and the values can be relaxed to lie in the interval $[0,1]$.

Given any point in H_n , that point can be considered as a set of probabilities for the atoms. This al-

lows consideration of a probabilistic version of SAT called *Probabilistic SAT (PSAT)* (see (Georgakopoulos et al., 1987; Nilsson, 1986) 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 $0 \leq \pi(\omega_k) \leq 1$, and $p_i = \sum_{\omega_k \models C_i} \pi(\omega_k)$, $\omega_k \in \Omega$.

Chop-SAT is based on the following observation: when an agent's knowledge base is represented as a CNF sentence, then every conjunct is a disjunction. A disjunction with k_i literals is made false by assigning a falsifying truth value to each of the disjunction's literals. This assignment, along with all possible 0-1 assignments to the $k = n - k_i$ literals not in the disjunction, defines the models which do not satisfy the conjunct. Geometrically, this set of models is a sub-hypercube, H_k , of the n -dimensional hypercube, H_n .

For each conjunct there exists an $(n-1)$ -dimensional hyperplane, h , that separates H_k from the corners of H_n which satisfy the conjunct. This leads to the notion of a chop: the hyperplane, h , is represented so that H_k is on h 's negative side (viewing the hyperplane equation as a signed distance function) while solution corners are on h 's

^a <https://orcid.org/0000-0002-0792-3882>

non-negative side. By intersecting the non-negative half-space of h with n -dimensional space, a convex feasible region is created which contains the solutions for the conjunct (this holds because the intersection of convex sets is a convex set). Applying this operation for each conjunct in the CNF sentence results in a convex feasible region which must contain solutions to the CNF sentence, if any.

Our position is that *Chop-SAT* offers the possibility to find efficient and effective ways to solve SAT. *Chop-SAT* 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 resulting feasible region is convex.

The feasible region represents the solution space for the KB. If any of the original hypercube corners exist in the feasible region, the KB is satisfiable, otherwise it is unsatisfiable.

2 BACKGROUND

Gomory (Gomory, 1958) introduced the cutting plane method as an extension to Dantzig’s linear programming approach to solving discrete variable extremum problems (Dantzig, 1957). Gomory provided a method so that when the objective function was maximized (e.g., using the simplex method), if the result was a non-integral solution, then a new constraint plane was found algorithmically to separate (cut) the non-integer solution from the integer solutions. Chvatal (Chvatal, 1973; Chvatal, 1985) developed this method further, proved supporting theorems for bounded polytopes, and applied the results to solve combinatorial problems. Note that integer programming is in *NP* (Papadimitriou, 1981).

In terms of solving SAT, Cook et al. (Cook et al., 1987) examined the complexity of cutting plane proofs, and in particular, those for the unsatisfiability of formulae in propositional calculus. This was done in terms of resolution theorem proving using the cutting planes method. Note that it is straightforward to produce an equation from a disjunction; e.g., consider $(a \vee a)$ which results in the equation $a + b \geq 1$. Each conjunct gives rise to an equation, and the set forms an integer programming problem where a and b take on 0-1 values. Note that linear programming

relaxation can be used to allow the a and b to take on values in the interval $[0, 1]$ – i.e., relaxing the 0-1 requirement. Hooker (Hooker, 1988) provided a way to handle generalized resolution by observing that the resolvent of two clauses corresponds to a certain cutting plane in integer programming. Buss et al. (Buss and Clote, 1996) describe an extension to Cook’s cutting plane refutation approach that applies to threshold logic analysis. For some recent work using cutting planes, see Devriendt et al. (Devriendt et al., 2021) who use the method to improve state-of-the-art pseudo-Boolean optimization. All these methods are still exponential time complexity.

3 THE CHOP-SAT METHOD

We look at the SAT problem geometrically: for an n -variable problem, the corners of the hypercube are the only possible SAT solutions, and the interior points are possible atom probabilities assignments. Given a Conjunctive Normal Form (CNF) sentence, $S = C_1 \wedge C_2 \wedge \dots \wedge C_m$, over a set of Boolean variables $A = \{a_1, a_2, \dots, a_n\}$, each conjunct, C_i , has at least one truth assignment that makes it false (note that $a \vee \neg a$ is not allowed). If there are k_i literals in conjunct C_i , then there are 2^k truth assignments that make it false, where $k = n - k_i$. A hyperplane can then be found for each conjunct which separates the solutions from the non-solutions. Moreover, this hyperplane can be positioned so as to cut the edges between non-solution corners and their feasible region neighbors anywhere in the range $0 < x \leq 1$ along those edges. The intersection of the non-negative side of the hyperplane with the initial feasible region (H_n or \mathcal{R}^n) produces a convex feasible region. Continuing this process for each conjunct, the result is a convex feasible region which may or may not contain a corner of H_n – i.e., a solution for the CNF sentence.

Suppose $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$ is a set of Boolean variables, and the set of \mathcal{A} -formulas is the inductive set of propositional well-formed formulas over \mathcal{A} . A Conjunctive Normal Form (CNF) sentence over \mathcal{A} is defined as a conjunction of a set of disjunctions of literals.

$$S = C_1 \wedge C_2 \wedge \dots \wedge C_m$$

$$C_i = L_{i_1} \vee L_{i_2} \vee \dots \vee L_{i_{k_i}}$$

$$L_{i_j} = a_{i_j} \text{ or } L_{i_j} = \neg a_{i_j}$$

The *Satisfiability Problem* is defined as (Davis et al., 1994):

Find an efficient algorithm for testing an \mathcal{A} -formula in CNF to determine whether it is truth-functional satisfiable.

By efficient, we understand “of polynomial complexity.”

3.1 Basic Approach

The SAT problem is cast as a linear programming problem:

$$\begin{aligned} & \text{Minimize } f \cdot x \\ & \text{Subject to: } Ax \leq c \end{aligned}$$

where each constraint is given by:

$$-\alpha_i \cdot x \leq c_i$$

A solution for the SAT sentence exists iff a solution exists for the LP problem with $x \in \{0, 1\}^n$.

Given a set of m conjuncts, $C_i, i = 1 : m$, each conjunct is used to produce a hyperplane of dimension $n - 1$ which separates the solutions (i.e., some subset of vertexes of the n -dimensional hypercube) from non-solutions. The hyperplane for the i^{th} conjunct is:

$$\alpha_i \cdot x + c_i = 0$$

Each of these hyperplanes produces an inequality:

$$-\alpha_i \cdot x \leq c_i$$

A matrix, A , is produced where each row is the $1 \times n$ -tuple $\alpha(i)$. An $n \times 1$ -vector, c , is constructed where the i^{th} element of c is c_i .

The way these hyperplanes are constructed, it is now possible to run the interior-point method for linear programming to find feasible points which minimize $f^T x$ for $x \in X$, where X is the feasible region and f is an n -D vector. This allows points to be found which lie on the feasible region boundary.

Given m conjuncts, $C_i, i = 1 \dots m$, then:

$$C_i = L_{i_1} \vee L_{i_2} \vee \dots \vee L_{i_{k_i}}$$

Observe that:

- If $k_i = n$, then this eliminates 1 solution (a vertex of H_n).
- If $k_i = n - 1$, then this eliminates 2 points (two vertexes joined by an edge of H_n).
- ...
- If $k_i = 1$, then this eliminates half the points in the hypercube (all in a hyperplane of dimension $n - 1$).

The individual hyperplane is determined as follows. Let $\mathcal{A} = \{1, 2, \dots, n\}$, and $I \subseteq \mathcal{A}$. Given $C_i = L_1 \vee L_2 \vee \dots \vee L_{k_i}$, then define α_i , the hyperplane normal vector, as follows.

$$\forall i_j \in I, \alpha_i(i_j) = 1 \text{ if } L_j \text{ is an atom } a_{i_j}, \text{ else } -1$$

$$\begin{aligned} \forall m \notin I, \alpha_i(m) &= 0 \\ \alpha_i &:= \frac{\alpha_i}{\|\alpha_i\|} \end{aligned}$$

In order to get the constant for the hyperplane equation, a point must be found on the hyperplane. This is selected so that the hyperplane cuts the edges of the hypercube at a distance ξ from the center of H_k . This distance depends on the number k_i of literals in the conjunct:

$$d = \|\xi \frac{\overline{b_{k_i}}}{k_i}\|$$

where $\overline{b_{k_i}}$ is a k_i -tuple of 1's. Next:

$$\begin{aligned} \forall i_j \in I, p(i_j) &= 0 \text{ if } L_i \text{ is an atom, else } 1 \\ \forall m \notin I, p(m) &= 0 \end{aligned}$$

Then p is a non-solution vertex. To find a point, q , on the hyperplane:

$$q = p + d\alpha_i$$

This allows a solution for the constant, c , in the hyperplane:

$$c_i = -(\alpha_i \cdot q)$$

This yields the hyperplane equation:

$$\alpha_i \cdot x + c = 0$$

and the resulting inequality:

$$-\alpha_i \cdot x \leq c$$

Note that the feasible region as defined above is not necessarily bounded (e.g., when the initial feasible region is \mathfrak{R}^n . Consider the simple case where $n = 2$ and $S = C_1 = a \vee b$. Then the feasible region resulting from this one cut is all points above the line through the points $[1;0]$ and $[0;1]$. An unbounded feasible region indicates that a SAT solution exists.

4 Some Examples

4.1 2D One Solution

Consider the two clauses in modus ponens:

$$\begin{aligned} & 1. a_1 \\ & 2. \neg a_1 \vee a_2 \end{aligned}$$

Then the feasible region is indicated in Figure 1 as the black point samples (the hyperplanes are shown as blue lines), and this region is unbounded. The hyperplane found for conjunct 1 (with $\xi = 0.9$) is:

$$1.0a_1 + 0a_2 - 0.9 = 0$$

while the hyperplane for conjunct 2 is:

$$-0.7071a_1 + 0.7071a_2 + 0.7071 = 0$$

The solution is: $a_1 = 1$ and $a_2 = 1$.

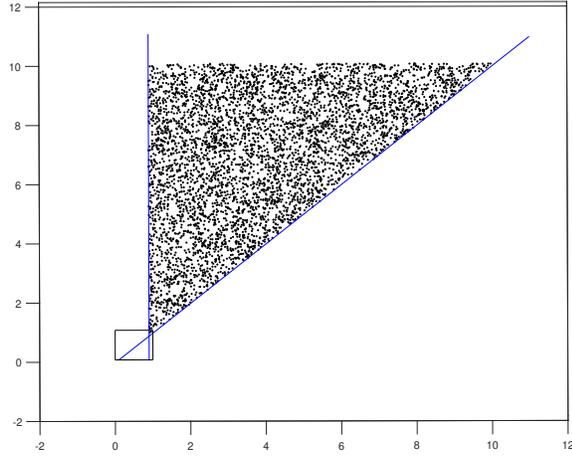


Figure 1: The Feasible Region for Unbounded Modus Ponens.

4.2 3D: One Solution

As a final example, consider the case with 3 variables with seven clauses that chop off all corners individually except [1;1;1]. Re-writing this in CNF yields:

1. : $a_1 \vee a_2 \vee \neg a_3$
2. : $a_1 \vee \neg a_2 \vee a_3$
3. : $a_1 \vee \neg a_2 \vee \neg a_3$
4. : $\neg a_1 \vee a_2 \vee a_3$
5. : $\neg a_1 \vee a_2 \vee \neg a_3$
6. : $\neg a_1 \vee \neg a_2 \vee a_3$
7. : $\neg a_1 \vee \neg a_2 \vee \neg a_3$

Then the hyperplane equations are:

$$C_1 : -0.5774a_1 - 0.5774a_2 + 0.5774a_3 + 0.5831 = 0$$

$$C_2 : -0.5774a_1 + 0.5774a_2 - 0.5774a_3 + 0.5831 = 0$$

$$C_3 : -0.5774a_1 + 0.5774a_2 + 0.5774a_3 + 0.0058 = 0$$

$$C_4 : 0.5774a_1 - 0.5774a_2 - 0.5774a_3 + 0.5831 = 0$$

$$C_5 : 0.5774a_1 - 0.5774a_2 + 0.5774a_3 + 0.0058 = 0$$

$$C_6 : 0.5774a_1 + 0.5774a_2 - 0.5774a_3 + 0.0058 = 0$$

$$C_7 : 0.5774a_1 + 0.5774a_2 + 0.5774a_3 + 0.5716 = 0$$

The only solution is: [1;1;1]. The feasible region is shown in Figure 2.

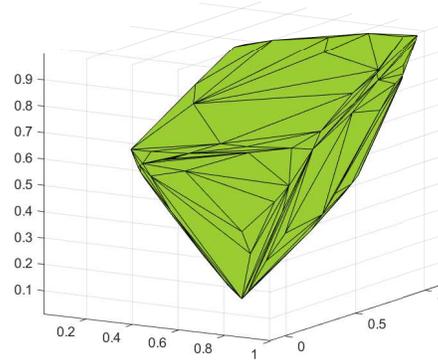


Figure 2: The Feasible Region for 3D with only [1;1;1] as a Solution.

5 Some Preliminary Investigations

Some useful facts have been found with respect to feasible region properties which differentiate satisfiable from unsatisfiable CNF sentences. For example, given a CNF sentence, S , such that no point in the feasible region arising from S is more than $\sqrt{n-1}/2$ distant from the center of H_n , then S is unsatisfiable.

Let H_n be an n -D hypercube. A *cut* is the removal from H_n of a k -D hypercube, $H_k \subset H_n$, $0 \leq k < n$. A cut is effected as follows. Let $V = \{v \mid v \text{ is a vertex of } H_k\}$, and define $V_n = \{v \mid v \text{ is a vertex of } H_n, v \notin V, v \text{ neighbors an element of } V\}$. Let h_{V_n} be the $(n-1)$ -D hyperplane defined by the points in V_n where:

- $v \in V$ is on the negative side of h_{V_n}
- the open edge segments in H_n that lie between elements of V and V_n are on the negative side of h_{V_n} , and
- $v \in V_n$ is on the non-negative side of h_{V_n} .

Let h_s be the n -D half-space on the non-negative side of h_{V_n} . Then a cut produces a feasible region as follows:

$$F = H_n \cap h_s$$

Given a set of cuts, $C = \{C_i\}$, then the feasible region for C is $\mathcal{F} = \cap F_i$, where F_i is the feasible region for C_i .

Lemma 1: Given H_n and a set of cuts, C , if these cuts remove every vertex of H_n , then $\forall H_1 \subset H_n$ (i.e., edges), $H_1 \cap \mathcal{F} = \emptyset$, where \mathcal{F} is the feasible region of C .

Proof: Consider an arbitrary $H_1 \subset H_n$, and call its end points A and B . Some cut removes A and in so doing

also removes the open line segment (A, B) since it is on the negative side of h_{V_n} . Since some cut removes B , then all of H_1 is removed.

Lemma 2: If C is a set of hyperplane cuts that removes all corners of H_n and results in the feasible region \mathcal{F} , then every 2-D face of H_n is either removed or at most one point remains, and it is in the center of the 2D square:

$$\begin{aligned} \forall H_2 \subset H_n, \\ H_2 \cap \mathcal{F} = \emptyset \end{aligned}$$

or $H_2 \cap \mathcal{F}$ is one point at the center of H_2

Proof: Let A, B, C, D be the set of vertexes of a 2-D square with edges $\{(A, B), (B, C), (C, D), (D, A)\}$ in H_n ; let E be the center of the square. Consider a cut that removes A . Any cut will at the most leave the triangle BCD in the feasible region. Cutting B leaves at most triangle CDE in the feasible region. Cutting C leaves at most segment DE in the feasible region. Finally, cutting D leaves at most the point E . This analysis holds for every 2-D square (i.e., any H_2) in H_n .

From these results we conclude that the center of $H_k \subset H_n$ is located at distance $\frac{\sqrt{n-k}}{2}$ from the center of H_n . Thus, corners of H_n are at the maximal distance, and centers of edges are next. But both of these types of points are removed according to the lemmas. Thus, the maximal possible distance is to the center of a 2-D square (H_2) and is distance $\frac{\sqrt{n-2}}{2}$ from the center of H_n .

5.1 Maximal Volume Inscribed Ellipsoid

The maximal volume inscribed ellipsoid (MVE) may play a significant role in determining whether or not a SAT solution exists. An ellipsoid is defined by the equation:

$$S = \{v \mid v = x + Es, \|s\| \leq 1\}$$

where x is the center of the ellipse, and E is the ellipsoid matrix that transforms points in the unit n -sphere to points in the ellipse. If the feasible region is unbounded, then that can be detected (e.g., add the hyperplanes for increasingly large hypercubes and linear programming will show the bounds increasing accordingly) and satisfiability determined. Even if not, the directions of the ellipsoids axes may point in the direction of a solution. Importantly, the MVE can be found in polynomial time (Khachiyan and Todd, 1990). Moreover, robust implementations exist and can be used in practice (see (Zhang, 1998; Zhang, 2003)).

Consider the 3D example given above. Given the center of the MVE, and the matrix defining it, the extreme point on the ellipsoid (and in the feasible region) can be located. Figure 3 shows some points sampled from the feasible region (blot dots), the center of the ellipse (red circle/star), and extreme point on the longest ellipsoid axis (blue circle/star), and the major ellipsoid axis (black line). We are currently studying how well this works in higher dimensions. Taking a look at the Matlab output for the singular

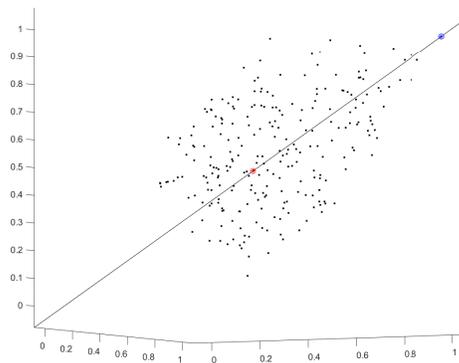


Figure 3: Demonstration of how the Maximal Volume Inscribed Ellipsoid in the Feasible Region Can Help Find a Solution Along its Longest Axis.

value decomposition of the ellipsoid matrix, we find that the major axis aligns with $x = y = z$ line:

$$U = \begin{bmatrix} -0.5774 & -0.8165 & 0 \\ -0.5774 & 0.4082 & -0.7071 \\ -0.5774 & 0.4082 & 0.7071 \end{bmatrix}$$

and that the most distance feasible region point along that line is:

$$pe = \begin{bmatrix} 0.9987 \\ 0.9987 \\ 0.9987 \end{bmatrix}$$

5.2 PSAT Approximations

Another area of application is in determining probabilities of atoms to help in the agent decision making process. For a detailed example, consider the Wumpus World problem popularized by Russell and Norvig and described in (Russell and Norvig, 2009). An agent looks for gold in a 4x4 grid where there are pits and a Wumpus which can kill the agent. Russell describes how to represent this using propositional logic, and in our formulation there are 80 atoms (or

logical variables) and 402 conjuncts in the CNF. The atoms correspond to 16 for a Breeze in a cell, 16 for gold in a cell, etc. If the probability of a pit in each cell (except the start cell) is 20%, then Monte Carlo can be used to determine the ground truth a priori atom probabilities. An agent would like to estimate the probability of danger corresponding to each of its possible actions, and this can be done using the feasible region for the CNF sentence.

We have determined the feasible region for the Wumpus World CNF, then found the solutions for projecting onto all 80 axes (in both directions) obtaining 160 extreme points on the feasible region. We then take the average of these to find an approximation to the atom probabilities. Figure 4 shows both the Monte Carlo result as well as that from *Chop-SAT*. It is clear that *Chop-SAT* provides a very good qualitative approximation produced just from the rules of the game; this is adequate when an agent must decide based on the lower probability of danger.

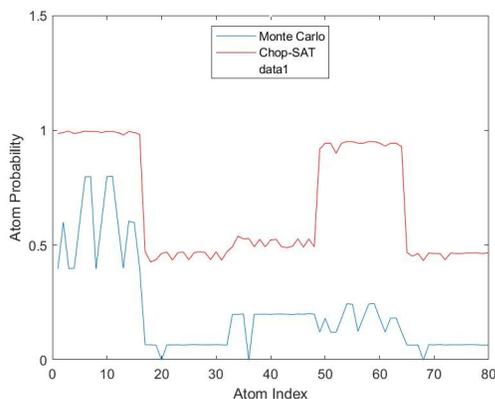


Figure 4: The Monte Carlo and *Chop-SAT* Estimates of the Atom Probabilities for Wumpus World.

This is simply an approximation to the atom probabilities; to get the model probabilities an approximation technique can be used. E.g., assume the atom variables are independent, then compute the top 100 model probabilities and assume the remainder are 0. Normalize these probabilities, and it is then possible to calculate an estimate for any logical sentence over the variables. We are also investigating how well this works.

Note that our approach is a polynomial time approximation (i.e., very fast) more in line with our previous work (Henderson et al., 2020), and is very different from standard approaches like Nilsson (Nilsson, 1986), or more recent works like Finger and De Bona (Finger and Bona, 2015) which rely on SAT solvers.

6 CONCLUSIONS

The position espoused here is that *Chop-SAT* merits further study as a way to solve both discrete and probabilistic SAT, and allows efficient and effective solution approximation techniques. This in turn allows a knowledge-based agent to improve its decision making process. Note that the feasible region is produced very efficiently since converting conjuncts to hyperplanes is linear time complexity, and obtaining the properties of the feasible region is done using linear programming based on the interior point method which is low polynomial time.

ACKNOWLEDGEMENTS

This work was supported in part by National Science Foundation award 2152454.

REFERENCES

- Buss, S. and Clote, P. (1996). Cutting Planes, Connectivity, and Threshold Logic. *Archive for Mathematic Logic*, 35:33–62.
- Chvatal, V. (1973). Edmonds Polytopes and a Hierarchy of Combinatorial Problems. *Discrete Mathematics*, 4:305–337.
- Chvatal, V. (1985). Cutting Planes in Combinatorics. *European Journal of Combinatorics*, 6:217–226.
- Cook, W., Coullard, C., and Turan, G. (1987). On the Complexity of Cutting-Plane Proofs. *Discrete Applied Mathematics*, 18:25–38.
- Dantzig, G. (1957). Discrete-Variable Extremum Problems. *Journal of Operations Research Society of America*, 5(2).
- Davis, M., Sigal, R., and E.J.Weyuker (1994). *Computability, Complexity, and Languages*. Morgan Kaufmann, San Diego, CA.
- Devriendt, J., Gocht, S., Demirovic, E., Nordstrom, J., and Stuckey, P. (2021). Cutting to the Core of Pseudo-Boolean Optimization: Combining Core-Guided Search with Cutting Planes Reasoning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*. Elsevier.
- Finger, M. and Bona, G. D. (2015). Probabilistic Satisfiability: Algorithms with the Presence and Absence of a Phase Transition. *Annals of Mathematics and Artificial Intelligence*, 75:351–389.
- Georgakopoulos, G., Kavvadias, D., and Papadimitriou, C. (1987). Probabilistic Satisfiability. *Journal of Complexity*, 4:1–11.

- Gomory, R. (1958). Outline of an Algorithm for Integer Solution to Linear Programs. *Bulletin of the American Mathematical Society*, 64(5):275–278.
- Henderson, T., Simmons, R., Serbinowski, B., Cline, M., Sacharny, D., Fan, X., and Mitiche, A. (2020). Probabilistic Sentence Satisfiability: An Approach to PSAT. *Artificial Intelligence*, 278:71–87.
- Henderson, T. C., Mitiche, A., Fan, X., and Sacharny, D. (2021). Some Explorations in SAT. Technical Report UUCS-21-016, University of Utah.
- Hooker, J. (1988). Generalized Resolution and Cutting Planes. *Annals of Operations Research*, 12:217–239.
- Khachiyan, L. and Todd, M. (1990). On the Complexity of Approximating the Maximal Volume Ellipsoid for a Polytope. Technical Report TR No. 893, Cornell University, Ithaca, NY.
- Nilsson, N. (1986). Probabilistic Logic. *Artificial Intelligence Journal*, 28:71–87.
- Papadimitriou, C. (1981). On the Complexity of Integer Programming. *Journal of the Association of Computing Machinery*, 28:765–768.
- Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, 3rd edition.
- Sipser, M. (2012). *Introduction to the Theory of Computation*. Cengage Learning, Independence, KY.
- Zhang, Y. (1998). An Interior-Point Algorithm for the Maximum-Volume Ellipsoid Problem. Technical Report TR98-15, Rice University, Houston, TX.
- Zhang, Y. (2003). On Numerical Solution of the Maximum Volume Ellipsoid Problem. *SIAM Journal on Optimization*, 14(1).