Survey on the structure of Boolean satisfiability problem

M.Sc. Thesis

by

Madhu

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY INDORE

MAY 2024

Survey on the structure of Boolean satisfiability problem

A THESIS

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Science

by

Madhu

(Roll No. 2203141019)

Under the guidance of

Dr. M. Tanveer

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY INDORE

MAY 2024

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **"Survey on the structure of Boolean satisfiability problem"** in the partial fulfillment of the requirements for the award of the degree of **Master of Science** and submitted in the **Department of Mathematics**, **Indian Institute of Technology Indore**, is an authentic record of my work carried out during the period from July 2023 to May 2024 under the supervision of **Dr. M. Tanveer**, Associate Professor, Department of Mathematics, IIT Indore. The matter presented in this thesis has not been submitted for the award of any other degree of this or any other institute.

Madhu 30-05-24

Signature of the student with date

(**Madhu**)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

30/05/2024

Signature of Thesis Supervisor with date

(**Dr. M. Tanveer**)

Madhu has successfully given her Thesis oral Examination held on 27^{th} May, 2024.

Signature of supervisor of M.Sc. Thesis

Date: 30/05/2024

V.K. Tewari

Signature of Convener, DPGC

Date: 30/05/2024

Dedicated to my family

"I have to remind myself that some birds aren't meant to be caged. Their feathers are just too bright. And when they fly away, the part of you that knows it was a sin to lock them up does rejoice."

–***Shawshank Redemption (Movie)***

Acknowledgements

I am incredibly grateful to my supervisor Dr. M. Tanveer for his assistance and guidance throughout my M.Sc. thesis period. Without his support my M.Sc. journey could not be completed. His way of observing the problem and breaking it down motivates me to look at problems differently. I am completing M.Sc. with much more maturity in mathematics as compared to when I came and his role is magnificent in this.

I extend my thanks to the MSc project evaluation committee and HOD Prof. Niraj Kumar Shukla and DPGC convener Dr. Vijay Kumar Sohani. I am also thankful to all our teachers for their kind and valuable suggestions.

I also want to thank the Department of Mathematics and Department of Science and Technology (DST) Govt. of India for the facilities provided at Bhaskacharya Lab.

I am thankful to Ph.D. scholars Ashwani Kumar Malik, Md Sajid, Anuradha Kumari, Abdul Quadir and Mushir Akhtar for clearing my doubts related to my Thesis, academics, and life. Despite having busy schedules they always took out time to help me.

I would like to thank my family as they are the reason of my existence. Their support is always there for me and it makes me feel blessed. Words can't describe their love for me and my love for them.

At last I want to thank some lovely people I met here and will be in my hearts forever Saumya, Vipin, Rinku, Chhavi, Pragya, Rahul, Nishi, Yashovardhan, Madhurima, Leela Krishna, Shubham, Kanchan, Ishita, Uttam, Akash, Mohit, Suman and Kevin amazing juniors and many others.

Abstract

P versus NP is a popular conjecture in Computer science. In this conjecture P stands for the problem for which there exists a polynomial time running algorithm and NP stands for the problems for which there exists a nondeterministic polynomial time algorithm. $P \stackrel{?}{=} NP$ is the conjecture where we have to answer the question of whether there exists a polynomial time running algorithm for all NP problems or not.

This thesis presents a comprehensive survey of the Boolean Satisfiability Problem (SAT), a cornerstone of theoretical computer science and practical applications in various domains. We explore the fundamental aspects of SAT, including its historical development, theoretical significance, and the progression of algorithms designed to solve it. The survey encompasses classic approaches such as the Davis-Putnam-Logemann-Loveland (DPLL) algorithm and modern advancements like Conflict-Driven Clause Learning (CDCL) solvers.

We delve into the empirical phenomena observed in SAT instances, such as the phase transition behavior and the easy-hard-easy pattern, providing insights from seminal works by Cheeseman et al. [**11**], and Mitchell et al. [**12**] The

investigation extends to the structure of SAT problems, highlighting the concepts of backbones and backdoors, and their influence on problem hardness.

Furthermore, the thesis examines the application of SAT solvers in industrial contexts, where instances often differ significantly from random benchmarks. The superior performance of CDCL solvers on industrial instances, as opposed to random instances, is analyzed, with a focus on the role of backdoor structures in facilitating this efficiency. Contributions by Gregory et al. [**13**] and Zulkoski et al. [**8**] are reviewed to understand how these solvers implicitly exploit structural features.

By synthesizing findings from various studies, this survey provides a cohesive understanding of SAT, bridging the gap between theoretical insights and practical applications. The thesis concludes with a discussion on the future directions in SAT research, emphasizing the potential for further advancements in solver algorithms and their applications across different fields.

Contents

CHAPTER 1

Introduction

Satisfiability also known as SAT is the first NP problem which was proven to be NP complete. Many problems of combinatorics can be reduced to SAT. This report is mainly focusing on the solution of 3-SAT problem. So firstly, discuss some definitions and define some terminologies that are useful throughout this report.

Consider a set of variables $X = \{x_1,\ x_2, \ldots, x_n\}$

1.1 Boolean Algebra

1. NP complete problem: A P is said to be NP complete if every NP problem can be reduced into the problem P in polynomial time.

2. Truth value: Let $x_i \in X$ which can take value either true or false. Then the true or false value of the given variable is known as the truth value.

3. AND operator: $x_1 \wedge x_2$ gives the result true iff both x_1 and x_2 are true

otherwise $x_1 \wedge x_2$ is false, where $\wedge$ denotes the logical AND operator.

4. OR operator: $x_1 \vee x_2$ gives the result true if either x_1 or x_2 is true otherwise $x_1 \vee x_2$ is false, where $\vee$ denotes the logical OR operator.

5. NOT operator: Let $x_i \in X$ then $\neg x_i$ gives the result true if the variable x_i is false vice versa, where $\neg$ denotes the logical NOt operator.

6. Boolean algebra: Boolean algebra is an expression of the variables from the set X composed with the logical operators AND, OR, and NOT. The output of a Boolean algebra is either true or false.

7. Literal: If a Boolean algebra consist a variable x then the variable x or $\neg x$ can be present in the given boolean algebra. So the literal of x is defined as x or the negation of x.

8. Clause: A Boolean algebra where the logical OR operator connects the literals.

9. Conjunctive normal form (CNF): The boolean algebra is called in conjunctive normal form if the logical operator AND connects all the clauses.

10. Tseytin transformation: Tseytin transformation is the algorithm which is used to convert a given boolean algebra into a conjunctive normal form in polynomial time.

11. Satisfibility (SAT): Satisfiblility also known as SAT, is the problem of answering the question of whether a given Boolean algebra in CNF is true for some particular true and false value of the variables of given boolean algebra. Also, note that SAT is a NP complete problem.

12. 3-SAT: $3 - SAT$ is the problem where all the clauses of the given CNF contain exactly three literals.

13. Cook-Levin Theorem: It states that $P = NP$ if and only if a polynomial time running algorithm exists for any one of NP complete problems.

1.2 Graph Theory

Let's consider boolean expression Φ in conjunctive normal form with variables $Z = \{z_1, z_2, \ldots, z_n\}$ and clauses K. Consider an undirected weighted graph $H(Y, v)$ where Y is the set of vertices and v is the edge weight function defined from $Y \times Y$ to $\mathbb{R}^+ \cup \{0\}$ satisfying the condition that $v(z_i, z_j) = v(z_j, z_i)$ then define:

1. Garph of Variable Incidence (GVI) for SAT: We will convert the given CNF formula in the graph $H(Y, v)$ with the variables Z as the set of vertices and the weight v is given by:

$$v(z_i, z_j) = \sum_{\substack{k \in K \\ z_i, z_j \in k}}^{m} \frac{1}{\binom{|k|}{2}},$$

   where z_i, $z_j \in Z$ and $|k|$ gives the number of variables present in the clause $k \in K$.

2. Diameter (Shortest Path Length): Consider z_i, $z_j \in Z$ then the diameter from z_i to z_j is defined as the distance of minimal weight required to go through z_i to z_j and is denoted by d_{z_i, z_j}.

3. Degree of Vertex (DoV): Let z_i, $z_j \in Z$ and let us denote the edge from the vertex z_i to z_j by $E_{i,j}$. Then the DoV of z_i denoted by $\deg_{z_i}$ is mathematically given by:

$$\deg_{z_i} = |\{E_{i,j} \text{ such that } z_i, z_j \in Z\}|.$$

   Further, we can define the degree of vertex z_i with weight. The weighted vertex degree of z_i is defined as the total weights of all the adjacent edges

of the vertex z_i and is denoted by $\deg w_{z_i}$ and is given by:

$$\deg w_{z_i} = \sum_{\substack{z_j \in Z \\ \exists E_{z_i, z_j}}} w(z_i, z_j).$$

4. Box: Let B is a box defined as a collection of vertices i.e. $B \subset Z$ with the magnitude of the cardinality of the set B such that $\forall z_i, z_j \in Z$ we have $d_{z_i, z_j} < |B|$.

5. Graph volume: Volume of the graph $H(Y, v)$ denoted by vol_H is defined as the sum of the weighted degree of vertices of H, mathematically vol_H is given by:

$$\text{vol}_H = \sum_{z \in Z} \deg w_z.$$

Empirical data from a variety of methodologies supports either the existence or the absence of dimensional measures in SAT problems. Structural measures are typically computed for every SAT instance and then means and standard deviations are determined. Applying structural measures as features, a unique method involves classifying SAT cases into arbitrary, crafted, and industrial classes and training a classifier model. The existence of structural parameters in SAT instances is indicated by high classification accuracy, implying that these measures can function as predictors of SAT features.

Research studies encompass the analysis of structural measurements associated with clause learning algorithms and examining the effects of introducing new clauses on the SAT structure. The objective is to comprehend how clause learning affects SAT structure and apply this knowledge to enhance SAT solver performance. A summary of the structural elements of SAT as discussed from both theoretical and experimental angles in the literature is given in this section.

CHAPTER 2

Structural Meaures for SAT

Since the early 1990s, phase transitions have remained at the forefront of research on artificial intelligence (AI). Informally speaking, a "phase transition" is a sudden change in a problem's behavior resulting in by a shift in one of the key factors. There is a clear phase transition between the relative values of clauses to variables and the level of difficulty of $k-SAT$ situations (see Table 1). Empirical examinations of random $k-SAT$ cases show that the mathematical challenge of solving them follows an "easy hard easy" pattern as the ratio of the number of clause to the number of variables varies; the most difficult scenarios, when 50 percent from the total examples are satisfiable, exists close to the point of transition of phase. The formal description of the change of phase depending on the ratio of the clause to the variable is as follows:

Phase Transition Let the class of issues generated from a set of SAT problems be denoted by $R_{\Phi}(|Z|, |K|)$. Where $|Z|$ is the number of variables and $|K|$ denoting the number of clauses, define Φ with variables Z and clauses K.

The likelihood that a randomly chosen problem from $R_\Phi(|Z|, |K|)$ is satisfiable is indicated by the variable $\text{Prob}_\Phi(SAT, R\Phi(|Z|, |K|))$. If and only if the value $\frac{|K|}{|Z|}$ has a threshold $\bar{\alpha}$ such that, Φ experiences a phase shift between satisfiability of Φ and unsatisfiability of Φ.

$$\lim_{|Z|\to\infty} \text{Prob}_\Phi(SAT, R\Phi(|Z|, |K|)) = \begin{cases} 0, & \alpha > \bar{\alpha}, \\ 1, & \alpha < \bar{\alpha}. \end{cases}$$

An "easy hard easy" pattern relative to certain problem parameters has already been identified. However, the study was limited due to the small size of the SAT examples and the basic search method of backtracking. Some researchers focused on uniform cases of random $3 - SAT$ and identified that clauses to variables ratio is the order parameter. Their experiments demonstrated that the Davis-Putnam (DP) solver's average performance followed the "easy hard easy" design, with the most challenging cases at the point of transition of phase which is 4.3. Similar patterns were found in various randomly generated SAT. Research into phase transitions in industrial-like randomly generated cases showed that these instances also exhibit a phase transition based on the ratio of the number of clauses to the number of variables, though at reduced values compared to benchmarks of randomly $k - SAT$. The better presentation of solvers based on CDCL relative to "look ahead based" solvers on these benchmarks suggests that industrial-like random benchmarks share structural similarities with industrial examples. The difficulty level of the benchmarks was measured using metrics such as the number of DP calls, branches, average proof tree nodes, CPU time, backtracks, or conflicts. Studies by Zulkoski et al. [8] found that traditional metrics like the number of clauses, variables, or the ratio of the number of the clauses to the number of variables do not give any kind of connection to the performance of the solver based on CDCL and industrial instances.

Backbone and Backdoor: Well-known structural characteristics of Boolean expressions that have been researched in the literature are backbones and back-

doors. The main topic of this survey is the foundation of solely satisfiable SAT instances. Calculation of backbone or the backdoor is generally impossible. As a result, these attributes' upper and lower bounds are established. Consider a CNF Boolean expression Φ over the set of variables Z and clauses K. Then we can define formally:

1. Backbone Literal: Let z be a literal of the boolean expression Φ then z is called a backbone literal if and only if the truth value of the literal z is fixed in all the assignments of Φ.

2. Backbone: The collection of all the backbone literals is defined as the backbone of Φ.

3. Weak Backdoor: Consider a sub-solver Γ then a collection of variables $BD_\Phi \subseteq Z$ is known as the weak backdoor with respect to the sub-solver Γ if and only if Γ gives the result satisfiability or unsatisfiability for some assignment of BD_Φ.

4. Strong Backdoor: Consider a sub-solver Γ then a collection of variables $BD_\Phi \subseteq Z$ is known as the strong backdoor with respect to the sub-solver Γ if and only if Γ gives the result satisfiablity for some assignment of BD_Φ.

5. LS Backdoor: Consider a sub-solver Γ then a collection of variables $BD_{LS} \subseteq Z$ is known as the LS backdoor with respect to the sub-solver Γ if and only if there is a search tree investigation order s.t. a CDCL SAT solver branches exclusively on variables in B_{LS}, and with Γ at the leaves of the search tree, determines the satisfiability of Φ.

6. LSR Backdoor: Consider a sub-solver Γ then a collection of variables $BD_{LSR} \subseteq Z$ is known as the LS backdoor with respect to the sub-solver Γ if and only if there is a tree search investigation order alongside renew s.t. a SAT solver

based on CDCL branches exclusively upon the variables in B_{LSR}, and with Γ at the leaves of the tree search, answers the satisfiability of Φ.

The Satisfiability Problem (SAT) is a fundamental challenge in computer science, highlighting the difficulty of finding efficient solutions for NP-complete problems. Researchers have identified structural properties, such as the backbone and backdoors, that significantly influence the efficiency of SAT solvers, particularly advanced ones like Conflict-Driven Clause Learning (CDCL) solvers. The backbone consists of variables that maintain consistent values across all satisfying assignments, indicating a highly constrained structure. Identifying these variables involves analyzing the solution space to determine which variables remain invariant. Although computationally intensive, this process provides valuable insights into the problem's complexity. SAT instances with a large backbone tend to be more challenging to solve because the fixed variables impose rigid constraints, reducing the solver's flexibility. Recognizing backbone variables allows solvers to concentrate on the more flexible parts of the problem, potentially narrowing the search space and enhancing efficiency.

Backdoors represent another strategic approach to optimizing SAT solvers. A backdoor is a small subset of variables that, when assigned appropriate values, significantly simplify the SAT instance, often transforming it into a problem solvable in polynomial time. Strong backdoors simplify the problem to a polynomial-time solvable instance regardless of the specific values assigned to the backdoor variables. In contrast, weak backdoors only simplify the problem under certain assignments, making the problem easier to solve only for specific values of the backdoor variables.

LS (Literal-Setting) and LSR (Literal-Setting Reduced) backdoors are specialized types used in specific solving strategies. LS backdoors are subsets of variables that, once set, allow the remainder of the problem to be solved efficiently using a particular algorithm, typically a polynomial-time procedure. LSR

backdoors identify the smallest subset of variables that achieve this simplification, minimizing the number of variables that need to be fixed to transform the problem into an easier instance.

Integrating the identification and exploitation of backdoors, including LS and LSR backdoors, can significantly enhance the performance of CDCL solvers. CDCL solvers iteratively refine their search process through conflict analysis and clause learning. By incorporating backdoor detection, CDCL solvers can prioritize the assignment of backdoor variables, effectively reducing the problem's complexity early in the solving process. This strategy allows the solver to bypass large portions of the search space, focusing computational resources on the most promising areas and improving convergence rates. For instance, when a strong backdoor is identified, the solver can confidently fix the values of the backdoor variables, knowing that the remaining subproblem is tractable. Conversely, when dealing with weak backdoors, the solver might employ heuristic methods to explore different assignments and determine the most effective ones for simplifying the problem.

The integration of backbone and backdoor analysis within the CDCL framework represents a sophisticated approach to SAT solving. By leveraging these structural properties, solvers can gain a deeper understanding of the problem's inherent complexity and devise more targeted strategies for exploration. This holistic approach not only improves the efficiency of the solving process but also enhances the solver's capability to handle larger and more complex SAT instances. Additionally, the continuous development of techniques for detecting and utilizing backdoors, such as dynamic backdoor identification—where the backdoor set can adapt as the solver progresses—holds promise for further advancements in SAT solver technology.

In conclusion, the concepts of the backbone and backdoors, including strong, weak, LS, and LSR backdoors, are pivotal in the study and application of SAT

solving. By focusing on these structural properties, researchers and practitioners can develop more powerful and efficient solvers. The strategic integration of backbone and backdoor insights within the CDCL framework exemplifies how theoretical understanding can drive practical improvements, ultimately enabling solvers to tackle a wider range of SAT problems with greater success.

Small-World: A mathematical graph's closeness ratio, clustering coefficient, and characteristic path length define its small-world attribute. The following formulas hold given a boolean expression Φ(say) in CNF form defined on a collection of variables Z with the clauses K, and given a method of encode GVI, $H(Y, v)$ of Φ:

1. Characteristic Path Length: The characteristic path length in short CPL is defined as the mean of the shortest diameter of all the pairs of z_i and z_j, mathematically:
$$CPL = \frac{\sum_i \sum_j d_{z_i,z_j}}{|Z|(|Z|-1)},$$
where $|Z|$ denotes total vertices present in graph H.

2. Vertex Clustering Coefficient (CLC): The ratio of a vertex z_i's degree to the number of total edges separating it from its surrounding vertex is its clustering coefficient, and it can be calculated as follows:
$$CLC_{z_i} = \frac{2L_{z_i}}{\deg_{z_i}(\deg_{z_i}-1)},$$
where the term L_{z_i} is used for showing the count of edges between the $\deg_{z_i}$ surrounding vertex of z_i.

3. Graph Clustering Coefficient (GCL): The mean clustering of all the variables in a graph $H(Y, v)$ is its Clustering Coefficient (GCL_H), which may be found as follows:
$$CLC = \frac{\sum_i GCL_{z_i}}{|Z|(|Z|-1)}.$$

4. Proximity Ratio (Pr): Pr is defined mathematically as follows:

$$Pr = \frac{CLC \times CPL_{rand}}{CPL \times CLC_{rand}},$$

where CPL_{rand} and CLC_{rand} are the characteristic path length and clustering coefficient of a random graph with the same number of variables and clauses in Φ, respectively.

5. Small-world: G is known as a small-world topology if and only if $Pr >> 1$.

2.0.1 Scale-Free

Many graphs in the actual world have been found to have a scale-free structure. Significant variation in the node arity, which appears to follow an exponential distribution, is a feature of these graphs. The following defines the scale-free property.
G is scale-free for the graph $H(Y, v)$ if and only if following are true:

1. Nodes' arity is determined by a randomly generated variable M that has a distribution of law of power, mathematically:

$$P(M = m) = m^{-\delta} \text{ such that } \delta \in [2, 3].$$

2. It is a self-similar distribution.

Uniform random $3 - SAT$ formulae, on the other hand, follow an Erdós-Rényi graph structure and are not scale-free.

2.0.2 Width of tree

A graph's treelikeness is determined by its widh of tree; the lower the width of tree, the graph is very similar to a tree. Consider a boolean expression Φ in CNF form defined on the collection of variables Z with clauses from the

collecction K transformed to an undirected graph $H(Y, v)$ whose vertices are $Z = \{z_1, z_2, \ldots, z_n\}$.

1. Decomposition: $D(\beta, T)$ is the tree decomposition of the graph H, where $\beta = \{B_1, B_2, \ldots, B_n\}$ is the collection of boxes and P with the elements of β forms a tree as its vertices such that:

   (a) All the elements of β together gives us Z.

   (b) $\forall$ edges $(z_i, z_j) \in Z$, $\exists k \in \mathbb{N}$ with $(z_i, z_j) \in B_K$

   (c) We can make a tree P with the elements of β.

2. Width of a Decomposition: The decomposition $D(\beta, T)$ of a tree P has the following width:

$$\Psi_D = \max_k\{|B_k| - 1\}.$$

3. Width of tree: The minimal width over all possible tree decompositions of the graph H with the following tree decompositions: $D_1, D_2, \ldots, D_n$. is defined as the width of tree of the graph H, given by

$$T_H = \min_i\{\Psi_{D_i}\}.$$

2.0.3 Centrality

The term "centrality" describes a node's relative importance in a graph. Central nodes can be defined in several ways, such as betweenness centrality and eigenvector centrality. Eigenvector centrality is a colloquial term that combines neighbor importance with degree centrality. The adjacency matrix is used to monitor neighbors. Below is a formal definition:

1. Eigenvector Centrality: For the graph $H(Y, v)$ and $p, q \in Z$, let $A = (ap, q)$ be defined as the adjacency matrix. If p is a neighbor of q then $a_{p,q} = 1$;

otherwise 0. We can also define the relative centrality of variable p as:

$$u_p = \frac{1}{\Lambda} \sum_{q \in V(p)} u_q$$
$$= \sum_{q \in H} a_{p,q} u_q,$$

where $V(p)$ is denoting the set of neighbors of p and Λ is any constant. The vector notation of the above equation can be written as:

$$Au = \Lambda p.$$

Katsirelos and Simon [**14**] initially discovered a correlation between centrality and certain features of how CDCL solvers behaved during the search, especially the branching heuristic. Their experimental findings indicated that, in comparison to unselected variables, the majority of decision variables have a higher average centrality. It was found that influential variables are more apparent in the neighborhood and rich-get-richer distributions, while all variables typically have equal influence on the uniform random distribution. Betweenness centrality, proposed by Jamali and Mitchell [**12**], is a variation of centrality defined as follows:

2. Betweennes Centrality: The idea of breaking down SAT cases into connected components was first presented and put to use by Biere and Sinz [**15**], however their experimental research revealed that this is insufficient to solve SAT instances quickly. Compared to linked components, the concept of community is more universal. To find communities, a quantitative metric known as modularity is utilized. The following equations hold given a boolean expression Φ in CNF form defined on the set of variables Z with the clauses K, and transformed in the GVI $H(Y, v)$ of Φ, and a set of box $\beta = B_1, B_2, \ldots, B_n$ on G:

$$BE_{z_i} = \sum_{(j,m), i \neq j \neq k} \frac{p_{z_j, z_m}(z_i)}{p_{z_j, z_m}},$$

where $p_{z_j,z_m}(z_i)$ is used for the number of the paths passing through the node z_i and p_{z_j,z_m} denotes the total number of shortest paths from z_j to z_m.

3. Community: The idea of breaking down SAT cases into connected components was first presented and put to use by Biere and Sinz [**15**]. However, their experimental research revealed that this is insufficient to solve SAT instances effectively. Compared to linked components, the concept of community is more universal. To find communities, a quantitative metric known as modularity is utilized.

   The following equations hold given a boolean expression Φ in CNF form defined over the set of variables Z and clauses K, an transformed GVI of Φ, $H(Y, v)$, and a box set $\beta = \{B_1, B_2, \ldots, B_n\}$ on H:

4. Modularity: Let the set of least number of boxes covering the graph $H(Y, v)$ is β. Modularity of R is given as:

$$R(H, \beta) = \sum_{B_i \in \beta} \left(\frac{\sum_{z_i, z_j \in B_i} v(z_i, z_j)}{\sum_{z_i, z_j \in Z} v(z_i, z_j)} - \left(\frac{\sum_{z_i \in B_i} \deg_{z_i}}{\sum_{z_i \in Z} \deg_{z_i}} \right)^2 \right).$$

5. Clear Community: If the modularity value of $Q(H, \beta)$ is greater than or equal to 0.7 i.e. $Q(H, \beta) \geq 0.7$ then the graph H has a clear community.

6. Hierarchical Community Graph: A graph $H(Y, v)$ can be recursively divided into subgraphs, forming a hierarchical community decomposition. T_H is the tree that depicts the hierarchical structure of communities in graph H, where node set C is defined as $C_1, C_2, \ldots, C_k$, and depth dep_H. The community depth and degree for a node $C_i \in C$ are shown by dep_{C_i} and $\deg_{C_i}$, respectively. A community of leaflets C_f is one with $\deg(C_f) = 0$. The community of i^{th} number within the l^{th} level is represented by $C_i^l \subseteq Z$. More specifically, H's initial set of vertices is represented by C_1^1.

7. Self-Similarity: Phenomenon of self-similarity represents a prevalent attribute evident in numerous real-world graphs. In essence, a graph exhibits

self-similarity when it upholds a consistent structure following rescaling, which involves the replacement of groups of nodes with a single node. Consider a CNF formula Φ defined with the variables form the set Z and the clauses K, and its transformation as a Variable Incidence Graph (GVI) $H(Z, v)$ with a set of boxes $\beta = \{B_1, B_2, ..., B_n\}$ on H, we can define the property of the self-similar as follows:

- Self-similar: Let $c(s)$ represent the bare minimum size s boxes needed to cover H. H resembles herself. if, for every value Λ, $\beta(s) \sim s^{-\Lambda}$, meaning that $\beta(s)$ reduces polynomially. The fractal dimension of H is denoted by Λ.

8. Entropy: Researchers have explored the entropy measure as a means of assessing the complexity of instances of SAT when viewed as graphs. The measure of uncertainty in random systems is expressed by entropy. The embedded system's unpredictability increases with increasing entropy. Below, you will find the formal definition of entropy, also referred to as entropy of one-dimensional and entropy of two-dimensional.

9. One-dimensional Entropy: For the graph $H(Y, v)$ we can define the structural entropy of one-dimensional as:

$$\mathcal{H}^1(H) = -\sum_{i=1, x_i \in Z}^{|Z|} \frac{\deg_{z_i}}{\text{vol}_H} \log_2 \frac{\deg_{z_i}}{\text{vol}_H},$$

where $\deg_{z_i}$ is the degree of z_i and vol_H is the volume of the graph.

10. Entropy of One-dimensional Entropy on the collection of Boxes: Consider a disjoint set of boxes $\beta = B_1, B_2, \ldots, B_n$ on H. Then over this set of boxes, we can define the entropy of one dimension of $H(Y, v)$ as follows:

$$\mathcal{H}^B(H),$$

where E_{B_i} is used for showing the total count of edges of the box B_i.

11. Entropy of Two-dimensional: For the graph $H(Y, v)$ we can define the structural entropy in two-dimensional as:

$$\mathcal{H}^2(H) = \min_B \mathcal{H}^\beta(H).$$

Research involving study of uniformly generated random 3-SAT and instances of industry has demonstrated a proportional relationship between entropy and problem hardness. Furthermore, the structural entropy of a variable directly influences its probability of being flipped, as detailed in Table 4.

CHAPTER 3

Analysis

The thesis presents a comprehensive survey of the Satisfiability Problem (SAT), emphasizing its structural aspects and implications within theoretical computer science and practical applications. The analysis delves into various dimensions of the SAT problem, providing insights into its complexity, algorithmic solutions, and the influence of different structural properties on its solvability.

Structural Complexity of SAT: The thesis begins by examining the intrinsic complexity of SAT, one of the first problems proven to be NP-complete. It highlights the problem's foundational role in computational complexity theory, serving as a benchmark for classifying other problems within the NP class. The survey underscores the pivotal significance of Cook's Theorem, which established SAT's NP-completeness, and explores subsequent research that has extended and deepened our understanding of SAT's complexity.

Algorithmic Approaches: A substantial portion of the thesis is dedicated to exploring various algorithmic strategies for solving SAT. This includes

an analysis of classical algorithms such as the Davis-Putnam-Logemann-Loveland (DPLL) algorithm and its modern enhancements like Conflict-Driven Clause Learning (CDCL). The discussion extends to heuristic methods, such as stochastic local search algorithms, which offer practical solutions for large instances of SAT despite the problem's theoretical intractability. By comparing these algorithms, the thesis elucidates the trade-offs between exact and heuristic approaches, highlighting their respective strengths and limitations.

Structural Properties and SAT: The thesis provides an in-depth analysis of how certain structural properties of SAT instances influence their solvability. Key concepts such as clause-to-variable ratios, phase transitions, and the role of symmetries are explored. The survey discusses empirical studies and theoretical models that demonstrate how these properties can predict the difficulty of SAT instances. For instance, it examines the critical threshold phenomenon, where the satisfiability of random SAT instances undergoes a sharp transition, and how this insight guides the design of more efficient algorithms.

Practical Applications: Beyond theoretical considerations, the thesis also addresses the practical implications of SAT and its variants, such as 3-SAT and k-SAT, in various domains. It reviews applications in fields like hardware and software verification, artificial intelligence, and operations research. By demonstrating the utility of SAT solvers in these areas, the thesis underscores the real-world relevance of studying the SAT problem's structure.

Advances and Open Problems: The survey concludes by highlighting recent advances in SAT research and identifying open problems that continue to challenge researchers. This includes ongoing efforts to improve SAT solver performance, the exploration of new algorithmic paradigms, and the quest to better understand the fine-grained complexity of SAT and related problems. The thesis emphasizes the dynamic nature of SAT research, driven by both theoretical breakthroughs and practical demands.

Conclusion: Overall, the thesis provides a thorough and nuanced survey of the structure of the Boolean Satisfiability Problem, blending theoretical insights with practical considerations. By dissecting the complexity, algorithmic strategies, structural properties, and applications of SAT, the survey offers a holistic view of a problem that lies at the heart of computational complexity theory and has profound implications across various technological domains.

CHAPTER 4

Challenging Part & Future Aim

4.1 Challenging Part:

One of the most challenging aspects of the Boolean Satisfiability Problem (SAT) lies in its inherent computational complexity. As an NP-complete problem, SAT epitomizes the difficulty of finding efficient solutions for large instances. This complexity manifests in several ways:

1. Scalability of Algorithms: While significant progress has been made in developing efficient SAT solvers, scalability remains a major challenge. Algorithms that perform well on small to moderately sized instances often struggle with the exponential growth in complexity as the size of the input increases. The DPLL and CDCL algorithms, for instance, while powerful, can face significant performance bottlenecks on larger or more complex SAT instances.

2. Phase Transition Phenomenon: The critical threshold phenomenon, where

the satisfiability of random SAT instances sharply transitions from satisfiable to unsatisfiable as the clause-to-variable ratio changes, poses a unique challenge. Understanding and predicting this transition requires deep theoretical insights and sophisticated empirical analysis, making it difficult to generalize findings across different SAT instances.

3. Structure and Symmetry: Identifying and exploiting structural properties and symmetries within SAT instances can greatly enhance solver efficiency. However, this task is inherently complex and often problem-specific. Developing general methods to detect and leverage these properties remains an ongoing challenge.

4. Heuristics and Heuristic-Based Methods: While heuristic methods provide practical solutions for many SAT instances, their unpredictable performance and lack of guaranteed optimality are significant drawbacks. Designing heuristics that are both effective and reliable across a wide range of instances is an ongoing area of research.

5. Real-World Applications: Applying SAT solvers to real-world problems often introduces additional layers of complexity. These problems may involve constraints and requirements that are not present in theoretical SAT formulations, necessitating customized or hybrid approaches that blend SAT solving with other techniques.

4.2 Future Aims:

The future aims of research into the structure of the Boolean Satisfiability Problem encompass several ambitious goals:

1. Enhanced Algorithmic Performance: One of the primary aims is to develop more advanced SAT solvers that can handle larger and more complex in-

stances efficiently. This involves improving existing algorithms, like CDCL, and exploring new algorithmic paradigms that can offer better performance.

2. Deeper Understanding of Phase Transitions: Future research aims to deepen the understanding of the phase transition phenomenon in SAT. By developing more precise theoretical models and conducting extensive empirical studies, researchers hope to better predict and exploit these transitions to improve solver performance.

3. Exploiting Structural Properties: Another key aim is to develop generalized methods for identifying and utilizing structural properties and symmetries in SAT instances. This includes advancing techniques for symmetry breaking and structural decomposition, which can significantly enhance the efficiency of SAT solvers.

4. Integration with Other Techniques: Combining SAT solving with other computational techniques, such as constraint programming and integer programming, is a promising area of future research. This hybrid approach could leverage the strengths of multiple paradigms to tackle complex, real-world problems more effectively.

5. Machine Learning and SAT: Incorporating machine learning techniques to predict the difficulty of SAT instances and guide the search process in solvers is a cutting-edge aim. Machine learning models can potentially learn from large datasets of SAT instances to provide insights and heuristics that improve solver performance.

6. Application-Specific Solvers: Developing SAT solvers tailored to specific application domains, such as hardware verification, artificial intelligence, and bioinformatics, is a practical aim. Customized solvers can exploit domain-specific knowledge to achieve better performance and reliability.

7. Benchmarking and Standardization: Establishing comprehensive benchmarks and standardized testing frameworks for SAT solvers is crucial for comparing and evaluating different approaches. Future efforts aim to create more robust and diverse benchmarks that reflect the wide range of SAT instances encountered in practice.

   By addressing these challenges and pursuing these future aims, research into the Boolean Satisfiability Problem can continue to advance, driving both theoretical understanding and practical applications forward.

Table 1

Benchmarks	Solvers	Results			
		random	Crafted	Industrial	metric
Uniform $3-SAT$ instances	SATz	No	NS	Yes.	Evidence
Industrial 2002-2005 SATRaces	CDCL: MiniSAT	NS	NS	NS	CPU time
		Positive	NS	Positive	Clause learning

Table 2

Benchmarks	Solvers	Results			
		random	Crafted	Industrial	metric
Industrial 2009	PrecoSAT	NS	NS	NS	Evidence
		NS	NS	-	CPU time
		NS	NS	NS	Clause learning
Uniform 2007/2009 SAT competitions	MapleCOMSPs	Yes	Yes	Yes.	Evidence
Crafted 2009-2014		-	-	-	CPU time
Industrial 2009-2014		NS	NS	NS	Clause learning

Table 3

Benchmarks	Solvers	Results			
		random	Crafted	Industrial	metric
Uniform $3-SAT$ instances	CDCL SAT solver: Picosat	No	Yes	Yes.	Evidence
Industrial 2018		NS	NS	NS	CPU time
		-	inverse	inverse	Clause learning

Table 4

Benchmarks	Solvers	Results			
		random	Crafted	Industrial	metric
Uniform $3-SAT$ examples	CCAsat	Yes	NS	Yes.	Evidence
Industrial 2018	Sparrow 2018	NS	NS	NS	CPU time
		NS	NS	NS	Clause learning

Note: NS indicates that the metric has not been studied and the symbol - denotes no reaction.

Bibliography

[1] Tasniem Nasser Alyahya, Mohamed El Bachir Menai, Hassan Mathkour.,*"On the Structure of the Boolean Satisfiability Problem: A Survey."* In: ACM Computing Surveys (CSUR) (2022).

[2] C. Ansótegui, M. L. Bonet, Jesús Giráldez-Cru, and Jordi Levy. 2014., *"Improving two-mode algorithm via probabilistic selection for solving satisfiability problem."* In: Automated Reasoning, Stéphane Demri, Deepak Kapur, and Christoph Weidenbach (Eds.). Springer International Publishing, Cham, 107–121.

[3] Masina, Gabriele and Spallitta, Giuseppe and Sebastiani, Roberto, *"Community structure in industrial SAT instances."* In: Journal of Artificial Intelligence Research 66 (2019), 443–472.

[4] Carlos Ansótegui, Jesús Giráldez-Cru, and Jordi Levy. 2012., *"The community structure of SAT formulas."* In: Theory and Applications of Satisfiability Testing–SAT 2012, Alessandro Cimatti and Roberto Sebastiani (Eds.). Springer Berlin, Berlin, 410–423.

[5] Carlos Ansótegui, Maria Bonet, Jesús Giráldez-Cru, and Jordi Levy, *"Structure features for SAT instances classification. "* In: Applied Logic 23 (2017), 27–39.

[6] Chunxiao Li, Jonathan Chung, Soham Mukherjee, Marc Vinyals, Noah Fleming, Antonina Kolokolova, Alice Mu, and Vijay Ganesh., *" On the hierarchical community structure of practical Boolean formulas."* In: Springer International Publishing, Cham, 359–376.

[7] Edward Zulkoski, Ruben Martins, Christoph M. Wintersteiger, Jia Hui Liang, Krzysztof Czarnecki, and Vijay Ganesh. 2018.,*" The effect of structural measures and merges on SAT solver performance."* In: CP (Lecture Notes in Computer Science), Vol. 11008. Springer, 436–452.

[8] Edward Zulkoski, Ruben Martins, Christoph M. Wintersteiger, Robert Robere, Jia Hui Liang, Krzysztof Czarnecki, and Vijay Ganesh.,*"Relating complexity-theoretic parameters with SAT solver performance."* In: arXiv preprint arXiv:1706.08611 (February 2017)

[9] Nathan Mull, Daniel J. Fremont, and Sanjit A. Seshia.,*"On the hardness of SAT with community structure."* In: Theory and Applications of Satisfiability Testing–SAT 2016, Nadia Creignou and Daniel Le Berre (Eds.). Springer International Publishing, Cham, 141–159.

[10] Knot Pipatsrisawat and Adnan Darwiche.,*"A lightweight component caching scheme for satisfiability solvers."* In: Theory and Applications of Satisfiability Testing–SAT 2007, João Marques-Silva and Karem A. Sakallah (Eds.). Springer Berlin, Berlin, 294–299.

[11] Peter Cheeseman, Bob Kanefsky, and William M. Taylor. , *" Where the really hard problems are. "* In: Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1 (IJCAI'91).

Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 331–337. http://dl.acm.org/citation.cfm?id=1631171.1631221.

[12] Sima Jamali and David Mitchell., *"Improving SAT solver performance with structure-based preferential bumping. "* In: GCAI 2017. 3rd Global Conference on Artificial Intelligence (EPiC Series in Computing), Christoph Benzmüller, Christine Lisetti, and Martin Theobald (Eds.), Vol. 50. EasyChair, 175–187.

[13] Peter Gregory, Maria Fox, and Derek Long. 2008, *"A new empirical study of weak backdoors. "* In: Principles and Practice of Constraint Programming, Peter J. Stuckey (Ed.). Springer Berlin, Berlin, 618–623.

[14] George Katsirelos and Laurent Simon. 2012., *"Eigenvector centrality in industrial SAT instances. "* In: CP (Lecture Notes in Computer Science), Vol. 7514. Springer, 348–356.

[15] Armin Biere and Carsten Sinz. 2006., *"Decomposing SAT problems into connected components."* JSAT 2, 1–4 (2006), 201–208.