Data Reductions, Fixed Parameter Tractability, and Random Weighted \(d\)-CNF Satisfiability

Yong Gao

Department of Computer Science, Irving K. Barber School of Arts and Sciences
University of British Columbia Okanagan, Kelowna, Canada V1V 1V7

Abstract

Data reduction is a key technique in the study of fixed parameter algorithms. In the AI literature, pruning techniques based on simple and efficient-to-implement reduction rules have also played a crucial role in the success of many industrial-strength solvers. Examples include unit propagation in satisfiability testing, constraint propagation in constraint programming, and heuristic functions in state space search. Understanding the effectiveness and the applicability of data reduction as a technique for designing heuristics for intractable problems has attracted much interest in the fields of AI and algorithmics, and is one of the main motivations in the general interest in the phase transition behavior of randomly-generated NP-complete problems.

In this paper, we take the initiative to study the power of data reductions in the context of random instances of a generic intractable parameterized problem, the weighted \(d\)-CNF satisfiability problem. We propose a non-trivial random model for the problem, design and analyze an algorithm that solves the random instances with high probability and in fixed parameter time, establish the exact threshold of the phase transition, and give some analyses on the parametric resolution complexity of unsatisfiable instances. Also discussed is a more general random model and the generalization of the results to this model.

To the best knowledge of the author, our algorithm based on simple data reduction rules for the problem and its analysis provide the first sound theoretical evidence on the effectiveness of simple reduction rules applied to intractable parameterized problem.

* Part of the results appeared in the Proceedings of AAAI’08 [25]. The research is supported by National Science and Engineering Research Council of Canada (NSERC) RGPIN 327587-06

Email address: yong.gao@ubc.ca (Yong Gao).
1 Introduction

The theory of parameterized complexity and fixed-parameter algorithms is becoming an active research area in recent years [22,42]. Parameterized complexity provides a new perspective on hard algorithmic problems, while fixed-parameter algorithms have found applications in a variety of research fields. Parameterized problems also arise in many areas of artificial intelligence research, including satisfiability, automated reasoning, logic programming, constraint programming, and probabilistic inference [8,10,43,44]. See [31,32] for thorough survey of recent literature.

Data reduction is a key technique in designing efficient algorithms for fixed parameter tractable problem [42] and exact exponential-time algorithms for NP-hard problems [24]. In several areas of artificial intelligence (AI), pruning techniques based on simple reduction rules have also been widely used, most notably in satisfiability testing and constraint processing where backtracking interleaved with the use of highly-efficient implementation of unit propagation and consistency propagation has played a key role in the success of most industrial-strength solvers, and in heuristic state space search where a heuristic function is usually employed to reduce the search space. The power of data reductions have also been demonstrated empirically for many NP-hard problems or intractable parameterized problems such as clique cover [33], dominating set [4], road network related problem [46], and Hamiltonian cycle [45]. Experiments also revealed that simple data reduction rules usually have a much better performance than those predicted by theoretical analyses [11,33,36].

Despite the many success stories, our understanding of the effectiveness and the applicability of data reduction as a technique for designing heuristics for intractable problems is far from complete. The continued interest over the past more than ten years in the study of the phase transition phenomenon of random instances of NP-complete problems is largely motivated by the expectation that such study will help shed lights on why simple heuristics work on typical problem instances and in what situations. See [1,7,14,17,18,27–30,34,47] and the references therein. Recently, the problem of detecting backdoor sets has also attracted much attention. The existence of small-sized backdoors naturally leads to efficient algorithms for problems that are otherwise hard to solve. While the backdoor detection problem are NP-complete and/or fixed-parameter intractable for many types of backdoors, practical SAT-solvers have been found to be able to exploit the existence of small-sized backdoors effectively [21,44].

In this work, we take the initiative to extend this line of research on phase transitions to intractable parameterized problems. We hope that the current
work on the fixed-parameter tractability of random instances of intractable parameterized problems, together with the previous work on the typical-case behavior of random NP-complete problems, will help shed further light on the power of data-reduction based heuristics and on the hardness of detecting small-sized backdoors.

We study random instances of the weighted $d$-CNF satisfiability problem (WEIGHTED $d$-SAT)\(^1\). An instance of the problem consists of a $d$-CNF formula $F$ and a fixed parameter $k > 0$. The question is to decide if there is a satisfying assignment that has a Hamming distance $k$ to the all-zero assignment.

A random instance of weighted $d$-CNF satisfiability consists of a fixed parameter $k$ and a random $d$-CNF formula $F_{k,d}^{n,p}$ generated as follows: for each subset of $d$ variables and with probability $p = p(n)$, a clause over the $d$ variables is selected uniformly at random from the set of $2^d - 1$ possible clauses that contain at least one negated literals. This random model of CNF formulas is similar to the various random models used in the study of the standard propositional satisfiability problem. Due to a recent result of Marx [37] on the complexity of parameterized Boolean constraint satisfaction problems, forbidding clauses that contain positive literals only is not a serious restriction as far as the parameterized tractability is concerned. A more detailed argument about this will be given in Section 2.

We propose and analyze an algorithm based on the technique of data reductions and show that the algorithm solves random instances from $F_{k,d}^{n,p}$ with high probability and in fixed parameter time $O(d^k n^2)$. The algorithm and other results obtained in this paper give an almost complete characterization of the typical-case behavior of the random instances of WEIGHTED $d$-SAT, including the exact threshold of the phase transition of the random model, the parametric resolution complexity of unsatisfiable instances.

We also discussed a more general model $F_{k,d}^{n,p}(d')$, $1 < d' < d$, where clauses containing less than $d'$ negated literals are forbidden. Except for the exact threshold of the phase transition and the complexity of typical instances for certain range of $p$, this model seems to pose an interesting challenge for researchers in the areas of AI and theoretical computer science.

To the best knowledge of the author, our algorithm and its analysis give the first sound theoretical evidence on the power of simple data reductions applied to intractable parameterized problems, and this work is the first in the literature on the fixed-parameter tractability of random instances of intractable pa-

\(^1\)In AI literature, $k$ is usually used for clause size. Unfortunately, in the study of parameterized algorithms, $k$ is always reserved for the parameter. We decide to use $k$ as the parameter and use $d$ for the clause size.
rameterized problems. We expect that the proposed random models and their analysis will help facilitate future studies on concrete parameterized problems such as the backdoor detection problem, on the solution space geometry of standard satisfiability problem, and on the design of more effective neighborhood operators for local search algorithms.

1.1 Main Results

The main results of this paper include (1) an algorithm and its analysis showing that instances from the random distribution $\mathcal{F}_{k,d}^{n,p}$ of WEIGHTED $d$-SAT are “typically” fixed-parameter tractable for any clause-probability $p(n)$, in particular, for $p = \frac{c \log n}{n^{d-1}}$ where $c > 0$ is a constant, (2) the exact threshold of the phase transition of $\mathcal{F}_{k,d}^{n,p}$, (3) results on the parametric resolution complexity of random unsatisfiable instances, and (4) extension of some of the above results to a more general model $\mathcal{F}_{k,d}^{n,p}(d')$.

**Theorem 1** There is an $O(d^k n^d)$-time algorithm that solves with high probability a random instance $(\mathcal{F}_{k,d}^{n,p}, k)$ of WEIGHTED $d$-SAT for any $p = p(n) \leq 1$.

The exponent $d$ in the term $n^d$ is mainly due to the consideration that the formula may have $O(n^d)$ clauses since the result applies to any $p(n)$. The algorithm is still a fixed parameter algorithm since $k$, instead of $d$ (the clause size), is the parameter. For sparser formulas, the exponent can be decreased accordingly. For example, in the case of $p(n) = O(\frac{\log n}{n^{d-1}})$, the term $n^d$ can be reduced to $O(n^2 \log n)$.

The fixed-paramter part $d^k$ is due to the fact that the algorithm needs to solve subproblems that are equivalent to the parameterized $(d-1)$-hitting set problem. In the case of $d = 2$, the term $d^k$ can be replaced by $k$. As the proof of Theorem 1 shows, if every variable is only involved in $l$ clauses, the term $d^k$ can be replaced by the running time $O(\binom{l}{k})$ or $O(2^{kd})$ of brute-force methods for these subproblems. Depending on the relation between $l$, $k$, and $n$, these bound may be worse than $O(d^k)$, such as the case of $p(n) \in \Omega(\frac{k \log n}{n^{d-1} d!})$, or better than $O(d^k)$, such as the case of $p(n) \leq \frac{\log n}{n^{d-1}}$.

For random instances of the so-called renormalized version of WEIGHTED $d$-SAT where the question is to find a satisfying assignment of weight $k \log n$, we show that the same algorithm in the above theorem still works if the clause probability $p(n)$ is in a certain range:

**Corollary 1.1** There is an $O(d^k n^d)$ algorithm that solves with high probability a random instance $(\mathcal{F}_{k,d}^{n,p}, k)$ of the renormalized version of WEIGHTED $d$-SAT for any $p = \frac{c \log n}{n^{d-1}}$ with $c > k(2^d - 1)(d - 1)!$. 

4
To get a clear picture of the behavior of random instances of $F_{k,d}^{n,p}$, it is helpful to understand the phase transition of their solution probability. We establish the following exact threshold of the phase transition. Note that the threshold does not depend on the fixed parameter $k$, which is somewhat surprising.

Theorem 2 Let $p = \frac{c \log n}{n^{d-1}}$ with $c > 0$ being a constant. Consider a random instance $(F_{k,d}^{n,p}, k)$ of WEIGHTED $d$-SAT (or its renormalized version). We have

$$\lim_{n} \mathbb{P}\{F_{k,d}^{n,p} \text{ is satisfiable}\} = \begin{cases} 1, & \text{if } c < c^*, \\ 0, & \text{if } c > c^*, \end{cases}$$

where $c^* = (2^d - 1)(d-1)!$. For the case of $d = 2$ and $d = 3$, the thresholds are respectively $c^* = 3$ and $c^* = 14$.

For unsatisfiable instances, it is not clear whether or not the fixed-parameter algorithm in Theorem 1 can be simulated by a resolution proof. But a similar idea does lead to the following

Theorem 3 Assume that $p \geq \frac{c \log n}{n^{d-1}}$ with $c > 2(2^d - 1)(d-1)!$. With high probability, a random instance $(F_{k,d}^{n,p}, k)$ of WEIGHTED $d$-SAT has a resolution proof of size $O(d^k n)$, and can be constructed in $O(d^k n^{O(1)})$ time.

For the case of WEIGHTED 2-SAT and its renormalized version, random unsatisfiable instances can be shown to have a resolution proof of size $O(kn)$:

Theorem 4 For $F_{k,2}^{n,p}$ where $p = \frac{3(\log n + c_1 \log \log n)}{n}$ with $c_1 > k - 1$, a parametric resolution proof of size $O(kn)$ exists with high probability and can be constructed in $O(kn^{O(1)})$ time.

In Table 1, we summarize the results obtained in this paper on the behavior of $F_{k,d}^{n,p}$.

1.2 Outline

The next section contains preliminaries, a detailed description of the random model $F_{k,d}^{n,p}$, and related work. In Section 3, we present the details of our fixed-parameter algorithm, W-SAT. In Section 4, we prove that the algorithm W-SAT succeeds with high probability for random instances of WEIGHTED $d$-SAT. In Section 5, we establish the exact threshold of the phase transition of the solution probability. In Section 6, we present results on the resolution complexity of random instances of WEIGHTED $d$-SAT. In Section 7, we discuss
Table 1
The behavior of random instances from $\mathcal{F}_{k,d}^{n,p}$. A mark $\sqrt{\text{ }}$ indicates that the case is completely resolved. A question mark indicates that the case is completely unknown. $c^*$ is the threshold for the corresponding model.

<table>
<thead>
<tr>
<th>Model Parameters</th>
<th>Threshold</th>
<th>FPT?</th>
<th>FPT Proof Size?</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEIGHTED d-SAT $\mathcal{F}_{k,d}^{n,p}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d = 2$</td>
<td>$\sqrt{\text{ }}$ (Th. 2)</td>
<td>$\sqrt{\text{ }}$ (Th. 1)</td>
<td>$\sqrt{\text{ }}$ (Th. 4)</td>
</tr>
<tr>
<td>$d &gt; 2$</td>
<td>$\sqrt{\text{ }}$ (Th. 2)</td>
<td>$\sqrt{\text{ }}$ (Th. 1)</td>
<td>$c &gt; 2c^*$ (Th. 3)</td>
</tr>
<tr>
<td>Renormalized Version of WEIGHTED d-SAT $\mathcal{F}_{k,d}^{n,p}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d = 2$</td>
<td>$\sqrt{\text{ }}$ (Th. 2)</td>
<td>$c &gt; c^*$ (Th. 4)</td>
<td>$c &gt; c^*$ (Th. 4)</td>
</tr>
<tr>
<td>$d &gt; 2$</td>
<td>$\sqrt{\text{ }}$ (Th. 2)</td>
<td>$c &gt; kc^*$ (Corol. 1.1)</td>
<td>?</td>
</tr>
<tr>
<td>WEIGHTED d-SAT $\mathcal{F}_{k,d}^{n,p}(d')$, $1 &lt; d' &lt; d$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d = 2$</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>$d &gt; 2$</td>
<td>$\sqrt{\text{ }}$ (Th. 5)</td>
<td>$c &gt; 2c^*$ (Th. 6)</td>
<td>$c &gt; 2c^*$ (Th. 6)</td>
</tr>
</tbody>
</table>

how the algorithm and its analysis can be extended to random instances of the renormalized version of WEIGHTED d-SAT. In Section 8, we discuss a generalized model $\mathcal{F}_{k,d}^{n,p}(d')$ and results on its threshold and resolution complexity. In the last Section, we discuss directions for future work.

2 Preliminaries, Random Models, and Related Work

2.1 Parameterized complexity

An instance of a parameterized decision problem [22,42] is a pair $(I,k)$ where $I$ is a problem instance and $k$ is a fixed problem parameter. Usually, the parameter $k$ either specifies the “size” of the desired solution or is related to some structural property of the underlying problem, such as the treewidth of a graph. For example, in an instance $(I,k)$ of the parameterized vertex cover problem, $I$ is a graph and $k$ is the size of the vertex cover. The question is to decide whether $I$ has a vertex cover of size $k$.

Whereas for fixed $k$, the vertex cover problem can be solved by brute-force in $O(n^k)$ time, the theory of parameterized complexity is concerned with whether or not there is an algorithm whose worst-case running time avoids the exponential dependency between the problem size $n$ and the parameter $k$. 

6
A parameterized problem is fixed-parameter tractable (FPT) if there is an algorithm that solves any instance \((I,k)\) in \(f(k)|I|^{O(1)}\) time, where \(f(k)\) is a computable function that depends only on \(k\). For example, it is known that the parameterized vertex cover problem is FPT since it can be solved in \(O(1.28^k + kn)\) time where \(n\) is the number of vertices of the graph [42].

Parameterized problems are inter-related by parameterized reductions, resulting in a classification of parameterized problems into a hierarchy of complexity classes

\[
FPT \subseteq W[1] \subseteq W[2] \cdots \subseteq XP.
\]

At the lowest level is the class of FPT problems. The class \(W[1]\) contains all parameterized problems that can be reduced to the weighted 2-CNF satisfiability problem to be defined in the next subsection. The top level \(XP\) contains all the problems that can be solved in time \(f(k)n^{g(k)}\). It is widely believed that the containment is strict and the notion of completeness can be defined via parameterized reductions [22,42]. Unless FTP = W[1], it is likely that no algorithm for WEIGHTED \(d\)-SAT \((d \geq 2)\) can be more efficient that the \(O(n^k)\)-time brute-force algorithm that enumerates all the \(n^k\) assignments. It can be proved that “FTP = W[1]” implies “P = NP”.

### 2.2 Weighted CNF Satisfiability Problem

As with the theory of NP-completeness, the satisfiability problem plays an important role in the theory of parameterized complexity. A CNF formula over a set of Boolean variables is a conjunction of disjunctions of literals. A \(d\)-clause is a disjunction of \(d\)-literals. A \(d\)-CNF formula is a CNF formula that consists of \(d\)-clauses only.

**Definition 2.1** An assignment to a set of \(n\) Boolean variables is a vector in \(\{TRUE, FALSE\}^n\). The weight of an assignment is the number of the variables that are set to TRUE in the assignment.

It is convenient to identify TRUE with 1 and FALSE with 0. Thus, an assignment can be regarded as a vector in \(\{0,1\}^n\) and the weight of an assignment is just its Hamming distance to the all-zero assignment.

A representative \(W[1]\)-complete problem is the following weighted \(d\)-CNF satisfiability problem [42]:

**Problem 1** \(WEIGHTED d\)-SAT

**Instance:** A CNF formula consisting of a collection of \(d\)-clauses.

**Parameter:** A positive integer \(k\).
**Question:** Is there a satisfying assignment of weight $k$?

Unlike the situation with the standard satisfiability problem, WEIGHTED $d$-SAT is already $W[1]$-complete for $d = 2$, and consequently is intractable from the perspective of parameterized complexity. It is a recent interest to consider the renormalization of a parameterized problem. The renormalized version of WEIGHTED $d$-SAT is as follows [42]:

**Problem 2 MINI-WEIGHTED $d$-SAT**

**Instance:** A CNF formula consisting of a collection of $d$-clauses.

**Parameter:** A positive integer $k$.

**Question:** Is there a satisfying assignment of weight $k \log n$?

### 2.3 Parameterized Proof Systems and Parameterized DPLL Algorithms

The study of the parameterized proof complexity of weighted CNF satisfiability has been recently initiated by Dantchev et al. [20], who established lower bounds on the parameterized resolution proof for CNF formulas that encode some first-order combinatorial principle. A formal definition of a parameterized tree-like resolution proof system for weighted CNF satisfiability is given in [20]. Basically, a parameterized resolution system for an instance of a co-$W[2]$ problem can be regarded as a classical resolution system that has access for free to all clauses with more than $k$ negated variables, where $k$ is the parameter of the weighted satisfiability. For instances of a co-$W[1]$ complete problem, we also need the free access to all clauses that contain more than $n - k$ positive literals since a contradiction for a co-$W[1]$ complete problem such as WEIGHTED $d$-SAT is “the formula doesn’t have a satisfying assignment of weight exactly $k$”.

Accordingly, one can consider the parameterized version of the DPLL algorithm for the satisfiability problem. It proceeds in the same way as the standard DPLL algorithm with the exception that a node in the search tree fails if either

1. a clause has been falsified by the partial assignment, or
2. the number of variables assigned to true in the partial assignment has exceeded $k$. 
2.4 Random Models for WEIGHTED $d$-SAT

We use $G(n, p)$ to denote the Erdős-Rényi random graph where $n$ is the number of vertices, $p$ is the edge probability, and each of the possible $\binom{n}{2}$ edges appears independently with probability $p$. A random hypergraph $\mathcal{G}(n, p, d)$ is a hypergraph where each of the $\binom{n}{d}$ possible hyperedges appears independently with probability $p$. Throughout this paper by “with high probability”, abbreviated as whp, we mean that the probability of the event under consideration is $1 - o(1)$.

We will be working with the following random model for WEIGHTED $d$-SAT. The model is similar in spirit to the well-known random models in the study of the standard (constraint) satisfiability. See [27,30,40] and the references therein.

**Definition 2.2** Let $X = \{x_1, \cdots, x_n\}$ be a set of Boolean variables, $p = p(n)$ be a function of $n$, and $k$ and $d$ be two positive constants. The random model $\mathcal{F}_{k,d}^{n,p}$ for WEIGHTED $d$-SAT parameterized by $k$ is defined as follows:

To generate an instance $\mathcal{F}$ from $\mathcal{F}_{k,d}^{n,p}$, we first construct a random hypergraph $\mathcal{G}(n, p, d)$ using $X$ as the vertex set. For each hyperedge $\{x_{i_1}, \cdots, x_{i_d}\}$ and with probability $p$, we select a $d$-clause uniformly at random from the set of $2^d - 1$ non-monotone $d$-clauses defined over the variables $\{x_{i_1}, \cdots, x_{i_d}\}$. (A monotone clause is a clause that contains positive literals only).

Note that since monotone clauses are forbidden in $\mathcal{F}_{k,d}^{n,p}$, the all-zero assignment is always a satisfying assignment. Also note that the range of the clause probability we are interested in is $p(n) = \Omega\left(\frac{\log n}{nd-1}\right)$. We explain in the following the rationale of doing so:

2.4.1 Trivial Random Instances

We note that without forbidding monotone clauses or for $p(n) = o\left(\frac{\log n}{nd-1}\right)$, random instances of WEIGHTED $d$-SAT are trivial. This can be proved by simple random-graph arguments. We state without proofs the following observations:

**Lemma 2.1** (1) If in $\mathcal{F}_{k,d}^{n,p}$ monotone clauses are not forbidden, then whp there exist $2k$ disjoint monotone clauses unless $p(n)$ is extremely small. Consequently, a random instance is unsatisfiable whp.

(2) If $c$ small enough (for the case of $d = 2$, $c < \frac{3}{2}$) or if $p(n) \in o\left(\frac{\log n}{nd-1}\right)$, random instances from $\mathcal{F}_{k,d}^{n,p}$ are trivially satisfiable due to the existence of a large number of “isolated” variables $x$ that only appear in clauses of the form $x \lor \overline{y}$. Simply setting $k$ of these variables to TRUE and the remaining variables to FALSE gives us a weight-$k$ satisfying assignment.
2.4.2 WEIGHTED \(d - SAT\) Instances Reduce to Instances with Monotone Clauses Forbidden.

In [37], Marx studied the complexity of general parameterized Boolean constraint satisfaction problems, which he calls WEIGHTED \(F\)-SAT. He proved that \(F\)-SAT is fixed-parameter tractable if either every variable has a bounded number of occurrences or the constraints are weakly separable. In the mean time, Marx showed the \(W[1]\)-completeness for the case where the implication clause \(x \rightarrow y\) is the only type of constraints. In proving these results, Marx observed that it is sufficient to consider constraints that are \(\theta\)-valid, i.e., constraints that are always satisfied by the all-zero assignment. We state in the following Marx’s observation in terms of WEIGHTED \(d\)-SAT:

**Lemma 2.2 (Lemma 4.1, Marx [37])** An instance \((F, k)\) of WEIGHTED \(d\)-SAT can be reduced to \(d^k\) WEIGHTED \(d\)-SAT instances in which monotone clauses are forbidden.

In view of the above Lemma, we believe that \(F_{k,d}^{n,p}\) is a natural model that captures the essential characteristics of a WEIGHTED \(d\)-SAT instance.

We also remark that when the clause-probability \(p(n) \in \Theta(\log n)\) as is the case discussed in this paper, the number of variables that have a bounded number of occurrences is in \(o(1)\) whp, so that Marx’s result [37] on the fixed-parameter tractability of \(F\)-SAT with bounded variable occurrences does not apply to the random instances we are dealing with.

2.4.3 Relation to Planted Models for Standard SAT.

Even though a random instance of \(F_{k,d}^{n,p}\) always has the all-zero assignment as a planted solution (in the sense of standard satisfiability), what WEIGHTED \(d\)-SAT asks for is a weight-\(k\) satisfying assignment, i.e., a satisfying assignment with Hamming distance exactly \(k\) to the all-zero assignment. This is the major difference between the WEIGHTED \(d\)-SAT instances considered in this paper and the standard SAT instances with hidden solutions studied in the literature. See, for example, [1,34,47] and the references therein.

2.5 Residual Graphs of CNF Formulas and Induced Formulas

Associated with a CNF formula is its residual graph over the set of variables involved in the formula. There is an edge between two variables if they both occur in some clause. The residual graph of a random instance of \(F_{k,d}^{n,p}\) is exactly the random graph \(G(n, p)\). The residual graph of a random instance of \(F_{k,d}^{n,p}\) is the primal graph of the random hypergraph \(G(n, p, d)\).
Let $F$ be a $d$-CNF formula and $V \subset X$ be a subset of variables. The induced formula $F_V$ over $V$ is defined as a CNF formula $F_V$ that consists of the following two types of clauses:

1. the clauses of $F$ that only involve the variables in $V$;
2. the clauses of size at least 2 obtained by removing any literal whose corresponding variable is in $X \setminus V$.

Note that for the case of 2-CNF formulas, $F_V$ contains clauses of the first type only.

### 2.6 Related Work

There is an extensive literature on the study of the phase transitions of random instances of NP-complete problems, their implication to the design of heuristics and practical solvers, and polynomial-time algorithms that solve random instances whp \cite{2,17,27–30,39,40}. The threshold behavior of random CNF satisfiability, general constraint satisfaction problems, and graph coloring on random graphs has attracted a lot of attention, and the exact thresholds of their phase transitions remain unsettled \cite{2}.

In \cite{45}, a backtracking algorithm with reduction rules based on local vertex degree information is able to solve random instances of the Hamiltonian cycle problem at phase transitions easily. One of the important observations made in the study of the phase transition of hard problems is the existence of backbone variables i.e., variable whose value is fixed in all optimal solutions. In \cite{48}, it is shown that even statistical information on the backbone variables can be utilized to guide the variable selection of a local search algorithm to obtain great performance improvement.

There has been recent interest in random instances with planted solutions \cite{a.k.a. hidden solutions} designing whp polynomial-time algorithms random instances with planted solutions \cite{a.k.a. hidden solutions} \cite{5,15,23,35,16} and using these random instances for empirical studies \cite{1,34,47}. In \cite{27}, random instances of the constraint satisfaction problem are designed that include a hidden consistency core rather than a solution.

Weighted CNF satisfiability is a generic intractable parameterized problem. In addition to the work of Marx \cite{37} we have mentioned in the previous subsection, other work that motivated the current work include the study of the parameterized proof complexity of weighted CNF satisfiability recently initiated by Dantchev et al. \cite{20} and the study of the backdoor set detection problem and its parameterized complexity in the recent artificial intelligence literature \cite{21,44,43}.
In the current work, we take the initiative to extend these lines of research to random instances of intractable parameterized problems. To the best knowledge of the author, this is the first work in the literature that studies the fixed-parameter tractability of random instances of a W[1]-complete problem.

3 A Fixed-Parameter Algorithm for Instances of $\mathcal{F}_{k,d}^{n,p}$

In this section, we present the details of our fixed-parameter algorithm for random instances of $\mathcal{F}_{k,d}^{n,p}$ and analyze its time complexity. The results in this section and in the next section together prove Theorem 1.

3.1 General Idea

The algorithm W-SAT has two major steps. The first step is a “data reduction” step and the second step solves the reduced formula by enumeration + dynamic programming.

3.1.1 STEP 1: Data Reduction

We use a data reduction rule similar to Buss’s reduction for vertex cover [22,42]: If a variable $x$ appears in a set of $m \geq k$ clauses of the form

$$
\begin{align*}
\bar{x} \lor y_{11} \lor \cdots \lor y_{1(d-1)} \\
\bar{x} \lor y_{21} \lor \cdots \lor y_{2(d-1)} \\
\vdots \\
\bar{x} \lor y_{m1} \lor \cdots \lor y_{m(d-1)}
\end{align*}
$$

such that the set of monotone clauses

$$
\begin{align*}
y_{11} \lor \cdots \lor y_{1(d-1)} \\
y_{21} \lor \cdots \lor y_{2(d-1)} \\
\vdots \\
y_{m1} \lor \cdots \lor y_{m(d-1)}
\end{align*}
$$

does not have a satisfying assignment of weight at most $k - 1$, then set $x$ to FALSE. The correctness of the rule is obvious.
For WEIGHTED 2-SAT, this rule can be implemented in \( O(n^2) \) time — just count the number of the clauses of the form \( x \lor y \). For WEIGHTED \( d \)-SAT with \( d > 2 \), we will show later that for random instances of \( F_{k,d}^n \), an \( O(d^k n) \)-time algorithm correctly implements the reduction \text{whp}.

Unlike the case of vertex cover, the above reduction rule won’t result in a reduced formula of fixed size, which is of course expected.

### 3.1.2 STEP 2: Connected Components and Dynamic Programming

We make use of the observation that the reduced formula, while still not fixed in size, breaks up into “connected components” of size at most \( \log n \). Section 4 is devoted to proving this observation.

Let \( \{F_i, 1 \leq i \leq m\} \) be the collection of connected components in the reduced formula. For each connected component \( F_i \), we use brute-force to find the set of integers \( L_i \) such that for each \( k' \in L_i \), there is a weight-\( k' \) assignment to the variables in \( F_i \) that satisfies \( F_i \).

Finally, an \( O(k^2 n) \)-time dynamic programming algorithm can be designed to find a collection of at most \( k \) positive integers \( \{k_{ij}, 1 \leq j \leq k\} \) such that

\[
\begin{align*}
  k_{ij} &\in L_{ij}, \text{ and} \\
  k_{i1} + k_{i2} + \cdots + k_{ik} &= k.
\end{align*}
\]

### 3.1.3 A Further Remark

A typical criticism to the above two steps is that they are more about the structure of the random instance distribution than the algorithms. We argue that all algorithms, especially the efficient ones, are designed by exploiting special problem structures in one way or another such as those due to optimal subproblem structures, or restricted graph classes, or restricted parameter size. The algorithm and its analysis is exactly an demonstration of the kind of structures that may appear in a typical random instance of an intractable problem and can be exploited by reductions that lead to efficient algorithms for other tractable problems.

As a matter of fact, similar structures have been exploited in many studies, most notably in N. Alon and N. Kahale’s seminal work on algorithms for random 3-colorable graphs [5] and its follow-ups [15,23,35,16]. In the author’s previous work [26], similar connected-component structure also arises in a different setting.
3.2 Definitions: k-frozen variables

We give the formal definitions of the concepts that will be used in the description and the analysis of the algorithm.

**Definition 3.1** Let \((F, k)\) be an instance of WEIGHTED \(d\)-SAT where \(F\) is a \(d\)-CNF formula and \(k\) is the parameter. Consider a variable \(x\) and a collection of subsets of variables \(Y = \{Y_i, 1 \leq i \leq m\}\) where

\[ Y_i = \{y_{ij}, 1 \leq j \leq d - 1\} \]

is a subset of \(X \setminus \{x\}\). We say that the collection \(Y\) **freeze** \(x\) if the following two conditions are satisfied:

1. for each \(1 \leq i \leq k\), the clause \(x \lor y_{i1} \lor \cdots \lor y_{i(d-1)}\) is in the formula \(F\).
2. The set of clauses \(\{y_{i1} \lor \cdots \lor y_{i(d-1)}, 1 \leq i \leq m\}\) has no satisfying assignment of weight at most \(k - 1\).

The variable \(x\) is said to be **k-frozen** with respect to a subset \(V\) of variables if it is frozen by a collection of subsets of variables \(\{Y_i, 1 \leq i \leq m\}\) such that \(Y_i \subset V, \forall 1 \leq i \leq m\).

A variable that is k-frozen with respect to the set of all variables is simply called a k-frozen variable.

It is obvious that a k-frozen variable cannot be set to TRUE in any satisfying assignment.

An important observation in the study of random instances of NP-complete problems is the existence of frozen variables (also known as backbone variables) [19,13,41]. Even though detecting backbone variables is usually a task that is no easier than that of solving the original satisfiability problem, information on backbone variables can be utilized to guide the variable selection heuristics in search algorithms [48]. The concept of a frozen variable (with respect to a partial assignment) and the concept of a core set of frozen variables have been used to study the geometric structure of the solution space at phase transitions [3]. Similar concepts also play an important role in the study of random CNF formulas with a planted solution [23,35,16].

Our notion of \(k\)-frozen variables can be viewed as a fixed-parameter generalization of a weak version of the concept of backbone variables. A \(k\)-frozen variable has a forced value in all satisfying assignment, but a variable that has a forced value in all satisfying assignment is not necessarily \(k\)-frozen. Consequently, \(k\)-frozen variables are easier to detect, especially for 2-CNF formulas in which case to see if a variable \(x\) is \(k\)-frozen, one only needs to count the
number of clauses of the form \( \pi \lor y \). For the case of \( d \)-CNF formulas with \( d > 2 \), the condition for a \( k \)-frozen variable is significantly stronger and an \( O(d^k n^d) \)-time algorithm has to be used to test if a variable is \( k \)-frozen.

The following is another concept that is necessary in the description of the algorithm:

**Definition 3.2** Let \( \mathcal{F} \) be a CNF formula. We use \( L_{\mathcal{F}} \) to denote the set of integers between 0 and \( k \) such that for each \( k' \in L_{\mathcal{F}} \), there is a satisfying assignment of weight-\( k' \) for \( \mathcal{F} \).

### 3.3 The Algorithm W-SAT and its Time Complexity

The algorithm is described in Algorithm 1. We will explain the purpose of the subroutine REDUCE() in the next subsection.

**Lemma 3.1** There is an \( O(d^k n^d) \) algorithm that checks if a variable \( x \) is \( k \)-frozen.

**Proof.** Consider the set of all clauses in which \( x \) is the only negated literal: \( \{ \pi \lor y_{i1} \lor \cdots \lor y_{i(d-1)}, 1 \leq i \leq m \} \). According to our definition, \( x \) is \( k \)-frozen if and only if the collection of subsets \( \{ \{ y_{i1}, \cdots, y_{i(d-1)} \}, 1 \leq i \leq m \} \) has no hitting set\(^2\) of size at most \( k-1 \), which can be solved by the bounded search tree method in time \( O(d^k m) \) that branches on the subset. \( \blacksquare \)

We remark that for random instances from \( \mathcal{F}_{k,d}^{n,p} \) with \( p = O(c \log \frac{n}{d^{1-\epsilon}}) \), the above lemma is not really necessary since it can be shown that \( \text{whp} \) all variable \( x \) appear in \( O(\log n) \) clauses. Consequently, a brute-force search runs in \( O(n^{cd}) \) time. But for \( p = \Omega(\frac{1}{n^{d-1}}), \epsilon > 0, \) to avoid the dependency of the running time on \( c \), the above lemma is necessary in order for the algorithm to run in fixed-parameter time. The above is also necessary in order to generalize the algorithm to instances of MINI-WEIGHTED-SAT where the question “does an instance has a satisfying assignment of weight \( k \log n \)”.

We now show the algorithm W-SAT runs in fixed-parameter time and is correct whenever it returns a satisfying assignment or reports “UNSAT”.

**Proposition 3.1** The algorithm W-SAT is correct when it returns a satisfying assignment or reports “UNSAT”. The running time of W-SAT is in \( O(d^k n^d) \) for any \( p(n) \leq 1 \).

\(^2\) A hitting set \( H \) of a collection of subsets \( \{ S_1, \cdots, S_n \} \) in a universe \( U \) is a subset \( H \subset U \) such that \( H \) contains at least one element from each subset \( S_i \).
**Algorithm 1 W-SAT**

**Input:** A random instance \((\mathcal{F}_{k,d}^n, k)\) of WEIGHTED \(d\)-SAT

**Output:** A satisfying assignment of weight \(k\), or UNSAT, or FAILURE

1: for variable \(x\) do
2: if \(x\) is \(k\)-frozen, then set \(x\) to FALSE.
3: Let \(\mathcal{F}' = \text{REDUCE}(\mathcal{F}, U)\) be the reduced formula.
4: Find the connected components \(\{\mathcal{F}_1, \cdots, \mathcal{F}_m\}\) of \(\mathcal{F}'\).
5: If there is a connected component of size larger than \(\log n\), return “FAIL-URE”.
6: Otherwise, for each connected component \(\mathcal{F}_i\), use brute force to find \(L_{\mathcal{F}_i}\).
7: Find a set of at most \(k\) indices \(\{i_j, 1 \leq j \leq k\}\) and a set of integers \(\{k_{i_j}, 1 \leq j \leq k\}\) such that \(k_{i_j} \in L_{\mathcal{F}_{i_j}}\) and
\[
\sum_{j=1}^{k} k_{i_j} = k.
\]

Return “UNSAT” if there is no such index set.
8: For each \(\mathcal{F}_{i_j}\), use brute-force to find a weight-\(k_{i_j}\) assignment to the variables in \(\mathcal{F}_{i_j}\) that satisfies \(\mathcal{F}_{i_j}\).
9: Combine the assignments found in the above to form a weight-\(k\) satisfying assignment to the formula \(\mathcal{F}\).

**Proof.** Since \(k\)-frozen variables are forced to take the truth value FALSE and since the subroutine \text{REDUCE}() never assigns TRUE to a variable (due to the fact that there is no monotone clause in the formula), a formula \(\mathcal{F}\) has a weight-\(k\) satisfying assignment if and only if the reduced formula \(\mathcal{F}'\) has one. Satisfying assignments to the connected components of \(\mathcal{F}'\) can be combined together without falsifying any clauses. So, as long as the sum of the weight of the assignments to the connected components is equal to \(k\), W-SAT finds a weight-\(k\) satisfying assignment in Line 6 through Line 8.

Due to Lemma 3.1 it takes \(O(d^k n^2)\)-time to find all the \(k\)-frozen variables in Step 1. Steps 3 - 5 take \(O(n^d)\)-time and do not depend on \(k\). Step 6 can be finished in \(\Omega(n^2)\)-time since we have to enumerate all the elements of \(L_{\mathcal{F}_i}\) and since \(|L_{\mathcal{F}_i}| \leq \log n\). Step 8 repeats part of the work done Step 6.

We now show that Step 7 can be done in \(O(k^2 n)\) time by dynamic programming. Consider an integer \(k\) and a collection \(\{L_i, 1 \leq i \leq m\}\) where each \(L_i\) is a subset of integers in \(\{0, 1, \cdots, k\}\). We say that an integer \(a\) is achievable by \(\{L_i, 1 \leq i \leq m\}\) if there is a set of indices \(I_a = \{i_j, 1 \leq j \leq l\}\) satisfying the following condition: for each \(i_j\), there is a \(k_{i_j} \in L_{i_j}\) such that
\[
\sum_{j=1}^{l} k_{i_j} = k.
\]
We call any such an index set $I_a$ a **representative set** of $a$.

The purpose of Line 6 is to check if the integer $k$ is **achievable**, and if YES, return a representative set of $k$.

**Lemma 3.2** Given a collection $\{L_i, 1 \leq i \leq m\}$ and an integer $k$ where each $L_i$ is a subset of integers in $\{0,1,\cdots,k\}$, there is a dynamic programming algorithm that finds a representative set of $k$ if $k$ is achievable, or reports that $k$ is not achievable. It runs in time $O(k^2m)$.

**Proof.** Let $A(t) = \{(a,I_a) : 0 \leq a \leq k\}$ be the set of pairs $(a,I_a)$ where $0 \leq a \leq k$ is an integer achievable by $\{L_i, 1 \leq i \leq t\}$ and $I_a$ is a representative set of $a$.

Let $A(0) = \emptyset$. We see that $A(t+1)$ is the union of $A(t)$ and the set of pairs of the form $((a+b),I_a)$ where

$$
\begin{cases}
(a,I_a) \in A(t), \\
b \in L_{t+1} \text{ such that } b \leq k - a, \text{ and} \\
I_a = I_a \cup \{t\}.
\end{cases}
$$

A typical application of dynamic programming computes $A(0), A(1), \cdots, A(m)$. The value $k$ is achievable by $\{L_i, 1 \leq i \leq m\}$ if and only if there is a pair $(k,I_k)$ in $A(m)$. Since the size of $A(t)$ is at most $k$, the above algorithm runs in $O(k^2m)$ time. ■

For the case of Line 6, there are at most $n$ subsets of integers, each of size at most $k$. The running time of Line 6 is $O(k^2n)$. The proposition follows. ■

### 3.4 The subroutine REDUCE($F,U$)

The purpose of the subroutine REDUCE($F,U$) is to simplify the formula $F$ after the variables in $U$ have been assigned FALSE. It works in the same way as the unit-propagation-based inference in the well-known DPLL procedure for the standard satisfiability search: It removes any clause that is satisfied by the assignment to the variables in $U$, deletes all the occurrences of a literal that has become FALSE due to the assignment, and assigns a proper value to the variables whose value is forced due to the literal-deletion. The procedure terminates when there is no variable whose value is forced.

Due to the nature of WEIGHTED $d$-SAT and our random model, we note that REDUCE() never assigns TRUE to a variable. This is because it begins with a set of variables $U$ that have been assigned FALSE, and the formula $F$
contains no monotone clause. As a consequence, REDUCE() will never create a contradiction either.

It is possible that REDUCE() returns an empty formula \( F' \), signifying that all the clauses have been satisfied during the process. However, since we are searching for an satisfying assignment of weight \( k \), we are not done yet in this case. In the description of Algorithm 1, we have omitted this simple special case. The following lemma describes how this situation can be handled:

**Lemma 3.3** If \( F' = REDUCE(F, U) \) is empty, then \( F \) has a weight-\( k \) satisfying assignment if and only if at least \( k \) variables have not been assigned after REDUCE returns.

**Proof.** \( F' \) is obtained by assigning all the variables in \( U \) FALSE and simplifying the formula. The lemma follows from the fact that REDUCE() only sets variables to FALSE and the fact that if \( F' \) is empty, then any variable not assigned yet can be assigned an arbitrary truth value. \( \blacksquare \)

### 4 The Algorithm W-SAT Succeeds With High Probability

In this section, we prove that the algorithm W-SAT succeeds \textit{whp} for random instances of \( F_{n,p}^{k,d} \). Due to Proposition 3.1, we only need to show that W-SAT reports “FAILURE” \textit{whp}. Recall that W-SAT fails only when the reduced formula \( F' \) obtained in Line 2 has a connected component of size at least \( \log n \).

First, we deal with the case of \( p(n) > \frac{c \log n}{n^{d-1}} \) with \( c = 2(2^d - 1)(d-1)! \). We show in the following Lemma that \textit{whp} all variables are \( k \)-frozen and have to be set to FALSE. And consequently the reduced formula is empty and the algorithm returns the correctly answer “NO”.

**Lemma 4.1** For random instances from \( F_{k,d}^{n,p} \) where \( p(n) > \frac{c \log n}{n^{d-1}} \) with \( c = 2(2^d - 1)(d-1)! \), \textit{whp} all variables are \( k \)-frozen.

**Proof.** See Appendix 10. \( \blacksquare \)

We now focus on the case \( p(n) = \frac{c \log n}{n^{d-1}} \) with \( c < 2(2^d - 1)(d-1)! \).

**Proposition 4.1** Let \( F = F_{k,d}^{n,p} \) be the input random CNF formula to W-SAT. With high probability, the residual graph of the induced formula \( F_V \) on \( V \) decomposes into a collection of connected components of size at most \( \log n \), where \( V \) is the set of variables that are not \( k \)-frozen.

**Proof.** Let \( X = \{x_1, \ldots, x_n\} \) be the set of Boolean variables, and let \( U \) be
the set of \textit{k-frozen} variables so that \( V = X \setminus U \).

Since \( p = \frac{c \log n}{n^d} \) with \( c > 0 \), there will be many \( k \)-frozen variables so that the size of \( U \) is large. If \( U \) were a randomly-selected subset of variables, the proposition is easy to prove. The difficulty is that in our case \( U \) is not randomly-selected, and consequently \( \mathcal{F}_V \) cannot be assumed to be distributed in the same manner as the input formula \( \mathcal{F} \).

To get around this difficulty, we instead directly upper bound the probability \( P^* \) that the residual graph of \( \mathcal{F}^{n,p}_{k,d} \) contains, as its subgraph, a tree \( T \) over a given set \( V_T \) of \( \log n \) variables such that every variable \( x \in V_T \) is not \( k \)-frozen.

Since the variables in \( \mathcal{F}_V \) are not \( k \)-frozen, an upper bound on the probability \( P^* \) is also an upper bound on the probability that the residual graph of \( \mathcal{F}_V \) contains, as its subgraph, a tree of the size \( \log n \). We then use this upper bound and Markov’s inequality to show that the probability for the residual graph of \( \mathcal{F}_V \) to have a connected component of size at least \( \log n \) tends to zero.

Let \( T \) be a fixed tree over a subset \( V_T \) of \( \log n \) variables. A further complication in estimating \( P^* \) is that the event that \( \mathcal{F}^{n,p}_{k,d} \) induces \( T \) and the event that no variable in \( T \) is \( k \)-frozen are not independent of each other. To decouple the dependency, we consider the following two events

1. \( \mathcal{A} \): the event that the residual graph of \( \mathcal{F}^{n,p}_{k,d} \) contains the tree \( T \) as its subgraph; and
2. \( \mathcal{B} \): the event that in \( \mathcal{F}^{n,p}_{k,d} \) none of the variables in \( V_T \) involves a set of \( k \) clauses of the following form:

\[ \{ \pi \lor y_{i1} \lor \cdots \lor y_{i(d-1)}, 1 \leq i \leq m \} \]

where \( y_{ij} \)'s are distinct variables and \( y_{ij} \in X \setminus V_T, \forall i, j \).

By definition, a variable \( x \) is not \( k \)-frozen implies that \( x \) is not \( k \)-frozen with respect to a subset \( X \setminus V_T \) of variables, which in turns implies that \( x \) is not involved in a set of \( k \) clauses of the form

\[ \{ \pi \lor y_{i1} \lor \cdots \lor y_{i(d-1)}, 1 \leq i \leq m \} \]

where \( y_{ij} \)'s are distinct variables and \( y_{ij} \in X \setminus V_T, \forall i, j \). It follows that

\[ P^* \leq \mathbb{P} \{ \mathcal{A} \cap \mathcal{B} \}. \quad (4.1) \]

We claim that

\textbf{Lemma 4.2} \textit{The two events} \( \mathcal{A} \) \textit{and} \( \mathcal{B} \) \textit{are independent, i.e.,}

\[ \mathbb{P} \{ \mathcal{A} | \mathcal{B} \} = \mathbb{P} \{ \mathcal{B} \} \quad (4.2) \]
Proof. Note that the event $A$ depends only on those $d$-clauses that contain at least two variables in $V_T$, and that the event $B$ depends only on those $d$-clauses that contain exactly one variable from $V_T$. Due to the definition of the random model $\mathcal{F}_{k,d}^{n,p}$, the appearance of a clause defined over a $d$-tuple of variables is independent from the appearance of the other clauses. The Lemma follows. ■

Based on Equation (4.1) and Lemma 4.2, we only need to estimate $\mathbb{P}\{A\}$ and $\mathbb{P}\{B\}$ separately. To proceed, we need the following Chernoff bound on the tail probability of a binomial random variable.

**Lemma 4.3** Let $I$ be a binomial random variable with expectation $\mu$. We have

$$\mathbb{P}\{|I - \mu| > t\} \leq 2e^{-\frac{t^2}{3\mu}}.$$ 

The following lemma bounds the probability that a variable is not $k$-frozen.

**Lemma 4.4** Let $x$ be a variable and $W \subset X$ such that $x \in W$ and $|W| > n - \log n$. Then, we have

$$\mathbb{P}\{x \text{ is not } k\text{-frozen with respect to } W\} \leq O(1) \max\left(\frac{1}{n^\delta}, \frac{\log^2 n}{n}\right)$$

where $0 < \delta < \frac{\alpha}{3(2^d - 1)(d - 1)}$.

Proof. Let $N_x$ be the number of clauses of the form $x \lor y_1 \lor \cdots \lor y_{d-1}$ with $\{y_1, \cdots, y_{d-1}\} \subset X \setminus V_T$. Due to the definition of $\mathcal{F}_{k,d}^{n,p}$, the random variable $N_x$ follows the binomial distribution $Bin(\overline{p}, m)$ where $\overline{p} = \frac{1}{2^d - 1} \frac{c\log n}{n^{d-1}}$ and $m = \binom{n - \log n}{d - 1}$.

Write $\alpha = \frac{\epsilon}{(2^d - 1)(d - 1)}$. By Lemma 4.3, we have

$$\mathbb{P}\{N_x < k\} \leq 2e^{-\frac{(\overline{p}m - k)^2}{3\overline{p}m}} \leq O(k)e^{-\frac{2}{3} \log n}$$

$$\in O(n^{-\delta}) \quad (\text{where } 0 < \delta < \frac{\alpha}{3}).$$ (4.3)

Let $D$ be the event that in the random formula $F$, there are two clauses

$$\begin{align*}
\exists x &\quad \{ \overline{x} \lor y_{11} \lor \cdots \lor y_{1(d-1)}, \text{ and} \\
&\quad \overline{x} \lor y_{12} \lor \cdots \lor y_{2(d-1)} \}
\end{align*}$$
such that \( \{y_{11}, \cdots, y_{1(d-1)}\} \cap \{y_{12}, \cdots, y_{2(d-1)}\} \neq \emptyset \). The total number of such possible pairs of clauses is at most

\[
(d - 1) \binom{n - \log n}{d - 1} \binom{n - \log n}{d - 2}.
\]

The probability for a specific pair to be in the random formula is

\[
\left( \frac{1}{2^d - 1} \frac{c \log n}{n^{d-1}} \right)^2.
\]

By Markov’s inequality, we have

\[
P \{D\} \in O\left( \frac{\log^2 n}{n} \right).
\]

Since the probability that the variable \( x \) is not \( k \)-frozen is at most

\[
P \{\{N_x < k\} \cup D\},
\]

the lemma follows.

\[\blacksquare\]

From Lemma 4.4, we have

**Lemma 4.5** For sufficiently large \( n \),

\[
P \{B\} \leq O(1) (\log n) \log n n^{-\log n}. \tag{4.4}
\]

for some \( 0 < \delta < \min\left(\frac{c}{5(2^d - 1)(d-1)}\right), 1 \).

**Proof.** Let \( E_x \) be the event that a variable \( x \in V_T \) is not \( k \)-frozen with respect to \( X \setminus V_T \). Since \( |V_T| = \log n \), the bound obtained in Lemma 4.4 applies to \( W = X \setminus V_T \). Since for any \( x \in V_T \), the event \( E_x \) only depends on the existence of clauses of the form

\[
\tau \lor y_{i1} \lor \cdots \lor y_{i(d-1)}
\]

with \( \{y_{i1}, \cdots, y_{i(d-1)}\} \subset X \setminus V_T \), we see that the collection of the events \( \{E_x, x \in V_T\} \) are mutually independent. The Lemma follows from Lemma 4.4.

\[\blacksquare\]

Next, we have the following upper bound on the probability \( P \{A\} \). Its proof is based on a counting argument that slightly generalizes that used in [23,35]. See Appendix 11.

**Lemma 4.6**

\[
P \{A\} \leq O(1) (\log n)^{\log n} n^{-\log n}.
\]
Continuing the proof of Proposition 4.1, we combine the results of Lemma 4.5 and Lemma 4.6 to get
\[ P\{A \cap B\} \leq O(1)(\log n)^2 \log n n^{-\log n} \left(n^{-\delta}\right)^{\log n}. \]

Since the total number of trees of size \( \log n \) is at most \( n^{\log n (\log n)^{\log n - 2}} \), the probability that the residual graph of \( \mathcal{F}_V \) contains a tree of size \( \log n \) is
\[
\frac{n^{\log n (\log n)^{\log n - 2}} P\{A \cap B\}}{O(1)(\log n)^3 \log n \left(n^{-\delta}\right)^{\log n}} \leq \frac{1}{O(1)(\log n)^3 \log n \left(n^{-\delta}\right)^{\log n}}
\]
(4.5)

Proposition 4.1 follows.

Proof. [Proof of Theorem 1] To use Proposition 4.1 to prove that Algorithm W-SAT succeeds whp, we note that the reduced formula \( \mathcal{F}' \) in Line 2 of the algorithm W-SAT is sparser than the induced formula \( \mathcal{F}_V \). In fact, it is easy to see that \( \mathcal{F}' \) is an induced sub-formula of \( \mathcal{F}_V \) over the set of variables that have not been assigned by the subroutine REDUCE(). Therefore by Proposition 4.1, with high probability \( \mathcal{F}' \) decomposes into a collection of connected components, each of size at most \( \log n \). It follows that W-SAT succeeds whp.

Combining all the above, we conclude that Algorithm W-SAT is a fixed-parameter algorithm and succeeds whp on random instances of \( \mathcal{F}_{n,p}^{k,d} \). This proves Theorem 1.

5 The Threshold Behavior of the Solution Probability

In this section, we prove Theorem 2 to establish the exact threshold of the solution probability phase transition. Unlike the threshold for random instances of most traditional NP-complete problems for which the exact threshold is still an open question, the exact threshold of the weighted satisfiability problem can be established by the first-moment method and the second-moment method.

Proof. [Proof of Theorem 2] Let \( T \) be the collection of all subsets of \( d \) variables and let \( s \) be an assignment to the variables. We say that a subset \( T = \{x_1, \cdots, x_d\} \in T \) is s-good if either

(1) \( T \) doesn’t contribute a clause to \( \mathcal{F}_{k,d}^{n,p} \), or
(2) the \( d \)-clause in \( \mathcal{F}_{k,d}^{n,p} \) contributed by \( T \) is satisfied by the assignment \( s \).

Let \( S \) be the set of all assignments of weight \( k \). Recall that the weight of an assignment is the number of variables that are set to TRUE in the assignment.
Consider an assignment \( s \in S \) where the \( k \) variables \( Y = \{ y_{i_1}, \ldots, y_{i_k} \} \) are set to TRUE. From the definition of \( F_{n,p}^{a,b} \), the probability for \( T \in T \) to be \( s \)-good is

\[
\mathbb{P}\{ T \text{ is } s\text{-good}\} = \begin{cases} 
1, & \text{if } T \cap Y = \emptyset; \\
1 - p(n) \frac{1}{2^d - 1}, & \text{otherwise}.
\end{cases} 
\tag{5.6}
\]

Let \( T_s \subset T \) be the collection of subsets of \( d \) variables that have a nonempty intersection with \( Y \). We have

\[
|T_s| = \sum_{j=1}^{d} \binom{n-k}{d-j} \binom{k}{j}.
\]

Let \( X \) be the number of assignments in \( S \) that satisfy \( F_{n,p}^{a,b} \). Then, we have

\[
\mathbb{E}[X] = \sum_{s \in S} \prod_{T \in T_s} \mathbb{P}\{ T \text{ is } s\text{-good}\} \\
= \sum_{s \in S} \prod_{T \in T_s} \mathbb{P}\{ T \text{ is } s\text{-good}\} \\
= \binom{n}{k} \left( 1 - p(n) \frac{1}{2^d - 1} \right) \sum_{j=1}^{k} \frac{(n-k)}{(d-j)} \binom{k}{j} \\
\sim n^k e^{-\frac{k(e \log n)}{(2^d - 1)(d-1)!}}. 
\tag{5.7}
\]

where \( \sim \) means “is asymptotically equivalent to”. The upper bound on the threshold \( c^* \) follows from Markov’s inequality and the above asymptotic expression of \( \mathbb{E}[X] \).

To lower bound the threshold \( c^* \), we use Chebyshev’s inequality

\[
\mathbb{P}\{ X = 0 \} \leq \frac{\mathbb{E}[X^2]}{(\mathbb{E}[X])^2} - 1.
\]

We say that two assignments \( s_1, s_2 \in S \) have \( i \) overlaps if exactly \( i \) variables are set to true in both assignments. Let \( D_i \) be the number of pairs of satisfying assignments in \( S \) that have \( i \) overlaps. We can represent \( \mathbb{E}[X^2] \) as

\[
\mathbb{E}[X^2] = \sum_{i=1}^{k} \mathbb{E}[D_i]. 
\tag{5.8}
\]

Let \( \epsilon > 0 \) be any number. We claim that for and \( c = 1 - \epsilon < \frac{1}{2^d - 1} d - 1! \),

\[
\begin{cases} 
\mathbb{E}[D_i] \in o(n^{2k\epsilon-i\epsilon}) \text{ for } i > 1, \text{ and} \\
\lim_{n} D_0 = (\mathbb{E}[X])^2 \tag{5.9}
\end{cases}
\]

\[\]
By definition, we have
\[
\mathbb{E} [D_i] = \sum_{s_1, s_2} \prod_T \mathbb{P} \{T \text{ is both } s_1\text{-good and } s_2\text{-good}\},
\]
where the sum is over all (ordered) pairs of weight-\(k\) assignments with \(i\)-overlaps, and the product is over all subsets \(T\) of \(d\) variables.

Consider two weight-\(k\) assignments \(s_1, s_2 \in S\) that have \(i\) overlaps. W.L.O.G, assume that the set of variables set to true by \(s_1\) is \(\{y_1, \ldots, y_{k-i}, y_{k-i+1}, \ldots, y_k\}\) and the set of variables set to true by \(s_2\) is \(\{y_{k-i+1}, \ldots, y_k, y_{k+1}, \ldots, y_{2k-i}\}\).

Write \(Y_{s_1, s_2} = \{y_1, \ldots, y_k, \ldots, y_{2k-i}\}\).

The probability for a set of \(d\) variables \(T \in \mathcal{T}\) to be both \(s_1\)-good and \(s_2\)-good can be estimated as follows:

1. If \(T \cap Y_{s_1, s_2} = \emptyset\), then \(\mathbb{P} \{T \text{ is } s_1\text{-good and } s_2\text{-good}\} = 1\).
2. If \(T \cap Y_{s_1, s_2} \neq \emptyset\), then \(\mathbb{P} \{T \text{ is } s_1\text{-good and } s_2\text{-good}\} \leq \left(1 - p(n) \frac{1}{2^d - 1}\right)^d\).

To see this, note that in this case \(T\) has a non-empty intersection with either \(\{y_1, \ldots, y_k\}\), or \(\{y_{k-i+1}, \ldots, y_{2k-i}\}\), or both. Therefore, either
\[
\mathbb{P} \{T \text{ is } s_1\text{-good}\} = \left(1 - p(n) \frac{1}{2^d - 1}\right),
\]
\[
\mathbb{P} \{T \text{ is } s_2\text{-good}\} = \left(1 - p(n) \frac{1}{2^d - 1}\right).
\]

Note that the total number of \(T \in \mathcal{T}\) such that \(T \cap Y \neq \emptyset\) is
\[
\sum_{j=1}^{d} \binom{n - (2k - i)}{d - j} \binom{2k - i}{j}.
\]

It follows that for \(i > 1\),
\[
\mathbb{E} [D_i] = \binom{n}{k} \binom{n-k}{k-i} \prod_{T:T \cap Y \neq \emptyset} \left(1 - p(n) \frac{1}{2^d - 1}\right)
\sim n^{2k-i} \left(1 - p(n) \frac{1}{2^d - 1}\right)^d \sum_{j=1}^{d} \binom{n-(2k-i)}{d-j} \binom{2k-i}{j}
\sim n^{2k-i} e^{-\frac{(2k-i):\log n}{(2^d-1)(d-1)!}}
\sim n^{2k-i} e^{-\frac{(2k-i):\log n}{(2^d-1)(d-1)!}}.
\]

where \(~\) means “is asymptotically equivalent to”. For the case of \(i = 0\), it can be shown that \(\lim_{n \to \infty} D_0 = (\mathbb{E} [X])^2\). This proves the claim. The theorem follows from equations (5.7), (5.8), and (5.10).

\[\square\]
Parametric Resolution Complexity of Unsatisfiable Instances of $\mathcal{F}_{k,d}^{n,p}$

Even though the algorithm W-SAT solves both satisfiable and unsatisfiable instances in fixed-parameter time, it is unclear to us how to simulate the dynamic programming phase (Line 6) of W-SAT using a resolution proof.

In this section, we prove Theorems 3 and 4 on the parametric resolution complexity of random unsatisfiable instances of $\mathcal{F}_{k,d}^{n,p}$. We start with Theorem 3 that deals with the case of $d > 2$ by making use of the fact that in certain range of $p$, every variable becomes $k$-frozen and that for a $k$-frozen variable $x$, a parametric resolution derivation of $\pi$ of size $O((d-1)^k k^k)$ can be constructed.

For random WEIGHTED 2-SAT, results on the minimum degree in a random graph can be applied to show that if $p = \frac{3(\log n + c_1 \log \log n)}{n}$ with $c_1 > k - 1$, every variable is $k$-frozen whp. A parametric resolution proof based on the $k$-frozenness of all the variables is immediate.

The proof of Theorem 4 on the resolution complexity of random instances of WEIGHTED 2-SAT exploits the existence of a Hamiltonian-cycle-like system of clauses so that the theorem holds for both WEIGHTED 2-SAT and MINI-WEIGHTED 2-SAT.

6.1 Proof of Theorem 3 on the Resolution Complexity of $\mathcal{F}_{k,d}^{n,p}$, $d > 2$

Proof. We establish Theorem 3 by making use of the fact that for $p = \frac{c \log n}{n^{d-1}}$ with $c > 2(2^d - 1)(d - 1)!$, all the variables are $k$-frozen whp.

Since by Lemma 4.1, all variables are $k$-frozen, one can construct a parametric (tree) resolution proof as follows. For each variable $x$, there is a parametric resolution derivation of size $O(d^k)$ as described in the following lemma:

**Lemma 6.1** If a variable $x$ is $k$-frozen, then there is a parametric resolution derivation of $\pi$ of size $O(d^k)$ exists.

Proof. We show that a parametric resolution derivation of size $O(d^k)$ can be obtained from the set $\mathcal{C}$ of $k$ clauses

\[
\begin{align*}
\mathcal{C} &\equiv \left\{ \pi \lor y_{i1} \lor \cdots \lor y_{i(d-1)} \right. \\
& \quad \cdots \\
& \left. \pi \lor y_{k1} \lor \cdots \lor y_{k(d-1)} \right\}
\end{align*}
\]
that freeze $x$. Recall that in a parametric resolution proof, we assume that the proof system has free access to all anti-monotone clauses of size $k + 1$.

Consider a DPLL-style search tree method that first assigns $x$ TRUE and then, enumerates all the possible assignments to the $k$ variables, one from each set \( \{ y_{ij}, 1 \leq j \leq (d - 1) \} \), $1 \leq i \leq k$, in order to satisfy the $k$ clauses in $C$ with $\tau$ removed. The size of the search tree is $O(d^k)$ and at each leaf node, some anti-monotone clause of size $k + 1$ will be made empty. In the same way a tree resolution proof can be constructed from a DPLL search tree for the CNF satisfiability problem, a tree parametric derivation of $\tau$ can be constructed from the above search tree. ■

Continuing the proof of the theorem, it follows from the above lemma that there is a size $O(d^kn)$ parametric resolution derivation of the set of single-literal clauses \( \{ \overline{x} \}, x \in X \). From the set of clauses \( \{ \overline{x} \}, x \in X \), together with the monotone clause $\lor_{i=1}^{n} x_i$ (which says that at least one variable has to be set to TRUE), a parametric resolution proof of size $O(n)$ can be easily constructed for the empty clause (i.e., the contradiction). ■

6.2 The Resolution Complexity of $F_{k,2}^{n,p}$

**Proof.** [Proof of Theorem 4] We prove Theorem 4 on the resolution complexity of $F_{k,2}^{n,p}$ by exploiting an interesting connection between the parameterized resolution complexity of WEIGHTED 2-SAT and the existence of a Hamiltonian cycle in a properly defined directed graph. The following lemma establishes the connection.

**Lemma 6.2** Consider an instance $(\mathcal{F}, k)$ of WEIGHTED 2-SAT and an arbitrary ordering \( \{ x_1, \cdots, x_n \} \) of the variables. If $\mathcal{F}$ contains the following cycle of “forcing” clauses,

$$\overline{x}_1 \lor x_2, \overline{x}_2 \lor x_3, \cdots, \overline{x}_{n-1} \lor x_n, \overline{x}_n \lor x_1,$$

then there is a parametric resolution proof of size $O(kn)$ for $\mathcal{F}$. Furthermore, the parameterized version of the DPLL algorithm constructs such a resolution proof.

**Proof.** Recall that the parametric resolution proof system has access to all the clauses that contain more than $k$ negated variables or more than $n - k$ positive variables. For each $i \geq 1$, resolving

$$\overline{x}_i \lor x_{i+1}, \cdots, \overline{x}_{i+k} \lor x_{i+k+1}$$

and $\overline{x}_{i+1} \lor \cdots \lor \overline{x}_{i+k+1}$ gives $\overline{x}_i$. These $\overline{x}_i$’s together with $x_1 \lor \cdots \lor x_{n-k+1}$ result in a contradiction. ■
We emphasize that a cycle of forcing clauses, if exists, can be automatically exploited by DPLL-style algorithms. In a random CNF formula generated from $F_{k,2}^{n,p}$, the existence of a cycle of “forcing” clauses can be shown using a result of McDiarmid on the relation between the existence of a Hamiltonian cycle (or a long path) in a random graph and the existence of a directed Hamiltonian cycle (or a directed long path) in a properly-defined “directed random graph”.

**Lemma 6.3** (McDiarmid, [38]) Let $0 < p < 1$ and $D(n, p)$ be a random directed graph obtained by randomly directing every edge in the random graph $G(n, 2p)$. Then,

$$\mathbb{P}\{D(n, p) \text{ has a directed Hamiltonian cycle}\} \geq \mathbb{P}\{G(n, p) \text{ has a Hamiltonian cycle}\}.$$

From Lemma 6.2, it is sufficient to show that there is a directed Hamiltonian cycle in the following directed graph $D_F(n, p')$: Each vertex corresponds to a variable. There is a directed edge from $x_i$ to $x_j$ if and only if the 2-clause $x_i \lor x_j$ is in the random formula $F_{k,2}^{n,p}$.

From the definition of $F_{k,2}^{n,p}$, $D_F(n, p')$ is exactly the random directed graph $D(n, p')$ discussed in Lemma 6.3 with edge probability $p' = \frac{1}{3} \frac{\log n}{n}$. Since $c > 3$, the result follows from Lemma 6.3 and the threshold of the existence of a Hamiltonian cycle in the random graph $G(n, p)$ [12].

7 W-SAT for MINI-WEIGHTED $d$-SAT

We show that a simple modification of the algorithm W-SAT can solve whp random instances of MINI-WEIGHTED $d$-SAT from $F_{k,d}^{n,p}$ when $p = p(n)$ is in a certain range, and thus prove Corollary 1.1.

Recall that for a random instance $(F_{k,d}^{n,p}, k)$ of MINI-WEIGHTED $d$-SAT, we are looking for a satisfying assignment of weight $k \log n$. Thus, Algorithm W-SAT needs to consider and make use of the existence of $k \log n$-frozen variables. To guarantee that W-SAT still succeeds whp, we need to prove a result similar to Proposition 4.1. This amounts to showing that the probability that a variable $x$ is $k \log n$-frozen is small enough. For $p = \frac{c \log n}{n^{d-1}}$ with $c > k2^{d-1}(d-1)!$, this is the case:

**Proposition 7.1** Let $F = F_{k,d}^{n,p}$ be the input random CNF formula to W-SAT customized to MINI-WEIGHTED $d$-SAT (i.e., based on $k \log n$-frozen variables). Let $p = \frac{c \log n}{n^{d-1}}$ be such that $c > k(2^d-1)(d-1)!$. With high probability,
the residual graph of the induced-formula $F$ decomposes into a collection of connected components of size at most $\log n$.

Proof. The proof is almost the same as the proof of Proposition 4.1 except that we need to establish an upper bound on the probability that a variable is not $k \log n$-frozen. For $c > k(2^d - 1)(d - 1)!$, Lemma 4.3 on the tail probability of a binomial random variable still applies and the arguments made in the second half of the proof of Lemma 4.4 and in the proof of Lemma 4.5 are still valid. The only difference is the accuracy of the upper bound. In this case, we have $\Pr \{B\} \leq O(1)\frac{1}{n^{\log \pi}}$ where where $0 < \delta < \min\left(\frac{(k-c)^2}{2^d(d-1)} \left(\frac{1}{(d-1)!}\right), 1\right)$. This is sufficient for the result to hold.

8 A More General Model $F_{n,p}^{n,p}(d')$

In this section, we discuss how to generalize the model $F_{n,p}^{n,p}$. Let $1 \leq d' \leq d$ and consider the model $F_{n,p}^{n,p}(d')$ defined as follows: instead of from the set of non-monotone clauses, we select uniformly at random from the set of clauses over $\{x_i, \ldots, x_{i,d}\}$ that contain at least $d'$ negated literals. Note that $F_{n,p}^{n,p}$ is just $F_{n,d}(1)$.

First, we have the following result on the exact threshold of the phase transition:

**Theorem 5** Consider a random instance $(F_{k,d}^{n,p}(d'), k)$ of WEIGHTED d-SAT. Let $p = \frac{c \log n}{n^{d-d'}}$ with $c > 0$ being a constant and let $c^* = a_d(d-d')!$ with $a_d$ being the number of $d$-clauses over a fixed set of $d$ variables that contain at least $d'$ negated literals. We have

$$\lim_n \Pr \{F_{k,d}^{n,p}(d') \text{ is satisfiable }\} = \begin{cases} 1, & \text{if } c < c^*; \\ 0, & \text{if } c > c^* \end{cases}$$

Proof. Same as that of Theorem 2. 

We have the following result on the parametric resolution complexity of unsatisfiable random instances of $F_{k,d}^{n,p}(d')$.

**Theorem 6** Let $p(n) = \frac{c \log n}{n^{d-d'}}$ with $c > 2(d-d')a_d(d-1)!$. With high probability, a random instance $(F_{k,d}^{n,p}, k)$ of WEIGHTED d-SAT has a resolution proof of size $O((d-d')^k n^{d'+1)}$, and can be constructed in $O((d-d')^k n^{d'+1})$ time.

28
Proof. Extending the notion of a $k$-frozen variable, we call a set of $d$ variables $S = \{x_{i_1}, \ldots, x_{i_d}\}$ a \textit{frozen tuple} if the following $k - d'$ clauses are in the random CNF formula:

\[
\begin{align*}
\{ \begin{array}{l}
\neg x_{i_1} \lor \cdots \lor \neg x_{i_d} \lor y_{11} \lor \cdots \lor y_{1(k-d')1} \\
\vdots \\
\neg x_{i_1} \lor \cdots \lor \neg x_{i_d} \lor y_{k-d'}(k-d') \\
\end{array} \}
\end{align*}
\]

where $y_{ij}$'s are distinct variables. It is clear that no satisfying weight-$k$ assignment is allowed to assign the $d'$ variables in a frozen tuple to TRUE. Similar to the proof of Theorem 3 in Section 6, we use the extended Jason inequality to show that \textit{whp} all the $d'$-tuples of variables are frozen tuples.

For a given set of $k$ variables $S$, let

\[
\mathcal{P} = \{(Y_i, 1 \leq i \leq k - d') : Y_i \subset X \setminus S, Y_i \cap Y_j = \emptyset, |Y_i| = d - d'\}
\]

be the collection of $(k - d')$ pairwise disjoint subsets of $(d - d')$ variables. For a given $Y \in \mathcal{P}$, let $B_Y$ be the event that $Y$ freezes $S$ in the random formula $F_{n,p}^{k,d}(d')$. We have

\[
\mathbb{P}\{B_Y\} = \left(\frac{c \log n}{n^{d-d'} a_d}\right)^{k-d'}.
\]

Since

\[
|\mathcal{P}| = \binom{n-d'}{d'-d'} \binom{n-(d-d')}{d-d'} \cdots \binom{n-(k-d'-1)(d-d')}{d-d'},
\]

we see that

\[
\mu = \sum_{Y \in \mathcal{P}} \mathbb{P}\{B_Y\}
\]

is asymptotically $(\frac{c \log n}{a_d(d-d')})^{k-d'}$. Similar to the proof of Theorem 3 in Section 6, by the extended Jason inequality, we get

\[
\mathbb{P}\{S \text{ is not } k\text{-frozen}\} = \mathbb{P}\left\{\bigcap_{Y} \overline{B_Y}\right\} \leq e^{-\frac{e}{2} \frac{c \log n}{a_d(d-d')} \log n}.
\]

Consequently based on Markov's inequality, we have that for $c > 2(d - d') a_d(d - d')!$, all the tuples of $d'$ variables are frozen tuple \textit{whp}. The rest of the proof is identical to that of Theorem 3 in Section 6 with the role of a single variable $x$ therein replaced by a set $S$ of $k$ variables. \qed
For satisfiable region, the best we currently have is that for clause probability $p(n) = o\left(\frac{\log n}{n^{d-1}}\right)$, random instances can be shown to be satisfiable due to an observation similar to Lemma 2.1. In contrast, the phase transition occurs at $p(n) = \Theta\left(\frac{\log n}{n^{d-d'}}\right)$. One possible approach to extending the idea of the algorithm W-SAT is to make use the notion of a $k$-frozen tuple in the proof of Theorem 6 and to reduce the random formula on those frozen tuples. We can show that there are many such $k$-frozen tuples in a random formula, but are unable to show that the extended algorithm succeeds with high probability.

We leave it as a future work and challenge for researchers in AI and algorithms to make progress toward closing this huge gap.

9 Conclusions

Data reduction is a powerful pruning technique that has been widely used in many areas of AI and algorithmics. Understanding the effectiveness and applicability of data reduction as a technique for the design of heuristics for intractable problems has been one of the main motivation behind the interest in the study of the phase transitions of randomly-generated NP-complete problems.

The current paper takes the first step to extend this line of research to intractable parameterized problem. We proposed a non-trivial random model for a generic intractable parameterized problem, the weighted $d$-CNF satisfiability problem, and provided an almost complete characterization of the probabilistic behavior of the model. To the best knowledge of the author, our algorithm and its analysis presents the first sound theoretical evidence on the effectiveness of simple reduction rules when applied to intractable parameterized problems.

We believe that the results and insights obtained in this study have potential applications in characterizing the structure of the solution space of standard propositional satisfiability problem [3] and in improving the effectiveness and the efficiency of local-search-based satisfiability algorithms. It is also interesting to use the models and ideas developed in this work to study other parametric problems, especially those related to backdoor detection problems and PSPACE-complete AI planning problems.

With regard to future work on random instances of weighted CNF satisfiability as discussed in this paper, we list in the following several interesting (and challenging) tasks:

(1) In this paper, we only have a limited success in establishing lower bounds
on the parametric resolution complexity of random instances of $F_{k,d}^{n,p}$ for MINI-WEIGHTED \(d\)-SAT. Establishing lower bounds for the size of general parametric (tree) resolution proof systems seems to require new techniques other than those that have been shown to be powerful in the study of (non-parametric) resolution complexity of random CNF formulas [9].

(2) As has been mentioned in Section 3, for \(p = \frac{c\log n}{n^{d-1}}\) with \(c\) small enough, there will be sufficient number of “isolated” variables and by simply setting \(k\) of these variables to TRUE and the remaining variables to FALSE, we get a weight-\(k\) satisfying assignment. It is interesting to see what will happen if these isolated variables are removed.

(3) More importantly, we see that a thorough understanding of the more general model $F_{k,d}^{n,p}(d')$ is a challenging task, requiring new ideas, analytical techniques, and significant empirical studies. We leave it as a challenge for researchers in the fields of AI and algorithms to study the behavior of this general model.

References


Note that Lemmas 4.4 and 4.5 are not strong enough to guarantee this because of the term $\log^2 n / n$ of the bound. Instead, we use the extended Janson inequality (see, e.g., Theorem 8.1.2 in [6]). However, approach doesn’t work for MINIWEIGHTED $d$-SAT instances.

**Lemma 10.1 (The Extended Janson Inequality, Theorem 8.1.2 [6])** Let $\{B_i\}$ be a collection of events, $\mu = \sum_i P\{B_i\}$, and $\Delta = \sum_{i \sim j} P\{B_i \cap B_j\}$ where $i \sim j$ means that “$i \neq j$ and $B_i \cap B_j \neq \emptyset$”. Assume that $\Delta \geq \mu$. We have

$$P\left(\bigcap_i B_i^c\right) \leq e^{-\mu^2/2\Delta}.$$  

To apply the extended Janson inequality to our situation, let $\mathcal{P}$ be the family of the collections of $k$ pairwise-disjoint subsets of variables in $X \setminus \{x\}$, i.e.,

$$\mathcal{P} = \{ (Y_i, 1 \leq i \leq k) : Y_i \subset X \setminus \{x\}, Y_i \cap Y_j = \emptyset, |Y_i| = d - 1 \}.$$
We also require that the variables in each collection \((Y_i, 1 \leq i \leq k)\) are distinct.

For a given \(Y = (Y_i, 1 \leq i \leq k) \in \mathcal{P}\), let \(B_Y\) be the event that the clauses \(\pi \lor Y_i, 1 \leq i \leq k\), are in the random formula \(\mathcal{F}_{k,d}^{n,p}\). Note that \(B_Y\) implies that the variable \(x\) is \(k\)-frozen. We have

\[
P\{B_Y\} = \left( \frac{c \log n}{n^{d-1}} \frac{1}{2^d - 1} \right)^k.
\]

Since

\[
|\mathcal{P}| = \binom{n}{d-1} \binom{n-(d-1)}{d-1} \cdots \binom{n-(k-1)(d-1)}{d-1},
\]

we have that

\[
\mu = \sum_{Y \in \mathcal{P}} P\{B_Y\}
\]

is asymptotically \(\left( \frac{c \log n}{(2^d - 1)(d-1)!} \right)^k\).

The \(\Delta\) in Lemma 10.1 is

\[
\Delta = \sum_{Y_1 \cap Y_2 \neq \emptyset} P\{B_{Y_1} \cap B_{Y_2}\}
\]

where the sum is over all the pairs \((Y_1, Y_2) \in \mathcal{P} \times \mathcal{P}\) such that \(Y_1 \cap Y_2 \neq \emptyset\).

For any pair \((Y_1, Y_2)\) such that \(|Y_1 \cap Y_2| = i\), we have

\[
P\{B_{Y_1} \cap B_{Y_2}\} = \left( \frac{c \log n}{n^{d-1}} \frac{1}{2^d - 1} \right)^{2k-i}.
\]

Since the total number of pairs \((Y_1, Y_2)\) with \(i\) overlaps is

\[
\binom{n}{d-1} \cdots \binom{n-(2k-i-1)}{d-1},
\]

we see that for \(n\) sufficiently large,

\[
\Delta = \sum_{i=1}^{k-1} \sum_{|Y_1 \cap Y_2| = i} P\{B_{Y_1} \cap B_{Y_2}\} = O \left( \left( \frac{c \log n}{(2^d - 1)(d-1)!} \right)^{2k-1} k \right).
\]

It follows from the extended Janson’s inequality that

\[
P\{x \text{ is not } k\text{-frozen}\} = P\left\{ \bigcap_{Y} \overline{B_Y} \right\} \leq e^{-\frac{c}{2^d-1}(d-1)!} \log n.
\]

35
By Markov’s inequality, the probability that all variables are \( k \)-frozen is at least
\[
1 - ne^{-\frac{c}{2^{(d-1)(d-1)!}} \log n} = 1 - o(1),
\]
proving the lemma.

11 Appendix 1: Proof of Lemma 4.6

Proof. Recall that \( \mathcal{A} \) is the event that a random instance of \( \mathcal{F}_{k,d}^{n,p} \) induces all the edge of a fixed tree \( T \) with vertex set \( V_T \) of size \( \log n \). We estimate the number of ways that a random formula induces a copy of the tree \( T \). The counting argument used below is a generalization of the one used in [23,35].

Let \( F_T \) be a set of clauses such that every edge of \( T \) is induced by some clause in \( F_T \). We say that \( F_T \) is \emph{minimal} if deleting any clause from it leaves at least one edge of \( T \) uncovered.

Consider the different ways in which we can cover the edges of \( T \) by clauses. Treat the clauses in \( F_T \) as being grouped into \( d - 1 \) different groups \( \{S_i, 1 \leq i \leq (d-1)\} \). A clause in the group \( S_i \) is in charge of covering exactly \( i \) edges of \( T \). Note that a clause in the group \( S_i \) may “accidently” cover other edges that are not its responsibility. As long as each clause has its own dedicated set of edges to cover, there won’t be any risk of under-couting.

Let \( s_i = |S_i|, 1 \leq i \leq d - 1 \). We see that \( 0 \leq s_i \leq \log n / i \). Since each clause in \( S_i \) is dedicated to \( i \) edges and there are in total \( \log n - 1 \) edges, we have
\[
\sum_{i=1}^{d-1} is_i = \log n - 1. \tag{11.11}
\]

Counting very crudely, there are at most \( \binom{\log n}{i}^{s_i} \) ways to pick the dedicated sets of \( i \) edges for the \( s_i \) clauses in group \( S_i \). Since \( T \) is a tree, for each set of \( i \) edges there are at most \( \binom{n}{d-(i+1)} (2^d - 1) \) ways to select the corresponding clauses. Therefore, by Markov’s inequality, we have that \( \Pr \{ \mathcal{A} \} \) can be upper bounded by
\[
\sum_{0 \leq s_i \leq \log n} \left[ (\log n)^\sum_{i}^{s_i} (2^d - 1)^{s_i} \sum_{i}^{(d-i-1)s_i} \frac{c \log n}{2^d - 1} \right] \\
< O(1) \sum_{0 \leq s_i \leq \log n} (\log n)^{\log n} n^{\sum_{i}^{(-s_i)}}
\]

and due to Equation (11.11), we have

\[
\mathbb{P} \{A\} \leq O(1)(\log n)^d(\log n)^{\log n} n^{-\log n+1} \\
\leq O(1)(\log n)^{2\log n} n^{-\log n}.
\]

This proves the Lemma 4.6. \qed