Artificial Intelligence 173 (2009) 1343–1366

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

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

ARTICLE INFO

Article history:
Received 19 December 2008
Received in revised form 11 June 2009
Accepted 13 June 2009
Available online 17 June 2009

Keywords:
Weighted CNF satisfiability
Fixed parameter tractability
Data reduction
Random instances
Phase transitions
Probabilistic analysis
Resolution complexity

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 also play a crucial role in the success of many industrial-strength solvers. Understanding the effectiveness and the applicability of data reduction as a technique for designing heuristics for intractable problems has been one of the main motivations in studying the phase transition of randomly-generated instances of 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 and study the probabilistic behavior of the random instances from the model. We design an algorithm based on data reduction and other algorithmic techniques and prove that the algorithm solves the random instances with high probability and in fixed-parameter polynomial time $O(d^k nm)$ where n is the number of variables, m is the number of clauses, and k is the fixed parameter. We establish the exact threshold of the phase transition of the solution probability and show that in some region of the problem space, unsatisfiable random instances of the problem have parametric resolution proof of fixed-parameter polynomial size. Also discussed is a more general random model and the generalization of the results to the model.

1. Introduction

The theory of parameterized complexity and fixed-parameter algorithms is becoming an active research area in recent years [24,46]. Parameterized complexity provides a new perspective on hard algorithmic problems and fixed-parameter algorithms have found applications in a variety of research fields. Parameterized problems also arise in many areas of artificial intelligence (AI) research, including satisfiability, automated reasoning, logic programming, constraint programming, and probabilistic inference [8,10,47,48]. We refer the reader to [34,35] for thorough survey of recent literature.

Data reduction is a key technique in designing efficient algorithms for fixed parameter tractable problems [37,46] and exact exponential-time branch-and-reduce algorithms (also known as the search-tree method) for NP-hard problems [27]. In several areas of AI, pruning techniques based on simple and efficient-to-implement 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

☆ Part of the results appeared in the Proceedings of AAAI'08 [Y. Gao, Phase transitions and complexity of weighted satisfiability and other intractable parameterized problems, in: Proceedings of the 23th AAAI Conference on Artificial Intelligence (AAAI'08), 2008, pp. 265–270. [28]]. The research is supported by National Science and Engineering Research Council of Canada (NSERC) RGPIN 327587-06 and RGPIN 327587-09.

E-mail address: yong.gao@ubc.ca.

0004-3702/$ – see front matter
doi:10.1016/j.artint.2009.06.005

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 [36], dominating set [4], road network related problem [50], and Hamiltonian cycle [49]. Experiments also revealed that simple data reduction rules usually have a much better performance than those predicted by theoretical analyses [11,36,40].

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 [1,7,14,19,20,30–33,38,51] and references therein. Recently, the problem of detecting backdoor sets has 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 [23,48].

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 $(\mathcal{F}, k)$ of the problem consists of a d-CNF formula $\mathcal{F}$ 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 over n Boolean variables consists of a fixed parameter k and a random d-CNF formula $\mathcal{F}_{k,d}^{n,p}$ 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 literal. 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 [41] 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 design and analyze an algorithm based on data reduction and other algorithmic techniques. It is shown that the algorithm solves random instances from $\mathcal{F}_{k,d}^{n,p}$ with high probability and in fixed-parameter polynomial time $O(d^k nm)$ where n is the number of variables and m is the number of clauses in the formula. We establish the exact threshold of the phase transition of the solution probability and show that in some region of the problem space, unsatisfiable random instances of the problem have parametric resolution proof of fixed-parameter polynomial size. The results obtained in this paper give an almost complete characterization of the typical-case behavior of the random instances of WEIGHTED d-SAT.

We also discuss a more general model $\mathcal{F}_{k,d}^{n,p}(d')$, $1 < d' < d$, 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, this work is the first in the literature on the fixed-parameter tractability of random instances of intractable parameterized problems. We expect that the proposed random models and their analysis will help facilitate future studies on other parameterized problems such as the backdoor detection problem, on the solution space geometry of the standard satisfiability problem, and on the design of more effective neighborhood operators for local search algorithms.

1.1. Main results

Throughout this paper, we will use n for the number of Boolean variables, m for the number of clauses, and d for the clause size in a CNF formula. We will use k for the parameter of the WEIGHTED d-SAT problem. The symbol $p = p(n)$ is reserved for the clause probability and c for a constant usually appearing in the expression of p. All logarithms in this paper are natural logarithms, i.e., to the base e.

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 particularly 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 $\mathcal{F}_{k,d}^{n,p}(d')$.

[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.

Table 1
The behavior of random instances from $\mathcal{F}_{k,d}^{n,p}$. A mark $\surd$ 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.

Parameters	Threshold	FPT?	FPT Proof size?
WEIGHTED d-SAT $\mathcal{F}_{k,d}^{n,p}$			
$d=2$	$\surd$ (Th. 2)	$\surd$ (Th. 1)	$\surd$ (Th. 4)
$d>2$	$\surd$ (Th. 2)	$\surd$ (Th. 1)	$c>2k^2c^*$ (Th. 3)
Renormalized version of WEIGHTED d-SAT $\mathcal{F}_{k,d}^{n,p}$			
$d=2$	$\surd$ (Th. 2)	$c>c^*$ (Th. 4)	$c>c^*$ (Th. 4)
$d>2$	$\surd$ (Th. 2)	$c>kc^*$ (Corol. 1.1)	?
WEIGHTED d-SAT $\mathcal{F}_{k,d}^{n,p}(d')$, $1<d'<d$			
$d=2$	N/A	N/A	N/A
$d>2$	$\surd$ (Th. 5)	$c>2c^*$ (Th. 6)	$c>2d'(k-d')^2c^*$ (Th. 6)

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

The term d^k in the above theorem is due to the fact that we need 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 indicates, if every variable is only involved in l clauses, the term d^k can be replaced by 2^{ld}, i.e. the running time of the brute-force method for the relevant subproblems. Depending on the relation between l, k, and n, these bounds may be worse than $O(d^k)$, such as the case of $p(n)\in\Omega(\frac{k\log n}{n^{d-1}})$, or better than $O(d^k)$, such as the case of $p(n)\leqslant\frac{\log n}{n^{d-1}}$.

For random instances of the renormalized version of WEIGHTED d-SAT [46] 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 $O(d^k nm)$ 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 $c>k(2^d-1)(d-1)!$.*

To get a clear picture of the behavior of random instances of $\mathcal{F}_{k,d}^{n,p}$, 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 $(\mathcal{F}_{k,d}^{n,p}, k)$ of WEIGHTED d-SAT. We have*

$$\lim_{n\to\infty}\mathbb{P}\{\mathcal{F}_{k,d}^{n,p}\ \textit{is satisfiable}\}=\begin{cases}1, & \textit{if } c<c^*,\\ 0, & \textit{if } c>c^*,\end{cases}\tag{1.1}$$

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 algorithm in Theorem 1 can be simulated by a resolution proof. But a similar idea does lead to the following

Theorem 3. *Let $p=\frac{c\log n}{n^{d-1}}$ with $c>2k^2(2^d-1)(d-1)!$. With high probability, a random instance $(\mathcal{F}_{k,d}^{n,p}, k)$ 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 $\mathcal{F}_{k,2}^{n,p}$ where $p=\frac{3(\log n+c_1\log\log n)}{n}$ with $c_1>1$, a parametric resolution proof of size $O(kn)$ exists with high probability and can be constructed in $O(k)n^{O(1)}$ time.*

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

1.2. Outline

The next section contains preliminaries, a detailed description of the random model $\mathcal{F}_{k,d}^{n,p}$, 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 prove the results on the resolution complexity of random instances of WEIGHTED d-SAT. In Section 7, we discuss 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 some results on its threshold and resolution complexity. In the last section, we conclude and discuss directions for future work. Some necessary lemmas and their proof are included in the three appendices, which are interesting in their own right.

2. Preliminaries, random models, and related work

2.1. Parameterized complexity

An instance of a *parameterized decision problem* [24,46] 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 at most 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.

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 [46].

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

$$\text{FPT} \subseteq \text{W[1]} \subseteq \text{W[2]} \subseteq \cdots \subseteq XP.$$

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 [24,46]. Unless FPT = W[1], it is likely that no algorithm for WEIGHTED d-SAT ($d \geqslant 2$) can be more efficient that the $O(n^k)$-time brute-force algorithm that enumerates all the $\binom{n}{k}$ assignments. It has been shown recently that under an even stronger assumption, it is unlikely to solve WEIGHTED d-SAT in time $n^{o(k)}$ [16]. It can be proved that "P = NP" implies "FTP = W[1]".

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 [46]:

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 [46]:

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. [22], 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 [22]. 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) more than k variables have been assigned to true in the partial assignment.

2.4. Random models for WEIGHTED d-SAT

We use $G(n, p)$ to denote the Erdös–Rényi random graph [12] 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 $1 - o(1)$ as n goes to infinity.

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 [30,33,44] and the references therein.

Definition 2.2. Let $X = \{x_1, \ldots, x_n\}$ 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 is defined as follows:

To generate an instance 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 $\{x_{i_1}, \ldots, x_{i_d}\}$, we select a d-clause uniformly at random from the set of $2^d - 1$ non-monotone d-clauses defined over the variables $\{x_{i_1}, \ldots, x_{i_d}\}$. (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 primarily interested in is $p(n) = \frac{c \log n}{n^{d-1}}$ for some constant $c > 0$. We explain in the following the rationale of doing so.

2.4.1. WEIGHTED d-SAT instances reduce to instances with monotone clauses forbidden

In [41], Marx studied the complexity of general parameterized Boolean constraint satisfaction problems, which he calls WEIGHTED $\mathcal{F}$-SAT. He proved that $\mathcal{F}$-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 $x \to y$ is the only type of constraints. In proving these results, Marx observed that it is sufficient to consider constraints that are 0-*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 2.1. *(See Lemma 4.1, Marx [41].) An instance $(\mathcal{F}, k)$ 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 $\mathcal{F}_{k,d}^{n,p}$ is a natural model that captures the essential characteristics of a WEIGHTED d-SAT instance.

We also remark that when the clause-probability $p(n) \geqslant \frac{c \log n}{n^{d-1}}$ for some constant c, as is the case discussed in this paper, the number of variables that have a bounded number of occurrences is $o(1)$ **whp**, so that Marx's result [41] on the fixed-parameter tractability of $\mathcal{F}$-SAT with bounded variable occurrences does not apply to the random instances we are dealing with.

2.4.2. Relation to planted models for standard SAT

Even though a random instance of $\mathcal{F}_{k,d}^{n,p}$ 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, [1,38,51] and references therein.

2.4.3. Trivial random instances

We note that without forbidding monotone clauses or for $p(n) = o(\frac{\log n}{n^{d-1}})$, random instances of WEIGHTED d-SAT are trivial. This can be proved by simple random-graph arguments. We state the following observations and give the proof in Appendix C.

Lemma 2.2.

(1) *If in $\mathcal{F}_{k,d}^{n,p}$ monotone clauses are not forbidden, then **whp** there exist k disjoint monotone clauses unless $p(n)$ is extremely small. Consequently, a random instance is unsatisfiable **whp**.*
(2) *If $p(n) = \frac{c \log n}{n^{d-1}}$ and c is a small enough constant[2] or if $p(n) \in o(\frac{\log n}{n^{d-1}})$, random instances from $\mathcal{F}_{k,d}^{n,p}$ are trivially satisfiable due to the existence of a large number of variables x that are "isolated" in the sense that x does not appear as a negated literal in any clause (or, we may want to call them positive pure literals). Simply setting k of these positive pure literals to TRUE and the remaining variables to FALSE gives us a weight-k satisfying assignment.*

2.5. Relation to other random models for d-CNF formulas

In the study of the phase transition of the standard CNF satisfiability problem, several slightly different random models have been used. The most widely-studied model is $\mathcal{F}(n, m)$ on n Boolean variables where m clauses are selected from all the $\binom{n}{d}2^d$ possible clauses uniformly at random, with or without replacement.

A common aspect of these models is that for the range of m that is of interest, the likelihood is extremely low for a random formula to contain two clauses over the same set of d variables. To convince the reader, we briefly show in the following why this is the case. Let $\{C_1, C_2, \ldots, C_m\}$ be the m clauses selected with replacement uniformly at random from the $\binom{n}{d}2^d$ possible clauses. Then, for any $i \neq j$, the probability that C_i and C_j are defined over the same set of d variables is

$$\frac{2^d}{\binom{n}{d}}.$$

The expected number of pairs of clauses that are over the same set of d variables is

$$\binom{m}{2}\frac{2^d}{\binom{n}{d}}.$$

By Markov's inequality, we see that for $m = o(n^{\frac{d}{2}})$, $d > 2$, the probability that two clauses in $\mathcal{F}(n, m)$ are over the same set of d variables is asymptotically zero.

In the theory of random graphs, there are also two closely related random models $G(n, p)$ and $G(n, m)$. In $G(n, p)$ each of the possible $\binom{n}{2}$ edges appears independently with probability p. In $G(n, m)$, m edges are selected uniformly at random without replacement. It turns out that in most of the situations, it is more pleasant to work with the model $G(n, p)$.

In this paper, we study the random model $\mathcal{F}_{k,d}^{n,p}$ where exactly one of the $2^d - 1$ clauses defined on each of the $\binom{n}{d}$ possible subsets of d variables appears in the formula with probability p. The decision to focus on this model is largely due to technical convenience. Since all the properties we are concerned with in this paper, including the satisfiability, the existence of k-frozen variables, and the size of the connected components of a CNF formula, are monotone, we can regard the model $\mathcal{F}_{k,d}^{n,p}$ and the model where $p\binom{n}{d}$ clauses are selected uniformly at random as being equivalent, just as the case of random graphs (Theorem 2.2 [12]).

The fact that in all of these models, two clauses over the same set of variables rarely appear together is indeed a limitation. This is one of the reasons that in the study of the probabilistic behavior of standard CNF formulas, researchers also study random models of the so-called generalized satisfiability problem where for each set of randomly selected variables, several clauses are selected. We believe that results similar to those in this paper can also be obtained for random instances of the generalized satisfiability problem and the constraint satisfaction problem.

[2] For the case of $d = 2$, $c < \frac{3}{2}$.

2.6. Residual graphs of CNF formulas and induced formulas

Associated with a CNF formula is its **residual graph** (also known as Gaifman 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 $\mathcal{F}_{k,2}^{n,p}$ is exactly the random graph $G(n, p)$. The residual graph of a random instance of $\mathcal{F}_{k,d}^{n,p}$ is the **primal graph**[3] of the random hypergraph $\mathcal{G}(n, p, d)$.

By a **connected component** of a CNF formula $\mathcal{F}$, we mean a maximal sub-formula $\mathcal{F}'$ such that the residual graph of $\mathcal{F}'$ is a connected component of the residual graph of the CNF formula $\mathcal{F}$.

Let $\mathcal{F}$ be a d-CNF formula and $V \subset X$ be a subset of variables. The **induced formula** $\mathcal{F}_V$ over V is defined as a CNF formula $\mathcal{F}_V$ that consists of the following two types of clauses:

(1) the clauses of $\mathcal{F}$ 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, $\mathcal{F}_V$ contains clauses of the first type only. In the above definition of an induced formula on V, we do not include "induced" unit clauses, i.e. clauses of size 1 obtained by removing any literal whose corresponding variable is in $X \setminus V$. This makes it easier and more clear to describe our algorithm and its analysis. For more details see the proof of Theorem 1 at the end of Section 4.

2.7. 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** [2,19,30–33,43,44]. The threshold behavior of random CNF satisfiability, general constraint satisfaction problems, and graph coloring on random graphs has attracted a lot of attention, and determining the exact threshold of their phase transitions remains to be a challenging open problem [2].

There have been many empirical studies on the power of data reduction as a heuristic for hard problems under different settings. Here, we mention two examples that are motivated by the study of phase transitions of random problem instances. In [49], a backtracking algorithm with reduction rules based on local vertex degree information is shown to be 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 [13,21,45]. Whereas identifying backbone variables is not an easy task itself, it has been shown in [52] that even statistical information on the backbone variables can be utilized to guide the variable selection of local search algorithms to obtain great performance improvement.

There has also been recent interest in designing **whp** polynomial-time algorithms for random instances with *planted solutions* (*a.k.a. hidden solutions*) [5,15,18,26,39] and using random instances with planted solutions for empirical studies [1,38,51]. In [30], random instances of the constraint satisfaction problem are designed that include a hidden consistency core rather than a planted solution.

Weighted CNF satisfiability is a generic intractable parameterized problem. In addition to the work of Marx [41] we have mentioned in the previous subsection, other work that motivated the current research on random instances of intractable parameterized problems include the study of the parameterized proof complexity of weighted CNF satisfiability recently initiated by Dantchev et al. [22] and the study of the *backdoor set detection* problem and its parameterized complexity in the recent AI literature [23,47,48].

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 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] The primal graph of a hypergraph is the graph where a pair of vertices is an edge if and only if the two vertices appear simultaneously in some hyperedge.

3.1.1. STEP 1: Data reduction

We use a data reduction rule similar to Buss's reduction for vertex cover [24,46]: If a variable x appears in a set of $l \geqslant k$ clauses of the form

$$\begin{cases} \bar{x} \vee y_{11} \vee \cdots \vee y_{1(d-1)} \\ \bar{x} \vee y_{21} \vee \cdots \vee y_{2(d-1)} \\ \cdots \\ \bar{x} \vee y_{l1} \vee \cdots \vee y_{l(d-1)} \end{cases}$$

such that the set of monotone clauses

$$\begin{cases} y_{11} \vee \cdots \vee y_{1(d-1)} \\ y_{21} \vee \cdots \vee y_{2(d-1)} \\ \cdots \\ y_{l1} \vee \cdots \vee y_{l(d-1)} \end{cases}$$

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 $\bar{x} \vee y$. For WEIGHTED d-SAT with $d > 2$, we will show later that there is an $O(d^k nm)$-time algorithm that implements the reduction correctly.

Unlike the case of vertex cover, the above reduction rule does not 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" each of which contains at most $\log n$ variables. Section 4 is devoted to proving this observation.

Let $\{\mathcal{F}_i,\ 1 \leqslant i \leqslant N\}$ where $N \leqslant n$ be the collection of connected components in the reduced formula. For each connected component $\mathcal{F}_i$, 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 $\mathcal{F}_i$ that satisfies $\mathcal{F}_i$.

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

$$\begin{cases} k_{i_j} \in L_{i_j}, \quad \text{and} \\ k_{i_1} + k_{i_2} + \cdots + k_{i_k} = k. \end{cases}$$

3.1.3. A further remark

A typical criticism to the above scheme is that it is more about the structure of the random instance distribution than the design of algorithms and heuristics. 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. Our algorithm and its analysis is exactly a demonstration of the kind of structures that may appear in a typical random instance of an intractable problem and can be exploited by reduction rules similar to those that have led to efficient algorithms for tractable problems.

We note that similar structures have been exploited in many studies, most notably in N. Alon and N. Kahale's seminal work on algorithms for coloring random 3-colorable graphs [5] and its follow-ups [15,18,26,39]. In the author's previous work [29], similar connected-component structure also arises in a different setting.

3.2. Definitions: k-frozen variables

We formalize the concepts mentioned in Section 3.1 that will be used in the description and the analysis of the algorithm.

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

$$Y_i = \{y_{ij},\ 1 \leqslant j \leqslant d-1\}$$

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

(1) for each $1 \leqslant i \leqslant l$, the clause $\bar{x} \vee y_{i1} \vee \cdots \vee y_{i(d-1)}$ is in the formula $\mathcal{F}$;
(2) the set of clauses $\{y_{i1} \vee \cdots \vee y_{i(d-1)},\ 1 \leqslant i \leqslant l\}$ 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 $\{Y_i,\ 1 \leqslant i \leqslant l\}$ such that $Y_i \subset V,\ \forall 1 \leqslant i \leqslant l$.

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) [13,21,45]. 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 [52]. 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 [18,26,39].

Our notion of k-frozen variables can be viewed as a fixed-parameter generalization of a stronger version of the concept of backbone variables. A k-frozen variable has a forced value in all satisfying assignments, but a variable that has a forced value in all satisfying assignments is not necessarily k-frozen. As we will show later unlike general backbone variables, k-frozen variables can be detected efficiently: For 2-CNF formulas, to see if a variable x is k-frozen, one only needs to count the number of clauses of the form $\bar{x} \vee y$. For d-CNF formulas with $d > 2$, k-frozen variables can be detected by an $O(d^k nm)$-time algorithm.

In the analysis of the probabilistic behavior of the algorithm in Section 4, we will make use of an even stronger notion of k-frozenness. We say that a variable x in a d-CNF formula $\mathcal{F}$ is **strongly k-frozen**[4] if there is a collection of k clauses $\{\bar{x} \vee y_{i1} \vee \cdots \vee y_{i(d-1)},\ 1 \leqslant i \leqslant k\}$ in $\mathcal{F}$ such that $Y_i = \{y_{i1}, \ldots, y_{i(d-1)}\},\ 1 \leqslant i \leqslant k$, are variable-disjoint subsets. A strongly k-frozen variable is obviously also k-frozen. The chief reason to focus on strongly k-frozen variables is that it is not clear if we can achieve the same or better bounds on some relevant probabilities by considering the k-frozen variables. We note that the task of deciding a strongly k-frozen variable is equivalent to a $(d-1)$-dimensional matching problem, which is NP-complete for $d \geqslant 4$ and polynomial solvable for $d = 2$ or 3 if k is part of the input. For fixed k as we are dealing with in this paper, the $(d-1)$-dimensional matching problem is fixed parameter tractable, but the currently best algorithms all have an exponential dependency on the parameter k which is worse than that of the straightforward bounded search-tree algorithm for the parameterized hitting set problem [17,25].

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 $O(d^k L)$-time algorithm that checks if a variable x is k-frozen where L is the number of clauses that contain $\bar{x}$. The running time is $O(n^2)$ for a 2-CNF formula.*

Proof. Consider the set of all clauses in which x is the only negated literal: $\{\bar{x} \vee y_{i1} \vee \cdots \vee y_{i(d-1)},\ 1 \leqslant i \leqslant l\}$. According to our definition, x is k-frozen if and only if the collection of subsets $\{\{y_{i1}, \ldots, y_{i(d-1)}\},\ 1 \leqslant i \leqslant l\}$ has no hitting set[5] of size at most $k-1$, which can be solved by the bounded search tree method in time $O(d^k L)$ that branches on the element of a subset [46].

In the case of 2-CNF formulas, one only needs to count the number of clauses of the form $\bar{x} \vee y$ to see if x is k-frozen. □

We remark that for random instances from $\mathcal{F}_{k,d}^{n,p}$ with $p = \frac{c \log n}{n^{d-1}}$, the above lemma is not really necessary since it can be shown by Chernoff inequality and Markov inequality that **whp** every variable appears only in about $c \log n$ clauses. Consequently, a brute-force search to check k-frozen variables runs in $O(n^c m)$ time. Lemma 3.1 is necessary in order to deal with the case of $p = \Omega(\frac{1}{n^{d-1-\epsilon}})$, $\epsilon > 0$, and to avoid the exponential dependency on c of the running time.

We now show that 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 $O(d^k nm)$ for any $p(n) \leqslant 1$ where m is the number of clauses in the d-CNF formula.*

Proof. Since k-frozen variables are forced to take the truth value FALSE and since the subroutine 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

[4] We thank one of the referees for suggesting this name.

[5] A hitting set H of a collection of subsets $\{S_1, \ldots, S_n\}$ in a universe U is a subset $H \subset U$ such that H contains at least one element from each subset S_i.

Input: A random instance $(\mathcal{F}^{n,p}_{k,d}, k)$ 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 and let $U = U \cup \{x\}$.
3. Let $\mathcal{F}' = \text{REDUCE}(\mathcal{F}, U)$ be the **reduced formula**.
4. Find the connected components $\{\mathcal{F}_1, \ldots, \mathcal{F}_N\}$ of $\mathcal{F}'$.
5. If there is a connected component that contains more than $\log n$ variables, return "FAILURE".
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 $\{i_j,\ 1 \leqslant j \leqslant k\}$ and a set of integers $\{k_{i_j},\ 1 \leqslant j \leqslant k\}$ 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 $\mathcal{F}$.

Algorithm 1. W-SAT.

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 $O(d^k nm)$-time to find all the k-frozen variables in Lines 1 and 2. Lines 3 through 5 take at most $O(nm)$-time and do not depend on k. Line 6 can be finished in $O(nm)$-time to **enumerate** all the elements of $L_{\mathcal{F}_i}$ since each $\mathcal{F}_i$ contains no more than $\log n$ variables. Line 8 repeats part of the work done in Line 6.

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

$$\sum_{j=1}^{l} k_{i_j} = a.$$

We call any such index set I_a a **representative set** for a. The purpose of Line 7 is to check if the integer k is **achievable**, and if YES, return a representative set for k. The next lemma shows how to implement this task.

Lemma 3.2. *Given an integer k and a collection $\{L_i,\ 1 \leqslant i \leqslant l\}$ of subsets of integers from $\{0, 1, \ldots, k\}$, there is a dynamic programming algorithm that finds a representative set for k if k is achievable, or reports that k is not achievable. It runs in time $O(k^2 l)$.*

Proof. Let $A(t) = \{(a, I_a)\colon\ 0 \leqslant a \leqslant k\}$ be the set of pairs (a, I_a) where $0 \leqslant a \leqslant k$ is an integer achievable by $\{L_i,\ 1 \leqslant i \leqslant t\}$ and I_a is a representative set for 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 $((a+b), \bar{I}_a)$ where

$$\begin{cases} (a, I_a) \in A(t), \\ b \in L_{t+1} \quad \text{such that } a + b \leqslant k, \quad \text{and} \\ \bar{I}_a = I_a \cup \{t\}. \end{cases}$$

A typical application of dynamic programming computes

$$A(0), \quad A(1), \quad \ldots, \quad \text{and} \quad A(l).$$

The value k is achievable by $\{L_i, 1 \leqslant i \leqslant l\}$ if and only if there is a pair (k, I_k) in $A(l)$. Since the size of $A(t)$ is at most k, the above algorithm runs in $O(k^2 l)$ time. □

For the case of Line 7, there are at most n subsets of integers, each of which corresponds to the $L_{\mathcal{F}_i}$ of a connected component $\mathcal{F}_i$. The running time of Line 7 is thus $O(k^2 n)$. The proposition follows. □

3.4. *The subroutine* REDUCE$(\mathcal{F}, U)$

The purpose of the subroutine REDUCE$(\mathcal{F}, U)$ is to simplify the formula $\mathcal{F}$ 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 $\mathcal{F}$ contains no monotone clause. As a consequence, REDUCE( ) will never create a contradiction either.

It is possible that REDUCE( ) returns an empty formula $\mathcal{F}'$, signifying that all the clauses have been satisfied during the process. However, since we are searching for a 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 $\mathcal{F}' = \text{REDUCE}(\mathcal{F}, U)$ is empty, then $\mathcal{F}$ has a weight-k satisfying assignment if and only if at least k variables have not been assigned after* REDUCE *returns.*

Proof. $\mathcal{F}'$ 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 $\mathcal{F}'$ is empty, then any variable not assigned yet can be assigned an arbitrary truth value. □

4. The algorithm W-SAT succeeds with high probability

In this section, we prove that the algorithm W-SAT succeeds **whp** for random instances of $\mathcal{F}_{k,d}^{n,p}$. Due to Proposition 3.1, we only need to show that the probability for W-SAT to report "FAILURE" is asymptotically zero. Recall that W-SAT fails only when the reduced formula $\mathcal{F}'$ obtained in Line 3 has a connected component that contains more than $\log n$ variables.

First, we present a result for the case of $p(n) = \frac{c \log n}{n^{d-1}}$ with $c > 2k^2(2^d - 1)(d-1)!$, which will be used later in the proof of Theorem 3. We show in the following lemma that **whp** all variables are k-frozen and have to be set to FALSE. Consequently the reduced formula is empty and the algorithm returns the correct answer "NO".

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

Proof. See Appendix A. □

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

Proposition 4.1. *Let $\mathcal{F} = \mathcal{F}_{k,d}^{n,p}$ be the input random CNF formula to W-SAT and V be the set of variables that are not k-frozen. With high probability, the residual graph of the induced formula $\mathcal{F}_V$ on V decomposes into a collection of connected components each of which contains at most* $\log n$ *variables.*

Proof. Let $X = \{x_1, \ldots, x_n\}$ be the set of Boolean variables, U be the set of k-frozen variables, and $V = X \setminus U$.

Since $p = \frac{c \log n}{n^{d-1}}$ 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 in our case is that 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 $\mathcal{F}_{k,d}^{n,p}$ 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 tends to zero for the residual graph of $\mathcal{F}_V$ to have a connected component over more than $\log n$ variables.

Let T be a fixed tree over a subset V_T of $\log n$ variables. A further complication in estimating P^* is that the two events

(1) $\mathcal{F}_{k,d}^{n,p}$ induces T and
(2) no variable in T is k-frozen

are not independent of each other. To further decouple the dependency, we consider the following two events

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

$$\{\bar{x} \vee y_{i1} \vee \cdots \vee y_{i(d-1)},\ 1 \leqslant i \leqslant k\}$$

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

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

$$\{\bar{x} \vee y_{i1} \vee \cdots \vee y_{i(d-1)},\ 1 \leqslant i \leqslant k\}$$

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

$$P^* \leqslant \mathbb{P}\{\mathcal{A} \cap \mathcal{B}\}. \tag{4.2}$$

We claim that

Lemma 4.2. *The two events $\mathcal{A}$ and $\mathcal{B}$ are independent, i.e.,*

$$\mathbb{P}\{\mathcal{A}|\mathcal{B}\} = \mathbb{P}\{\mathcal{B}\}. \tag{4.3}$$

Proof. Note that the event $\mathcal{A}$ depends only on those d-clauses that contain at least two variables in V_T, and that the event $\mathcal{B}$ depends only on those d-clauses that contain exactly one variable from V_T. By 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 Eq. (4.2) and Lemma 4.2, we only need to estimate $\mathbb{P}\{\mathcal{A}\}$ and $\mathbb{P}\{\mathcal{B}\}$ 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 μ. We have*

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

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 $x \notin W$ and $|W| = n - \log n$. Then, we have*

$$\mathbb{P}\{x \text{ is not } k\text{-frozen with respect to } W\} \leqslant O(1) \max\left(\frac{1}{n^{\delta}}, \frac{\log^2 n}{n}\right) \tag{4.4}$$

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

Proof. Let N_x be the number of clauses of the form $\bar{x} \vee y_1 \vee \cdots \vee y_{d-1}$ with $\{y_1, \ldots, y_{d-1}\} \subset W$. Due to the definition of $\mathcal{F}_{k,d}^{n,p}$, the random variable N_x follows the binomial distribution $\mathrm{Bin}(\overline{p}, m)$ where $\overline{p} = \frac{1}{2^d-1}\frac{c\log n}{n^{d-1}}$ and $m = \binom{n-\log n}{d-1}$. Note that for any fixed constant $\epsilon > 0$, there is an integer $N(\epsilon) > 0$ such that for $n > N(\epsilon)$

$$\begin{aligned}
\overline{p}m &= \frac{1}{2^d-1}\frac{c\log n}{n^{d-1}}\binom{n-\log n}{d-1} \\
&\geqslant \frac{c}{(2^d-1)(d-1)!}\left(1 - \frac{\log n + d}{n}\right)^{d-1} \\
&\geqslant \frac{(1-\epsilon)c}{(2^d-1)(d-1)!}\log n.
\end{aligned}$$

Write $\alpha = \frac{c}{(2^d-1)(d-1)!}$. By Lemma 4.3 and for sufficiently large n such that $\frac{k}{\overline{p}m} < \epsilon$, we have

$$\begin{aligned}
\mathbb{P}\{N_x < k\} &\leqslant \mathbb{P}\left\{|N_x - \overline{p}m| > \overline{p}m - k\right\} \\
&\leqslant 2e^{-\frac{(\overline{p}m-k)^2}{3\overline{p}m}} \\
&= 2e^{-\frac{1}{3}\overline{p}m(1-\frac{k}{\overline{p}m})^2} \\
&\leqslant 2e^{-\frac{1}{3}(1-\epsilon)^3\alpha\log n} \\
&\in O\left(n^{-\delta}\right)
\end{aligned} \tag{4.5}$$

for some $0 < \delta < \frac{\alpha}{3}$.

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

$$\begin{cases} \bar{x} \vee y_{11} \vee \cdots \vee y_{1(d-1)}, & \text{and} \\ \bar{x} \vee y_{12} \vee \cdots \vee y_{2(d-1)} \end{cases}$$

such that $\{y_{11}, \ldots, y_{1(d-1)}\} \cap \{y_{12}, \ldots, y_{2(d-1)}\} \neq \emptyset$. 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

$$\mathbb{P}\{\mathcal{D}\} \in O\left(\frac{\log^2 n}{n}\right).$$

By definition, if a variable x is not k-frozen, then either $N_x < k$ or the event $\mathcal{D}$ occurs. Therefore, the probability that the variable x is not k-frozen is at most

$$\mathbb{P}\big\{\{N_x < k\} \cup \mathcal{D}\big\}.$$

The lemma follows. □

From Lemma 4.4, we have

Lemma 4.5. *For sufficiently large n,*

$$\mathbb{P}\{\mathcal{B}\} < O(1)\big(n^{-\delta}\big)^{\log n} \tag{4.6}$$

for some $0 < \delta < \min(\frac{c}{3(2^d-1)(d-1)!}, 1)$.

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 $|V_T|$ is $\log n$, 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

$$\bar{x} \vee y_{i1} \vee \cdots \vee y_{i(d-1)}$$

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

Next, we have the following upper bound on the probability $\mathbb{P}\{\mathcal{A}\}$. Its proof is based on a counting argument that slightly generalizes that used in [26,39]. See Appendix B for the details.

Lemma 4.6. *The probability of the event $\mathcal{A}$ can be bounded as*

$$\mathbb{P}\{\mathcal{A}\} \leqslant (\log n)^{d-1} n^{f(d,c)+1} (\log n)^{2\log n} n^{-\log n}$$

where $f(d, c)$ is a function that only depends on d and c.

Continuing the proof of Proposition 4.1, we combine the results of Lemma 4.5 and Lemma 4.6 to get

$$\mathbb{P}\{\mathcal{A} \cap \mathcal{B}\} \leqslant (\log n)^{d-1} n^{f(d,c)+1} (\log n)^{2\log n} n^{-\log n} \big(n^{-\delta}\big)^{\log n}.$$

Since the total number of trees of size $\log n$ over n vertices is at most

$$n^{\log n}(\log n)^{\log n - 2},$$

the probability that the induced formula $\mathcal{F}_V$ has a connected component over more than $\log n$ variables can be upper bounded by

$$n^{\log n}(\log n)^{\log n-2}\mathbb{P}\{\mathcal{A} \cap \mathcal{B}\} \leqslant (\log n)^{d-1} n^{f(d,c)+1} (\log n)^{3\log n} \big(n^{-\delta}\big)^{\log n} \tag{4.7}$$

which tends to zero since the first three terms on the right-hand side of the above are in $o(n^{\epsilon \log n})$ for any $\epsilon > 0$. Proposition 4.1 follows. □

Proof of Theorem 1. Recall that by our definition the induced formula $\mathcal{F}_V$ on the set V of variables that are not k-frozen consists of two types of clauses:

(1) the clauses of $\mathcal{F}$ 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 $X \setminus V$.

We claim that the formula $\mathcal{F}'$ obtained in Line 3 of the algorithm W-SAT is a sub-formula of $\mathcal{F}_V$ and is thus sparser. This is because any clause that is still not satisfied after the application of REDUCE( ) must contain at least two literals from the variables in V. Therefore by Proposition 4.1, with high probability $\mathcal{F}'$ decomposes into a collection of connected components each of which contains at most $\log n$ variables. 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 $\mathcal{F}_{k,d}^{n,p}$. 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 phase transition of the solution probability. Unlike the threshold for random instances of most 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 of Theorem 2. Let $\mathcal{T}$ be the collection of all subsets of d variables and let s be an assignment to the variables. We say that a subset $T = \{x_1, \ldots, x_d\} \in \mathcal{T}$ 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 $\mathcal{F}_{k,d}^{n,p}$, the probability for $T \in \mathcal{T}$ to be s-good is

$$\mathbb{P}\{T \text{ is } s\text{-good}\} = \begin{cases} 1, & \text{if } T \cap Y = \emptyset; \\ 1 - p(n)\frac{1}{2^d - 1}, & \text{otherwise.} \end{cases} \tag{5.8}$$

Let $\mathcal{T}_s \subset \mathcal{T}$ be the collection of subsets of d variables that have a non-empty intersection with Y. We have

$$|\mathcal{T}_s| = \sum_{j=1}^{d} \binom{n-k}{d-j}\binom{k}{j}.$$

Note that in the above summation, the first term $\binom{n-k}{d-1}\binom{k}{1}$ dominates. Let X be the number of assignments in S that satisfy $\mathcal{F}_{k,d}^{n,p}$. We have

$$\begin{aligned} \mathbb{E}[X] &= \sum_{s \in S} \prod_{T \in \mathcal{T}} \mathbb{P}\{T \text{ is } s\text{-good}\} \\ &= \sum_{s \in S} \prod_{T \in \mathcal{T}_s} \mathbb{P}\{T \text{ is } s\text{-good}\} \\ &= \binom{n}{k}\left(1 - \frac{c \log n}{n^{d-1}} \frac{1}{2^d - 1}\right)^{\sum_{j=1}^{d} \binom{n-k}{d-j}\binom{k}{j}} \\ &\sim \binom{n}{k}\left(1 - \frac{c \log n}{n^{d-1}} \frac{1}{2^d - 1}\right)^{\binom{n-k}{d-1}\binom{k}{1}} \\ &\sim n^k e^{-\frac{kc \log n}{(2^d-1)(d-1)!}} \end{aligned} \tag{5.9}$$

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 Chebyschev's inequality

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

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

$$\mathbb{E}\big[X^2\big] = \sum_{i=1}^{k} \mathbb{E}[D_i]. \tag{5.10}$$

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

$$\begin{cases} \mathbb{E}[D_i] \in o\big(n^{2k\epsilon - i\epsilon}\big) & \text{for } i > 1, \quad \text{and} \\ \lim_n D_0 = \big(\mathbb{E}[X]\big)^2. \end{cases} \tag{5.11}$$

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 assigned TRUE in s_1 is $\{y_1, \ldots, y_{k-i}, y_{k-i+1}, \ldots, y_k\}$ and the set of variables assigned TRUE in 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 $\mathbb{P}\{T \text{ is } s_1\text{-good and } s_2\text{-good}\}$ is at most $(1 - p(n)\frac{1}{2^d - 1})$. 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

$$\mathbb{P}\{T \text{ is } s_1\text{-good}\} = \left(1 - p(n)\frac{1}{2^d - 1}\right), \quad \text{or} \quad \mathbb{P}\{T \text{ is } s_2\text{-good}\} = \left(1 - p(n)\frac{1}{2^d - 1}\right).$$

As 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 $i > 1$,

$$\begin{aligned} \mathbb{E}[D_i] &= \binom{n}{k}\binom{n-k}{k-i} \prod_{T:\, T \cap Y \neq \emptyset} \left(1 - p(n)\frac{1}{2^d - 1}\right) \\ &\sim n^{2k-i}\left(1 - p(n)\frac{1}{2^d - 1}\right)^{\sum_{j=1}^{d} \binom{n-(2k-i)}{d-j}\binom{2k-i}{j}} \\ &\sim n^{2k-i} e^{-\frac{(2k-i)c\log n}{(2^d-1)(d-1)!}} \\ &\sim n^{2k\epsilon - i\epsilon}, \end{aligned} \tag{5.12}$$

where $\sim$ means "is asymptotically equivalent to". For the case of $i = 0$, it can be shown that $\lim_n D_0 = (\mathbb{E}[X])^2$. This proves the claim. The theorem follows from Eqs. (5.9), (5.10), and (5.12).

Finally, we note that all the calculations still hold for random instances of MINI-WEIGHTED d-SAT where the question is to find a satisfying assignment with Hamming distance $k \log n$ to the all-zero assignment. □

6. 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, it is unclear to us how to simulate the dynamic programming phase (Line 7) of W-SAT by 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 begin with Theorem 3 that deals with the case of $d > 2$. The proof is based on 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 $\bar{x}$ of size $O(d^k)$ can be constructed.

For random WEIGHTED 2-SAT, results on the minimum degree in a random graph (Theorem 3.5 in [12]) 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 instead 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.

The study of the parameterized proof complexity of weighted CNF satisfiability has been recently initiated by Dantchev et al. [22], who established lower bounds on the parameterized resolution proof for CNF formulas that encode some first-order combinatorial principle. Following their definition, a parameterized resolution proof system for unsatisfiable instances of a co-W[1] complete problem is defined to be a classical resolution system that has access for free to all clauses that contains more than k negated literals or more than $n-k$ positive literals where k is the problem parameter.

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

Proof of Theorem 3. We establish Theorem 3 by making use of the fact established in Lemma 4.1 that for $p = \frac{c \log n}{n^{d-1}}$ with $c > 2k^2(2^d-1)(d-1)!$, all the variables are k-frozen **whp**.

Since all variables are k-frozen, one can construct a parametric (tree) resolution proof as follows. First, 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 $\bar{x}$ of size $O(d^k)$.*

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

$$\begin{cases} \bar{x} \vee y_{11} \vee \cdots \vee y_{1(d-1)} \\ \cdots \\ \bar{x} \vee y_{k1} \vee \cdots \vee y_{k(d-1)} \end{cases}$$

that freeze x.

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 $\{y_{ij},\ 1 \leqslant j \leqslant (d-1)\}$, $1 \leqslant i \leqslant k$, in order to satisfy the k clauses in $\mathcal{C}$ with $\bar{x}$ 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. Since in a parametric resolution proof the proof system has free access to all anti-monotone clauses of size $k+1$, 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 $\bar{x}$ can be constructed from the above search tree. □

Continuing the proof of Theorem 3, it follows from the above lemma that there is a size $O(d^k n)$ parametric resolution derivation of the set of single-literal clauses $\{\bar{x}\}$, $x \in X$. From the set of clauses $\{\bar{x}\}$, $x \in X$, together with the monotone clause $\bigvee_{i=1}^{n} x_i$ (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 $\mathcal{F}^{n,p}_{k,2}$

Proof of Theorem 4. We prove Theorem 4 on the resolution complexity of $\mathcal{F}^{n,p}_{k,2}$ 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 $\{x_1, \ldots, x_n\}$ of the variables. If $\mathcal{F}$ contains the following cycle of "forcing" clauses,*

$$\bar{x}_1 \vee x_2, \quad \bar{x}_2 \vee x_3, \quad \ldots, \quad \bar{x}_{n-1} \vee x_n, \quad \bar{x}_n \vee x_1,$$

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. Since 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 $i \geqslant 1$, we can resolve

$$\bar{x}_i \vee x_{i+1}, \quad \ldots, \quad \bar{x}_{i+k} \vee x_{i+k+1}$$

and $\bar{x}_{i+1} \vee \cdots \vee \bar{x}_{i+k+1}$ to derive $\bar{x}_i$. These $\bar{x}_i$'s together with $x_1 \vee \cdots \vee x_{n-k+1}$ 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 $\mathcal{F}_{k,2}^{n,p}$, 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 the "directed random graph" defined as follows.

Definition 6.1. (See Section 4, McDiarmid [42].) Let V be a vertex set and $p > 0$. The directed random graph $D_p^2(V, E)$ is defined as a directed graph where each directed edge is in E with probability p under the constraints that

(1) the appearance of directed edges on different unordered pairs are independent, and
(2) for any pair of vertices u and v,

$$\mathbb{P}\big\{(u, v) \in E \text{ and } (v, u) \in E\big\} = \max\{0, 2p - 1\}.$$

It is noted in [42] that if $p < \frac{1}{2}$, D_p^2 can be obtained by randomly directing the edges in the undirected random graph $G(n, 2p)$.

Lemma 6.3. *(See Theorem 4.4, McDiarmid [42].) Let $0 < p < 1$ and D_p^2 be a random directed graph on n vertices. Then*

$$\mathbb{P}\big\{D_p^2 \textit{ has a directed Hamiltonian cycle}\big\} \geqslant \mathbb{P}\big\{G(n, p) \textit{ has a Hamiltonian cycle}\big\}.$$

By Lemma 6.2, it is sufficient to show that there is a directed Hamiltonian cycle in the following directed graph $D_{\mathcal{F}}(n, p')$: Each vertex corresponds to a variable. There is a directed edge from x_i to x_j if and only if the 2-clause $\bar{x}_i \vee x_j$ is in the random formula $\mathcal{F}_{k,2}^{n,p}$.

By the definition of $\mathcal{F}_{k,2}^{n,p}$, $D_{\mathcal{F}}(n, p')$ is exactly the directed random directed graph $D_{p'}^2$ discussed in Lemma 6.3 with edge probability $p' = \frac{1}{3}p$. To see this, consider the random graph $G(X, E)$ on the set of variables such that $(x, y) \in E$ if and only if either $\bar{x} \vee y$ or $x \vee \bar{y}$ is in $\mathcal{F}_{k,2}^{n,p}$. Thus, $G(X, E)$ is a random graph with edge probability $\frac{2}{3}p$. The directed graph $D_{\mathcal{F}}(n, p')$ follows the distribution of a directed random graph by randomly directing the edge of $G(X, E)$.

The result follows from Lemma 6.3 and the threshold of the existence of a Hamiltonian cycle in the random graph $G(n, p')$: for edge probability $p' = \frac{\log n + b \log\log n}{n}$ with $b > 1$, $G(n, p')$ is Hamiltonian (Theorem 8.9 in [12]). □

7. W-SAT for MINI-WEIGHTED *d*-SAT

In this section, we show that a simple modification of the algorithm W-SAT can solve **whp** random instances of MINI-WEIGHTED d-SAT from $\mathcal{F}_{k,d}^{n,p}$ when $p = p(n)$ is in a certain range, and thus prove Corollary 1.1.

Recall that for a random instance $(\mathcal{F}_{k,d}^{n,p}, k)$ of MIN-WEIGHTED d-SAT, we are looking for a satisfying assignment of weight $k \log n$. Thus, the algorithm W-SAT needs to 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 not $k \log n$-frozen is small enough. For $p = \frac{c \log n}{n^{d-1}}$ with $c > k(2^d - 1)(d - 1)!$, this is the case.

Proposition 7.1. *Let $p = \frac{c \log n}{n^{d-1}}$ with $c > k(2^d - 1)(d - 1)!$. Let $\mathcal{F} = \mathcal{F}_{k,d}^{n,p}$ be the input random CNF formula to W-SAT customized to MINI-WEIGHTED d-SAT (i.e., based on $k \log n$-frozen variables) and V be the set of variables that are not $k \log n$-frozen. With high probability, the residual graph of the induced-formula $\mathcal{F}_V$ decomposes into a collection of connected components each of which contains 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 works. Using the notation in the proof of Lemma 4.4, we have

$$\begin{aligned}
\mathbb{P}\{N_x < k \log n\} &\leqslant \mathbb{P}\big\{|N_x - \bar{p}m| > \bar{p}m - k \log n\big\} \\
&\leqslant 2e^{-\frac{(\bar{p}m - k \log n)^2}{3\bar{p}m}} \\
&= 2e^{-\frac{1}{3}\bar{p}m(1 - \frac{k \log n}{\bar{p}m})^2} \\
&\leqslant 2e^{-\frac{1}{3}(1-\epsilon) f(c,d,k) \log n}
\end{aligned}$$

where $f(c, d, k)$ depends on c, d, k only. 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, but it is sufficient for the result to hold. □

A further complication is the detection of $k\log n$-frozen variables. Note that the bounded search-tree method in Lemma 3.1 does not work since the resulting running time would be $O(d^{k\log n}) = O(n^k)$. To overcome this difficulty, we make use of the fact that with $p(n) = \frac{c\log n}{n^{d-1}}$ where c is a constant, the set of clauses in which a given variable x appears also decomposes into connected components each of which contains at most $\log n$ variables, and consequently we can check whether a variable is $k\log n$-frozen by using brute-force on each of connected components. We make the observation precise in the following proposition.

Proposition 7.2. *Let $\mathcal{F}_{k,d}^{n,p}$ be a random d-CNF formula and x be a variable. Consider the $(d-1)$-CNF formula $\mathcal{F}(x)$ consists of all the clauses $y_1 \vee \cdots \vee y_{d-1}$ such that $\bar{x} \vee y_1 \vee \cdots \vee y_{d-1}$ is in $\mathcal{F}_{k,d}^{n,p}$. Let E be the event that $\mathcal{F}(x)$ has a connected component on $\log n$ variables. We have*

$$\mathbb{P}\{E\} \leqslant n^{f(d,c)}(\log n)^{2\log n} n^{-\frac{1}{d-2}\log n}.$$

Proof. Let T be a fixed tree on a vertex set $V_T \subset X \setminus \{x\}$ of size $\log n$. Similar to the argument in the proof of Lemma 4.6, we have that the probability for the residual graph of the $(d-1)$-CNF formula $\mathcal{F}(x)$ to contain the tree T as its subgraph is at most

$$(\log n)^{d-2} n^{f(d,c)} (\log n)^{2\log n} n^{-\log n - \frac{1}{d-2}\log n}. \tag{7.13}$$

As the number of trees of size $\log n$ is $n^{\log n}(\log n)^{\log n - 2}$, the proposition follows. □

By Markov's inequality and Proposition 7.2, we have that **whp** all the variables x are such that their corresponding $(d-1)$-CNF formula $\mathcal{F}(x)$ has no connected component on more than $\log n$ variables.

8. A more general model $\mathcal{F}_{k,d}^{n,p}(d')$

In this section, we discuss how to generalize the model $\mathcal{F}_{k,d}^{n,p}$. Let $1 \leqslant d' \leqslant d$ and consider the model $\mathcal{F}_{k,d}^{n,p}(d')$ defined as follows: instead of from the set of non-monotone clauses, we select uniformly at random from the set of clauses over $\{x_{i_1}, \ldots, x_{i_d}\}$ that contain at least d' negated literals. Note that $\mathcal{F}_{k,d}^{n,p}$ is just $\mathcal{F}_{k,d}^{n,p}(1)$.

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

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

$$\lim_n \mathbb{P}\left\{\mathcal{F}_{k,d}^{n,p}(d') \text{ is satisfiable}\right\} = \begin{cases} 1, & \text{if } c < c^*, \\ 0, & \text{if } c > c^*. \end{cases}$$

Proof. Same as the proof of Theorem 2. □

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

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

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 *frozen tuple* if the following $k-d'$ clauses are in the random CNF formula:

$$\begin{cases} \bar{x}_{i_1} \vee \cdots \vee \bar{x}_{i_{d'}} \vee y_{11} \vee \cdots \vee y_{1(d-d')} \\ \cdots \\ \bar{x}_{i_1} \vee \cdots \vee \bar{x}_{i_{d'}} \vee y_{(k-d')1} \vee \cdots \vee y_{(k-d')(d-d')} \end{cases}$$

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.

We claim that for if $c > 2\alpha_d d'(k-d')^2(d-d')!$, **whp** all the d'-tuples of variables are frozen tuples. For a given set of d' variable $S = \{x_{i_1}, \ldots, x_{i_{d'}}\}$, consider the sub-hypergraph $\mathcal{G}$ that contains all the hyperedges $(y_1, \ldots, y_{d-d'})$ such that the clause

$$\bar{x}_{i_1} \vee \cdots \vee \bar{x}_{i_{d'}} \vee y_1 \vee \cdots \vee y_{d-d'}$$

is in $\mathcal{F}_{k,d}^{n,p}(d')$. We see that $\mathcal{G}$ follows the distribution of the random hypergraph $\mathcal{G}(n-d', p, d-d')$. Applying Lemma A.2, we have

$$\mathbb{P}\{S \text{ is not a frozen tuple}\} \leqslant n^{-\frac{(1+\epsilon)c}{2\alpha_d(k-d')^2(d-d')!}}.$$

Since there are $\binom{n}{d'}$ possible subsets of d' variables, the claim follows from Markov's inequality.

Similar to the proof of Theorem 3, for each frozen tuple $S = \{x_{i_1}, \ldots, x_{i_{d'}}\}$, a parametric resolution derivation of size $(d-d')^{k-d'}$ exists for the clause

$$\bar{x}_{i_1} \vee \cdots \vee \bar{x}_{i_{d'}}.$$

From the collection of $\binom{n}{d'}$ clauses $\{\bar{x}_{i_1} \vee \cdots \vee \bar{x}_{i_{d'}},\ 1 \leqslant i_j \leqslant n,\ i_j \neq i_l\}$, a parametric resolution derivation of size $\binom{n}{d'-1}n$ exists for the collection of $\binom{n}{d'-1}$ clauses of size $d'-1$

$$\{\bar{x}_{i_1} \vee \cdots \vee \bar{x}_{i_{(d'-1)}},\ 1 \leqslant i_j \leqslant n,\ i_j \neq i_l\}.$$

Continuing in this way, we see that a parametric resolution derivation for all the clauses $\{\bar{x}_i,\ 1 \leqslant i \leqslant n\}$ of size $O((d-d')^{k-d'}\binom{n}{d'})$ exists, from which a size-n parametric proof of size n exists for the contradiction. This completes the proof. □

For satisfiable region, the best we currently have is that for clause probability $p(n) = o(\frac{\log n}{n^{d-1}})$, random instances can be shown to be satisfiable due to an observation similar to Lemma 2.2. In contrast, the phase transition occurs at $p(n) = \Theta(\frac{\log n}{n^{d-d'}})$. One possible approach to extending the idea of the algorithm W-SAT is to make use of 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 motivations behind the study of 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 present the first sound theoretical evidence on the effectiveness of using simple reduction rules to solve 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 the random models of weighted CNF satisfiability proposed 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 $\mathcal{F}_{k,d}^{n,p}$ 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 general model $\mathcal{F}_{k,d}^{n,p}(d')$ 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.

Acknowledgements

I would like to thank the three anonymous referees for their careful reading of the paper and for their thoughtful feedbacks. Their detailed suggestions and criticisms have helped improve the paper significantly.

Appendix A. Proof of Lemma 4.1

For the case of WEIGHTED 2-SAT, Lemma 4.1 is a statement similar to that on the minimum vertex degree is a random graph, and results in random graph literature can be applied (Theorem 3.5 in [12]).

For the case of WEIGHTED d-SAT with $d > 2$, the complication is that we require the hyperedges that a variable belongs to are pairwise vertex-disjoint. Note that Lemmas 4.4 and 4.5 deal with the same complication, but because of the term $\frac{\log^2 n}{n}$ in the bound, they are not strong enough to guarantee that **whp** all the variables are k-frozen. To achieve this, we resort to the extended Janson inequality (see, e.g., Theorem 8.1.2 in [6]), but note that this approach doesn't work for MINI-WEIGHTED d-SAT instances.

Lemma A.1 *(The Extended Janson Inequality, Theorem 8.1.2 [6]). Let $(\{0,1\}^{\Omega}, P)$ be an independent product probability space where Ω is a finite set, I be a finite index set, and A_i, $i \in I$, be a subset of Ω. For each $i \in I$, let B_i be the event*

$$\big\{\omega = (\omega_j) \in \{0,1\}^{\Omega} \colon\ \omega_j = 1\ \forall j \in A_i\big\},$$

$\mu = \sum_i \mathbb{P}\{B_i\}$, and $\Delta = \sum_{i\sim j} \mathbb{P}\{B_i \cap B_j\}$ where $i \sim j$ means that "$i \neq j$ and $A_i \cap A_j \neq \emptyset$". Assume that $\Delta \geqslant \mu$. We have

$$\mathbb{P}\left\{\bigcap_i \overline{B_i}\right\} \leqslant e^{-\frac{\mu^2}{2\Delta}}.$$

We will first establish a result on the existence of k-disjoint hyperedges in a random hypergraph which is interesting in its own right.

Remark. We thank one of the referees who noticed that our original proof, given in the context of random CNF formulas, actually applies to the more general case of random hypergraphs. After checking the literature further, we see that the result can be regarded as a generalization of some result in random graphs on the bound of the lower tail probability of small subgraph counts (see, e.g., Theorem 8.7.2 in [6] and Theorem 4.1 in [12]).

Lemma A.2. *Let $\mathcal{G}(n,p,d)$ be a random hyper graph with edge probability $p = p(n) = \frac{b\log n}{n^d}$ and let $\mathcal{M}$ be the event that there exists a set of k vertex-disjoint hyperedges in $\mathcal{G}(n,p,d)$ and $\overline{\mathcal{M}}$ the complement of $\mathcal{M}$. We have*

$$\mathbb{P}\{\overline{\mathcal{M}}\} \leqslant n^{-\frac{(1+\epsilon)b}{2k^2 d!}} \tag{A.1}$$

for some constant $\epsilon > 0$.

Proof. We prove the lemma by using the extended Janson inequality. For this purpose, let $\mathcal{P}$ be the family of the collections of k pairwise disjoint subsets of d vertices, i.e.,

$$\mathcal{P} = \big\{(Y_i,\ 1 \leqslant i \leqslant k) \colon\ Y_i \subset V,\ Y_i \cap Y_j = \emptyset,\ |Y_i| = d\big\}.$$

We also require that the vertices in each collection $(Y_i,\ 1 \leqslant i \leqslant k)$ are distinct. Here, the set of all possible hyperedges plays the role of Ω, the set $\mathcal{P}$ plays the role of I, and sets of k hyperedges play the role of A_i's in the extended Janson's inequality.

For a given $\mathcal{Y} = (Y_i,\ 1 \leqslant i \leqslant k) \in \mathcal{P}$, let $B_{\mathcal{Y}}$ be the event that the hyperedges Y_i, $1 \leqslant i \leqslant k$, are in the random hypergraph $\mathcal{G}(n,p,d)$. We have

$$\mathbb{P}\{B_{\mathcal{Y}}\} = \big(p(n)\big)^k.$$

There are $\binom{n}{d}\binom{n-d}{d}\cdots\binom{n-(k-1)d}{d}$ ways to choose k vertex-disjoint subsets of size d. Consequently, the size of $\mathcal{P}$ is

$$|\mathcal{P}| = \frac{1}{k!}\binom{n}{d}\binom{n-d}{d}\cdots\binom{n-(k-1)d}{d}$$

since there are $\frac{1}{k!}$ ways to arrange the k subsets. It follows that the expected number of sets of k disjoint hyperedges

$$\mu = \sum_{\mathcal{Y}\in\mathcal{P}} \mathbb{P}\{B_{\mathcal{Y}}\}$$

can be bounded as

$$\left(1 - \frac{(k-1)d}{n}\right)^{kd} \frac{1}{k!d!}\big(p(n)n^d\big)^k \leqslant \mu \leqslant \frac{1}{k!}\big(p(n)n^d\big)^k. \tag{A.2}$$

To apply the extended Janson inequality, we estimate the Δ in Lemma A.1,

$$\Delta = \sum_{\mathcal{Y}_1 \cap \mathcal{Y}_2 \neq \emptyset} \mathbb{P}\{B_{\mathcal{Y}_1} \cap B_{\mathcal{Y}_2}\},$$

where the sum is over all the ordered pairs $(\mathcal{Y}_1, \mathcal{Y}_2) \in \mathcal{P} \times \mathcal{P}$ such that $\mathcal{Y}_1 \cap \mathcal{Y}_2 \neq \emptyset$.

For any pair $(\mathcal{Y}_1, \mathcal{Y}_2)$ such that $|\mathcal{Y}_1 \cap \mathcal{Y}_2| = i$, we have

$$\mathbb{P}\{B_{\mathcal{Y}_1} \cap B_{\mathcal{Y}_2}\} = \big(p(n)\big)^{2k-i}. \tag{A.3}$$

The total number of ordered pairs $(\mathcal{Y}_1, \mathcal{Y}_2)$ with i overlaps, which is equal to the total number of ways to select $i + (k - i) + (k - i) = 2k - i$ pairwise disjoint subsets of k variables where i of them play the special role of shared subsets, is

$$\frac{1}{i!(k-i)!(k-i)!}\binom{n}{d}\binom{n-d}{d}\cdots\binom{n-(2k-i-1)d}{d}, \tag{A.4}$$

where the term $\frac{1}{i!(k-i)!(k-i)!}$ takes care of the different ways to arrange the subsets in different parts of a pair of subsets.

Let

$$s_i = \sum_{|\mathcal{Y}_1 \cap \mathcal{Y}_2| = i} \mathbb{P}\{B_{\mathcal{Y}_1} \cap B_{\mathcal{Y}_2}\}.$$

From Eqs. (A.3) and (A.4) with $p(n) = \frac{b \log n}{n^d}$, it follows that

$$\begin{aligned} s_i &\leqslant \frac{1}{i!(k-i)!(k-i)!}\left(\frac{b\log n}{n^d}\right)^{2k-i}\left(\frac{n^d}{d!}\right)^{2k-i} \\ &= \frac{1}{i!(k-i)!(k-i)!}\left(\frac{b\log n}{d!}\right)^{2k-i}. \end{aligned}$$

Write $t = \frac{b\log n}{d!}$ so that $s_i \leqslant \frac{1}{i!(k-i)!(k-i)!}t^{2k-i}$. We have

$$\begin{aligned} \Delta &= \sum_{i=1}^{k-1} \sum_{|\mathcal{Y}_1 \cap \mathcal{Y}_2| = i} \mathbb{P}\{B_{\mathcal{Y}_1} \cap B_{\mathcal{Y}_2}\} = \sum_{i=1}^{k-1} s_i \\ &\leqslant \sum_{i=1}^{k-1} \frac{1}{i!(k-i)!(k-i)!} t^{2k-i} \\ &= \frac{1}{(k-1)!(k-1)!} t^{2k-1}\left(1 + \frac{g_k^2}{t} + \frac{g_k^3}{t^2} + \cdots + \frac{g_k^{k-1}}{t^{k-1}}\right) \end{aligned} \tag{A.5}$$

where $g_k^i = \frac{(k-1)!(k-1)!}{i!(k-i)!(k-i)!}$ is a constant since k is a constant.

Since $\lim_{n\to\infty} t = \infty$, for any $\epsilon > 0$ there is an integer $N = N(\epsilon) > 0$ such that for any $n > N(\epsilon)$, the last term $(1 + \frac{g_k^2}{t} + \frac{g_k^2}{t^2} + \cdots + \frac{g_k^{k-1}}{t^{k-1}})$ in the above is upper bounded by $1 + \epsilon$. Therefore, we have

$$\Delta \leqslant (1+\epsilon)\frac{1}{(k-1)!(k-1)!}t^{2k-1}.$$

Therefore, $\frac{\mu^2}{2\Delta}$ is asymptotically lower bounded by

$$\frac{(1+\epsilon)b\log n}{2k^2 d!}.$$

It follows from the extended Janson's inequality that

$$\mathbb{P}\{\overline{\mathcal{M}}\} = \mathbb{P}\left\{\bigcap_{\mathcal{Y}} \overline{B}_{\mathcal{Y}}\right\} \leqslant e^{-\frac{(1+\epsilon)b}{2k^2 d!}\log n} = n^{-\frac{(1+\epsilon)b}{2k^2 d!}}.$$

This completes the proof of lemma. □

An immediate application of Lemma A.2 is the following corollary on the minimum number of vertex-disjoint hyperedges on a vertex in a random hypergraph.

Corollary A.1. *Let $\mathcal{G}(n,p,d)$ be a random hyper graph with edge probability $p = p(n) = \frac{b\log n}{n^d}$. Let δ be the minimum number of vertex-disjoint hyperedges on a vertex. We have for any $b > 2k^2(d-1)!$*

$$\lim_{n\to\infty} \mathbb{P}\{\delta < k\} = 0. \tag{A.6}$$

Note the extra term $2k^2$ in the lower bound for b, as compared to similar results for the minimum vertex degree in a random graph (Theorem 3.5 in [12]). We believe this is due to the requirement of vertex-disjointness, and do not know if it can be improved.

Proof. For a given vertex, the sub-hypergraph on $V \setminus x$ that contains all the hyperedges $\{y_1,\ldots,y_{d-1}\}$ such that $\{x, y_1,\ldots,y_{d-1}\}$ is a hyperedge in $\mathcal{G}(n,p,d)$ follows the distribution of the random hypergraph $\mathcal{G}(n-1,p,d-1)$. Applying Lemma A.2 and using Markov's inequality, we see that **whp** all vertices are incident to at least k vertex-disjoint hyperedges in $\mathcal{G}(n,p,d)$. □

We are now ready to use Lemma A.2 to prove Lemma 4.1. Let x be a fixed variable. Consider the sub-hypergraph $\mathcal{G}$ on the set of variables $X \setminus \{x\}$ that contains all the hyperedges $Y = (y_1,\ldots,y_{d-1})$ such that $\bar{x} \vee y_1 \vee \cdots \vee y_{d-1}$ is a clause appearing in the random CNF formula $\mathcal{F}_{k,d}^{n,p}$. We see that $\mathcal{G}$ is a random hypergraph $\mathcal{G}(n-1,p,d-1)$ with edge probability $p(n) = \frac{c\log n}{n^{d-1}}\frac{1}{2^d-1}$. Applying Lemma A.2 to $\mathcal{G}(n-1,p,d-1)$ with $p = \frac{b\log(n-1)}{(n-1)^{d-1}}$ where $b = \frac{c}{2^d-1}$, it follows from Lemma A.2 that

$$\begin{aligned}\mathbb{P}\{x \text{ is not } k\text{-frozen}\} &\leqslant \mathbb{P}\{\mathcal{G} \text{ does not contain } k \text{ disjoint hyperedges}\}\\ &\leqslant e^{-\frac{(1+\epsilon)c}{2k^2(2^d-1)(d-1)!}\log n},\end{aligned} \tag{A.7}$$

for any $\epsilon > 0$. If $c > 2k^2(2^d-1)(d-1)!$, take an enough small $\epsilon > 0$ such that $(1+\epsilon)c > 2k^2(2^d-1)(d-1)!$. For $n > N(\epsilon)$ and by Markov's inequality, the probability that all variables are k-frozen is at least

$$1 - ne^{-\frac{(1+\epsilon)c}{2k^2(2^d-1)(d-1)!}\log n} = 1 - o(1).$$

This proves Lemma 4.1.

Appendix B. 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 [26,39].

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 *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 $\{S_i,\ 1 \leqslant i \leqslant (d-1)\}$. 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-counting.

Let $s_i = |S_i|,\ 1 \leqslant i \leqslant d-1$. Since there are $\log n - 1$ edges in T and a clause in S_i is dedicated to i edges of T, we have $0 \leqslant s_i \leqslant \log n/i$ and

$$\sum_{i=1}^{d-1} i s_i = \log n - 1. \tag{B.1}$$

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, each set of i edges in T involves at least $i+1$ variables, and consequently there are at most $\binom{n}{d-(i+1)}(2^d-1)$ ways to select a clause for a set of i edges. Given fixed s_i, $1 \leqslant i \leqslant d-1$, let $E_T(s_1,\ldots,s_{d-1})$ be the expected number of clause sets F_T that cover the edges of T such that each clause in the group S_i is dedicated to i edges. We have

$$\begin{aligned}E_T(s_1,\ldots,s_{d-1}) &\leqslant \prod_{i=1}^{d}\binom{\log n}{i}^{s_i}\binom{n}{d-(i+1)}(2^d-1)\left(\frac{c\log n}{2^d-1}\frac{1}{n^{d-1}}\right)^{s_i}\\ &\leqslant (\log n)^{\sum_{i=1}^{d-1} i s_i} n^{\sum_{i=1}^{d-1}(d-i-1)s_i}(2^d-1)^{\sum_{i=1}^{d-1}(d-i-1)s_i}\left(\frac{c\log n}{2^d-1}\frac{1}{n^{d-1}}\right)^{\sum_{i-1}^{d-1} s_i}\\ &\leqslant (\log n)^{\log n} n^{\sum_{i=1}^{d-1}(-i s_i)} n^{f(d,c)} (\log n)^{\log n}\\ &= n^{f(d,c)}(\log n)^{2\log n} n^{-\log n+1}\end{aligned}$$

where $f(d,c)$ is a function that only depends on d and c. By Markov's inequality, the $\mathbb{P}\{\mathcal{A}\}$ can be upper bounded as

$$\begin{aligned}\mathbb{P}\{\mathcal{A}\} &\leqslant \sum_{s_1=1}^{\log n}\sum_{s_2=1}^{\log n}\cdots\sum_{s_{d-1}=1}^{\log n} E_T(s_1,\ldots,s_{d-1})\\ &\leqslant (\log n)^{d-1} n^{f(d,c)+1}(\log n)^{2\log n} n^{-\log n}.\end{aligned}$$

This proves Lemma 4.6. □

Appendix C. Proof of Lemma 2.2

We give a proof to Lemma 2.2 on the trivial random instances. First, consider the case where monotone clauses are not forbidden in $\mathcal{F}_{k,d}^{n,p}$. Let $\mathcal{G}$ be the hypergraph on the set X of n variables that contains only those edges $Y=(y_1,\ldots,y_d)$ such that the monotone clause $y_1\vee y_2\vee\cdots\vee y_d$ appears in $\mathcal{F}_{k,d}^{n,p}$. We see that $\mathcal{G}$ has the distribution of the random hypergraph $\mathcal{G}(n,p',d)$ with edge probability $p'=p(n)\frac{1}{2^d-1}$. It follows from Lemma A.2 that for any $p(n)\geqslant\frac{b\log n}{n^d}$

$$\mathbb{P}\left\{\mathcal{F}_{k,d}^{n,p} \text{ contains } k \text{ disjoint monotone clause}\right\}\geqslant 1-n^{-\frac{(1+\epsilon)b}{2k^2d!}}=1-o(1).$$

Note that $\frac{b\log n}{n^d}$ is extremely small — with such a probability, the average number of clauses in $\mathcal{F}_{k,d}^{n,p}$ is $O(\log n)$. This proves Lemma 2.2(1).

To prove Lemma 2.2(2), consider the probability p^* that a given variable does not appear as a negated literal in $\mathcal{F}_{k,d}^{n,p}$. We have

$$\begin{aligned}p^* &= \left(\big(1-p(n)\big)+p(n)\frac{2^{d-1}-1}{2^d-1}\right)^{\binom{n}{d-1}}\\ &\leqslant \left(1-\frac{2^{d-1}}{2^-1}p(n)\right)^{\frac{n^{d-1}}{(d-1)!}}.\end{aligned}$$

For $p=\frac{c\log n}{n^{d-1}}$, p^* is asymptotically greater than

$$n^{-c\frac{2^{d-1}}{2^d-1}\frac{1}{(d-1)!}}.$$

It follows that the expected number of variables that does not appear as a negated literal in $\mathcal{F}_{k,d}^{n,p}$ is

$$p^*n\geqslant n^{1-c\frac{2^{d-1}}{2^d-1}\frac{1}{(d-1)!}}$$

which goes to infinity if

$$c<\left(\frac{2^{d-1}}{2^d-1}\frac{1}{(d-1)!}\right)^{-1}.$$

Using Chebyschev's inequality, it can be shown that **whp** there are more than k variables that does not appear as a negated literal in $\mathcal{F}_{k,d}^{n,p}$ whenever $p=\frac{c\log n}{n^{d-1}}$ with small enough c or $p(n)\in o(\frac{\log n}{n^{d-1}})$.

References

[1] D. Achlioptas, H. Jia, C. Moore, Hiding satisfying assignments: Two are better than one, Journal of Artificial intelligence 24 (2005) 623–639.
[2] D. Achlioptas, Y. Peres, The random k-SAT threshold is $2^k\log 2-o(k)$, in: Proceedings of the 35th Annual Symposium on Theory of Computing (STOC'03), 2003, pp. 223–231.
[3] D. Achlioptas, F. Ricci-Tersenghi, On the solution-space geometry of random constraint satisfiability problems, in: Proceedings of the 38th Annual Symposium on Theory of Computing (STOC'06), 2006, pp. 130–139.
[4] J. Alber, N. Betzler, R. Niedermeier, Experiments on data reduction for optimal domination in networks, Annals of Operations Research 146 (2006) 105–117.
[5] N. Alon, N. Kahale, A spectral technique for coloring random 3-colorable graphs, SIAM J. Computing 26 (1997) 1733–1748.
[6] N. Alon, J. Spencer, The Probabilistic Method, Wiley, 2000.
[7] P. Beame, R. Karp, T. Pitassi, M. Saks, The efficiency of resolution and Davis–Putnam procedures, SIAM Journal on Computing 31 (4) (2002) 1048–1075.
[8] A. Becker, R. Bar-Yehuda, D. Geiger, Randomized algorithms for the loop cutset problem, Journal of Artificial Intelligence (2000) 219–234.
[9] E. Ben-Sasson, A. Wigderson, Short proofs are narrow-resolution made simple, Journal of ACM 49 (2) (2001) 149–169.
[10] C. Bessiére, E. Hebrard, B. Hnich, Z. Kiziltan, C. Quimper, T. Walsh, The parameterized complexity of global constraints, in: Proceedings of the 23th AAAI Conference on Artificial Intelligence (AAAI'08), 2008, pp. 235–240.
[11] S. Böcker, S. Briesemeister, G. Klau, Exact algorithms for cluster editing: Evaluation and experiments, in: Proceedings of the 7th International Workshop on Experiment Algorithms (WEA'08), 2008, pp. 289–302.

[12] B. Bollobas, Random Graphs, Cambridge University Press, 2001.
[13] B. Bollobas, C. Borgs, J. Chayes, J. Kim, D. Wilson, The scaling window of the 2-SAT transition, Random Structures and Algorithms 18 (3) (2001) 201–256.
[14] P. Cheeseman, B. Kanefsky, W. Taylor, Where the *really* hard problems are, in: Proceedings of the 12th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, 1991, pp. 331–337.
[15] H. Chen, A. Frieze, Coloring bipartite hypergraphs, in: Proc. of the 5th International IPCO Conference on Integer Programming and Combinatorial Optimization, 1996, pp. 345–358.
[16] J. Chen, B. Chor, M. Fellows, X. Huang, D. Juedes, I. Kanj, G. Xia, Tight lower bounds for certain parameterized NP-hard problems, Information and Computation (2005) 216–231.
[17] J. Chen, S. Lu, S. Sze, F. Zhang, Improved algorithms for path, matching, and packing problems, in: Proceedings of the 18th Annual ACM–SIAM Symposium on Discrete Algorithms, 2007, pp. 298–307.
[18] A. Coja-Oghlan, M. Krivelevich, D. Vilenchik, Why almost all satisfiable k-cnf formulas are easy, in: Proc. of the 13th International Conference on Analysis of Algorithms, 2007, pp. 89–102.
[19] S. Cook, D. Mitchell, Finding hard instances of the satisfiability problem: A survey, in: Du, Gu, Pardalos (Eds.), Satisfiability Problem: Theory and Applications, in: DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 35, American Mathematical Society, 1997.
[20] N. Creignou, H. Daudé, U. Egly, Phase transition for random quantified XOR-formulas, Journal of Artificial Intelligence Research 29 (2007) 1–18.
[21] J. Culberson, I. Gent, Frozen development in graph coloring, Theoretical Computer Science 265 (1–2) (2001) 227–264.
[22] S. Dantchev, B. Martin, S. Szeider, Parameterized proof complexity, in: Proceedings of the 48th Annual Symposium on Foundations of Computer Science (FOCS'07), IEEE Press, 2007, pp. 150–160.
[23] B. Dilkina, C. Gomes, A. Sabharwal, Tradeoffs in the complexity of backdoor detection, in: Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming (CP'07), 2007, pp. 256–270.
[24] R. Downey, M. Fellows, Parameterized Complexity, Springer, 1999.
[25] M. Fellow, C. Knauer, P. Ragde, F. Rosamond, U. Stege, D. Thilikos, S. Whitesides, Faster fixed-parameter tractable algorithms for matching and packing problems, Algorithmica 2 (2008) 167–176.
[26] A. Flaxman, A spectral technique for random satisfiable 3CNF formulas, in: Proc. of 14th ACM–SIAM Symposium on Discrete Algorithms, 2003, pp. 357–363.
[27] F. Fomin, F. Grandoni, D. Kratsch, Some new techniques in design and analysis of exact (exponential) algorithms, Bulletin of EATCS 87 (2005) (Tech. rep.).
[28] Y. Gao, Phase transitions and complexity of weighted satisfiability and other intractable parameterized problems, in: Proceedings of the 23th AAAI Conference on Artificial Intelligence (AAAI'08), 2008, pp. 265–270.
[29] Y. Gao, J. Culberson, An analysis of phase transition in NK landscapes, Journal of Artificial Intelligence Research 17 (2002) 309–332.
[30] Y. Gao, J. Culberson, Consistency and random constraint satisfaction models, Journal of Artificial Intelligence Research 28 (2007) 517–557.
[31] I. Gent, T. Walsh, Analysis of heuristics for number partitioning, Computational Intelligence 14 (3) (1998) 430–451.
[32] C. Gomes, C. Fernandez, B. Selman, C. Bessiere, Statistical regimes across constrainedness regions, Constraints 10 (4) (2005) 313–337.
[33] C. Gomes, T. Walsh, Randomness and structure, in: F. Rossi, P. van Beek, T. Walsh (Eds.), Handbook of Constraint Programming, Elsevier, 2006, pp. 639–664.
[34] G. Gottlob, F. Scarcello, M. Sideri, Fixed-parameter complexity in AI and nonmonotonic reasoning, Artificial Intelligence 138 (1–2) (2002) 55–86.
[35] G. Gottlob, S. Szeider, Fixed-parameter algorithms for artificial intelligence, constraint satisfaction, and database problems, The Computer Journal 51 (3) (2008) 303–325.
[36] J. Gramm, J. Guo, F. Hüffner, R. Niedermeier, Data reduction, exact, and heuristic algorithms for clique cover, in: Proceedings of the 8th Workshop on Algorithm Engineering and Experiments (ALENEX'06), 2006, pp. 86–94.
[37] J. Guo, R. Niedermeier, Invitation to data reduction and problem kernelization, SIGACT News 38 (1) (2007) 31–45.
[38] H. Jia, C. Moore, D. Strain, Generating hard satisfiable formulas by hiding solutions deceptively, Journal of Artificial Intelligence Research (2007) 107–118.
[39] M. Krivelevich, D. Vilenchik, Solving random satisfiable 3CNF formulas in expected polynomial time, in: Proc. of 17th ACM–SIAM Symposium on Discrete Algorithms, 2006, pp. 454–463.
[40] M. Langston, A. Perkins, A. Saxton, J. Schaffer, B. Voy, Innovative computational methods for transcriptomic data analysis: A case study in the use of FPT for practical algorithm design and implementation, The Computer Journal 51 (2008) 26–38.
[41] D. Marx, Parameterized complexity of constraint satisfaction problems, Computational Complexity 2 (2005) 153–183.
[42] C. McDiarmid, General percolation and random graphs, Adv. Appl. Prob. 13 (1981) 40–60.
[43] M. Mezard, R. Zecchina, The random k-satisfiability problem: From an analytic solution to an efficient algorithm, Phys. Rev. E 66 (2002).
[44] M. Molloy, Models and thresholds for random constraint satisfaction problems, in: Proceedings of the 34th ACM Symposium on Theory of Computing, ACM Press, 2002, pp. 209–217.
[45] R. Monasson, R. Zecchina, Statistical mechanics of the random k-sat model, Phys. Rev. E 56 (1997) 1357.
[46] R. Niedermeier, Invitation to Fixed-Parameter Algorithms, Oxford Univ. Press, 2006.
[47] N. Nishimura, P. Ragde, S. Szeider, Detecting backdoor sets with respect to Horn and binary clauses, in: Proceedings of the Seventh International Conference on Theory and Applications of Satisfiability Testing (SAT'04), 2004.
[48] S. Szeider, Backdoor sets for DLL subsolvers, Journal of Automated Reasoning 1–3 (2005) 73–88.
[49] B. Vandegriend, J. Culberson, The $G_{n,m}$ phase transition is not hard for the Hamiltonian Cycle problem, Journal of Artificial Intelligence Research 9 (1998) 219–245.
[50] K. Weihe, Covering trains by stations or the power of data reduction, in: Proceedings of the Workshop on Algorithm and Experiments (ALEX'98), 2006, pp. 86–94.
[51] K. Xu, F. Boussemart, F. Hemery, C. Lecoutre, Random constraint satisfaction: Easy generation of hard (satisfiable) instances, Artificial Intelligence 171 (8–9) (2007) 514–534.
[52] W. Zhang, A. Rangan, M. Looks, Backbone guided local search for maximum satisfiability, in: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03), 2003, pp. 1179–1184.