

TEL-AVIV UNIVERSITY  
THE RAYMOND AND BEVERLY SACKLER FACULTY OF EXACT SCIENCES  
SCHOOL OF COMPUTER SCIENCES

# A rigorous study of some satisfiability problems

BY

**Dan Vilenchik**

UNDER THE SUPERVISION OF  
PROFESSOR MICHAEL KRIVELEVICH AND PROFESSOR URI ZWICK

THESIS SUBMITTED FOR THE DEGREE OF

“DOCTOR OF PHILOSOPHY”

SUBMITTED TO THE SENATE OF TEL-AVIV UNIVERSITY  
AUGUST 2008



# Acknowledgements

First and foremost, I would like to thank my advisor, Prof. Michael Krivelevich, who has provided a never-ending stream of knowledge and motivation throughout my graduate studies. His insight into problem-solving techniques and ability to predict which research directions will ultimately be fruitful made our meetings an instrumental learning experience. It is also a pleasure to thank Prof. Uriel Feige for many invaluable discussions. I have tremendously benefited from his expertise in numerous mathematical fields. I want to thank Prof. Uri Zwick for his helpful advises.

I would also like to thank my other coauthors, for suggesting interesting problems to investigate and not leaving me alone in the endless battle for solution. More specifically, I was privileged to collaborate with Sonny Ben-Shimon, Prof. Amin Coja-Oghlan, Dr. Abraham Flaxman, Prof. Alan Frieze, Prof. Elchanan Mossel, and Prof. Benjamin Sudakov.

Finally, I am indebted to my fellow graduate students for their friendship, support, and stimulating conversations.



# Abstract

This thesis broadly concerns the field of average case analysis, focusing on one of the most fundamental problems in computer science – *satisfiability*. Remarkable phenomenon occurring for  $k$ -CNF formulas is a phase transition in satisfiability. More precisely (yet rather informally), there exists a number  $d = d(k, n)$  such that almost all  $k$ -CNF formulas on  $n$  variables with  $m$  clauses such that  $m/n < d$  are satisfiable, whereas almost all formulas for which  $m/n > d$  are not satisfiable (although some of course are). As  $k$  grows, the dominant part of  $d$  is  $2^k \ln 2$ .

For the study of *satisfiable*  $k$ -CNF formulas, the satisfiability threshold is of no a-priori meaning. Still, the below threshold regime was extensively studied (mathematical rigorous analysis and experimental study) and the above threshold (satisfiable) regime was far less explored. In this work we rigorously study the *above* threshold satisfiable regime. Our interest in the problem is twofold. From the algorithmic perspective, we are interested in the design of efficient heuristics for solving satisfiable  $k$ -CNF formulas, and their rigorous analysis under some reasonable probabilistic assumptions (over the inputs). Another line of research that we found appealing was the study of structural properties of typical  $k$ -CNF formulas (again, some distribution over the inputs is used to define what “typical” means). A particular intriguing property that we study is the structure of the solution space of satisfiable  $k$ -CNF formulas under different distributions. Previous experience suggests a possible connection between the structure of the solution space of a typical formula and the algorithmic approaches which do and do not succeed (two such examples are algorithms that were developed for 3CNF formulas with a planted solution, and the intuition that served the development of the Survey Propagation algorithm).

The highlights of this work can be briefly described as follows.

1. We rigorously describe the structure of the solution space of a typical  $k$ -CNF formula drawn from different distributions over satisfiable  $k$ -CNF formulas. One major consequence of this study is showing that almost all  $k$ -CNF formulas with  $m$  clauses over  $n$  variables have a very simple solution-space structure (for  $m \geq (1 + \lambda)2^k \ln 2$ ,  $\lambda$  goes to 0 as  $k$  grows).
2. We study several SAT-solving heuristics (some are well-known and some we developed). We rigorously analyze their average-case complexity under several natural distributions over satisfiable  $k$ -CNF formulas. In fact, we give the first rigorous analysis of a message passing algorithm (Warning Propagation) when assuming a non-trivial structure of the input.



# Contents

<b>Acknowledgements</b>	<b>iii</b>
<b>Abstract</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Average Case Analysis . . . . .	1
1.2 Different SAT distributions . . . . .	2
1.3 Related Work . . . . .	3
1.4 Our Contribution . . . . .	5
<b>2 The Geometry of the Solution Space</b>	<b>9</b>
2.1 Statement of Results . . . . .	9
2.2 Techniques and Outline (Theorem 2.2) . . . . .	10
2.3 Properties of Random $\mathcal{S}_{n,m}$ and $\mathcal{P}_{n,m}$ Instances . . . . .	11
2.3.1 Setting the Exchange Rate . . . . .	12
2.3.2 The Discrepancy Property . . . . .	12
2.3.3 The Core Variables . . . . .	13
2.3.4 Proof of Theorem 2.2 . . . . .	15
2.4 Proof of Proposition 2.10 . . . . .	16
2.4.1 The Planted Setting . . . . .	17
2.5 Proof of Lemma 2.7 . . . . .	17
2.6 Proof of Proposition 2.14 . . . . .	19
2.7 Proof of Proposition 2.16 . . . . .	21
2.7.1 Proof of Proposition 2.16 . . . . .	22
2.7.2 Proof of Lemma 2.30 . . . . .	24
2.8 Proof of Theorem 2.1 . . . . .	29
2.8.1 Relating the uniform and the planted distributions . . . . .	29
2.8.2 The Planted Setting . . . . .	30
2.8.3 Proof of Theorem 2.1 . . . . .	33
<b>3 Why almost all satisfiable <math>k</math>-CNF instances are easy</b>	<b>35</b>
3.1 The Majority Vote . . . . .	36
3.2 Proof of Theorem 3.1 . . . . .	38

<b>4</b>	<b>The Support Paradigm</b>	<b>41</b>
4.1	Our Result . . . . .	42
4.2	The Algorithm . . . . .	43
4.2.1	The Directed Walk . . . . .	43
4.3	Properties of a Random Instance from $\mathcal{P}_{n,p}$ . . . . .	44
4.4	Analysis of the Directed Walk . . . . .	45
4.5	Algorithm's Analysis – Proof of Theorem 4.1 . . . . .	48
<b>5</b>	<b>The Message Passing Paradigm: Warning Propagation</b>	<b>49</b>
5.1	Warning Propagation . . . . .	49
5.1.1	3SAT and Factor Graphs . . . . .	50
5.1.2	The Warning Propagation Algorithm . . . . .	51
5.2	Related Work . . . . .	51
5.3	Our Results . . . . .	52
5.4	An overview . . . . .	54
5.5	Properties of a Random $\mathcal{P}_{n,p}$ Instance . . . . .	56
5.5.1	Stable Variables . . . . .	57
5.5.2	Discrepancy Properties . . . . .	59
5.5.3	The Core Variables . . . . .	60
5.5.4	The Factor Graph of the Non-Core Variables . . . . .	61
5.6	Proof of Theorem 5.1 and Proposition 5.2 . . . . .	61
5.6.1	Analysis of WP on the core factor graph . . . . .	62
5.6.2	Analysis of WP on the <i>non</i> -core factor graph . . . . .	63
5.6.3	Proof of Proposition 5.29 . . . . .	70
5.7	Proof of Proposition 5.12 . . . . .	72
5.7.1	Outline of Proof of Proposition 5.13 . . . . .	77
5.8	Discussion . . . . .	78
	<b>Bibliography</b>	<b>80</b>

# Chapter 1

## Introduction

Constraint satisfaction problems play an important role in many areas of computer science, e.g. computational complexity theory [20], coding theory [31], and artificial intelligence [48], to mention just a few. The main challenge is to devise efficient algorithms for finding satisfying assignments (when such exist), or conversely to provide a certificate of unsatisfiability. One of the best known examples of a constraint satisfaction problem is  $k$ -SAT.

A  $k$ -CNF formula over the variables  $x_1, x_2, \dots, x_n$  is a conjunction of clauses  $C_1, C_2, \dots, C_m$  where each clause is a disjunction of  $k$  literals. Each literal is either a variable or its negation. A  $k$ -CNF is satisfiable if there is a boolean assignment to the variables such that every clause contains at least one literal which evaluates to true.  $k$ -SAT is the language of all satisfiable  $k$ -CNF formulas.

While 2SAT is known to be in P, 3SAT is one of the most famous NP-complete problems [20]. The plethora of worst-case NP-hardness results for many problems in computer science motivates the study of efficient heuristics that give “useful” answers for “typical” subset of the problem instances, where “useful” and “typical” are usually not well defined. A rigorous candidate for the analog of a “useful” answer is the notion of approximation. Although satisfactory approximation algorithms are known for several NP-hard problems, 3SAT is not amongst them. In fact, Håstad [32] proved that it is NP-hard to approximate MAX-3SAT (the problem of finding an assignment that satisfies as many clauses as possible) within a ratio better than  $7/8$  (which is exactly the expected number of clauses satisfied by a random assignment). On the other hand, while the problem is believed unsolvable in polynomial time in the worst case, many heuristics have been proposed and some of them seem to perform quite well “on average” (or in practice).

### 1.1 Average Case Analysis

One way of evaluating and comparing heuristics is by running them on a collection of input instances (“benchmarks”), and checking which heuristic usually gives better results. Empirical results are sometimes informative, but we also seek more rigorous measures of evaluating heuristics. When very little can be done in the worst case, comparing heuristics’ behavior on “average” instances comes to mind. Thus trying to understand the inherent hardness of the problem, many researchers analyzed structural properties of formulas drawn from different

distributions. Studying random satisfiability instances is part of a more general theory of random structures which has been the focus of extensive research in recent years, see [30] for a comprehensive survey.

One such natural distribution is the following: fix  $c, k, n > 0$  ( $c = c(k, n)$  and  $k$  may depend on  $n$ ), choose  $m = cn$  clauses uniformly at random out of  $2^k \binom{n}{k}$  possible ones. We denote this distribution by  $\mathcal{Q}_{n,m}$  (for simplicity we omit the  $k$  subscript in the notation  $\mathcal{Q}_{n,m}$  as we shall confine ourselves to the “canonical” case  $k = 3$ ). Despite its simplicity, many essential properties of this model are yet to be understood. In particular, the hardness of deciding if a random formula is satisfiable, and finding a satisfying assignment for a random formula, are both major open problems [22, 40].

Remarkable phenomena occurring in the random model  $\mathcal{Q}_{n,m}$  are **phase transitions**. Also the property of being satisfiable has a threshold behavior. More precisely, there exists a number  $d = d(k, n)$  such that a random  $k$ -CNF formula with clause-variable ratio greater than  $d$  is *whp*<sup>1</sup> not satisfiable, while one with ratio smaller than  $d$  is [29]. A simple first-moment calculation shows that  $d_k \leq 2^k \ln 2$ . The holy-grail for many researchers in the field was, and still is, pinpointing the exact value of  $d_k$ . The state of the art in that respect is the work of Achlioptas and Peres [3] asserting that for all  $k \geq 3$ ,  $d_k \geq 2^k \ln 2 - k/2 - O(1)$ . Some researchers conjecture, based on predictions using tools from statistical mechanics, that  $d_k = 2^k \ln 2 - \Theta(1)$ .

## 1.2 Different SAT distributions

The distribution  $\mathcal{Q}_{n,m}$  that we defined above was extensively studied in the below threshold regime (both rigorous study and experimental results). Comparably, much fewer works deal with the above threshold regime. One reason for this is the difficulty in defining “natural” distributions over the negligible fraction of satisfiable  $k$ -CNF’s in that regime. Even if such distributions can be defined, analyzing them and finding heuristics that typically solve them is usually a complicated task. Nevertheless, if one is interested in the study of *satisfiable*  $k$ -CNF formulas, there is no reason to confine oneself to the below-threshold regime. For satisfiable instances, the satisfiability-threshold is a-priori a meaningless point.

In our research we focus on satisfiable  $k$ -CNF formulas in the above threshold regime, and to this end  $\mathcal{Q}_{n,m}$  is not a suitable distribution. We explore three distributions over satisfiable  $k$ -CNF formulas (one of which we introduce for the first time, although analogous distributions in the context of graph optimization problems were studied in the past).

The first distribution, and arguably the most natural one, is the **uniform** distribution defined by the following easy procedure: fix  $m$  ( $m$  may depend on  $n, k$ ), choose  $m$  clauses uniformly at random out of  $2^k \binom{n}{k}$  possible ones *conditioned* on satisfiability. We denote this distribution by  $\mathcal{S}_{n,m}$ .

Another option, popular with many researchers, is the **planted** distribution. To generate a planted instance, first an assignment  $\varphi$  to the variables is fixed, then  $m$  clauses consistent with  $\varphi$  are chosen uniformly at random (out of  $(2^k - 1) \binom{n}{k}$  possible ones). Planted distributions were

<sup>1</sup>Writing *whp* we mean with probability tending to 1 as  $n$  goes to infinity.

studied in the context of other optimization problems: maximum clique, minimal bisection, and  $k$ -colorability [6, 7, 11, 25] to mention just a few.

A third distribution, which we just mention here but do not elaborate on, we defined for the first time in a joint work with Michael Krivelevich and Benjamin Sudakov [37]. We considered the following “on-line” version of the uniform distribution, inspired by the work of Erdős et al. [21] on triangle-free graphs. The model is: randomly permute all  $2^k \binom{n}{k}$  possible clauses and starting from the empty formula, go over the clauses one by one (in the random permutation’s order) including each new clause as you go along if after its addition to the formula, it remains satisfiable. The interested reader is referred to [37] for the complete details and results.

To recap what we have just defined:

- $\mathcal{S}_{n,m}$  is the uniform distribution over satisfiable  $k$ -CNF formulas with  $m$  clauses over  $n$  variables,
- $\mathcal{P}_{n,m}$  denotes the planted distribution,
- $\mathcal{Q}_{n,m}$  denotes the uniform distribution (without any conditioning).

Some readers may wonder at this point why isn’t it that  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  are the same distribution. Here is a short discussion clarifying why this need not be the case. Let  $\Delta_{n,m,\varphi}$  be the number of formulas over  $n$  variables with  $m$  clauses that are satisfied by a fixed assignment  $\varphi$ . Due to symmetry,  $\Delta_{n,m,\varphi}$  is the same for every  $\varphi$  (thus we omit the  $\varphi$  subscript). Fix some  $k$ -CNF  $F$  on  $n$  variables and  $m$  clauses with exactly  $i$  satisfying assignments. In the above notation,

$$P_r^{\text{planted}}[F] = \frac{i}{2^n} \cdot \frac{1}{\Delta_{n,m}}. \quad (1.1)$$

This is because with probability  $\frac{i}{2^n}$  the planted assignment is one of  $F$ ’s satisfying assignments, and once the assignment is fixed, every formula which is satisfied by it is chosen with the same probability – hence the  $\frac{1}{\Delta_{n,m}}$  factor. Clearly, as Equation (1.1) implies, the more assignments  $F$  has the better its chances of being sampled. This is not the case in the uniform distribution (where no such bias occurs). On the other hand, if for most instances (with  $m$  clauses) the number of satisfying assignments is roughly the same, then the two distributions are “close”. This is known to be the case for example when  $m/n = c \log n$ ,  $c$  some sufficiently large constant (when typically there is only one satisfying assignment). Also when no such concentration is known (or even known not to exist) some relation between the two distributions can be established, allowing an analysis of  $\mathcal{S}_{n,m}$  via the more accessible  $\mathcal{P}_{n,m}$ . Indeed we use the latter technique to establish some of our results.

### 1.3 Related Work

Within the framework of this general introduction we will survey main papers that studied average-case complexity of  $k$ -SAT and closely-related optimization problems. More related

work, specific and relevant to a certain result that we present, will be covered in the relevant chapter.

Almost all (exact) polynomial-time heuristics suggested so far for random instances (either SAT or graph optimization problems) were analyzed when the input is sampled according to a planted-solution distribution, or various semi-random variants thereof. As a rule of thumb, the denser the instance is (more edges in the graph case, or more clauses in CNF formulas) the algorithmic challenge becomes easier. The early works in the area go back to Koutsoupias and Papadimitriou [36] where a simple greedy algorithm was shown to solve planted  $k$ -CNF formulas *whp* when  $m = \Omega(n^2)$ . Then, Ben-Sasson et al. [10] noticed that the simple Majority Vote procedure already finds *whp* a satisfying assignment for  $\mathcal{P}_{n,m}$  with  $m = \Omega(n \log n)$ . In fact, in that regime  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  are statistically close (since typically there is only one satisfying assignment); thus [10] also answer the algorithmic question for  $\mathcal{S}_{n,m}$  (in that regime). The main challenge remained for some time to solve  $k$ -CNF formulas in the  $o(\log n)$ -density regime. Inspired by Alon and Kahale’s algorithm for  $k$ -colorability [6], Flaxman [28] presented a polynomial time algorithm, based on spectral techniques, that *whp* solves  $\mathcal{P}_{n,m}$  instances for  $m/n \geq c$ ,  $c$  some sufficiently large constant.

On the other hand, very little work was done on non-planted distributions, such as  $\mathcal{S}_{n,m}$ . In this context one can mention the work of Chen [16] which provides an *exponential* time algorithm for  $\mathcal{S}_{n,m}$  with  $m/n$  greater than some constant. Ben-Sasson et al. [10] also study  $\mathcal{S}_{n,m}$  but with  $m/n = \Omega(\log n)$ , a regime where  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  are statistically close; [10] leave as an open question to characterize  $\mathcal{S}_{n,m}$  for  $m/n = o(\log n)$ , and in particular they ask whether there exists a *polynomial* time algorithm that finds *whp* a satisfying assignment in this regime.

We now survey two variations on the “classical” average-case analysis setting. One is expected polynomial time and the other is semi-random models. Although we have established new results in both lines of research we decided, in the framework of this dissertation, not to elaborate in those directions. Thus we confine ourselves to the following short exposition to allow the reader a more complete view.

An algorithm  $A$  works in *expected polynomial time* over a distribution  $\mathcal{D}$  if it *always* produces the correct answer, and  $\sum_I t_A(I) \cdot Pr_{\mathcal{D}}[I]$  is polynomial, where  $t_A(I)$  is the running time of  $A$  on the instance  $I$ .

An expected polynomial time algorithm also performs well on average (that is, works *whp*), the converse however isn’t necessarily true. In many cases, in order for an algorithm to run in expected polynomial time, a distinction between the typical instances (which are easy, and require polynomial time) and atypical ones (which are rare, but may require super-polynomial time) is due. Moreover, loosely speaking, the amount of work put in should be inversely proportional to how atypical the instance is. To perform this dichotomy, a deeper understanding of the underlying probability space is required, as well as a more careful analysis of properties of the random instance and many a times new algorithmic ideas. Therefore, by designing algorithms that work in expected polynomial time, one might expect to obtain more robust and efficient algorithms.

In the context of the planted distribution one can mention the expected polynomial time

algorithm of Böttcher [12] for coloring planted  $k$ -colorable graphs, and our work [39] for planted 3CNF instances. Of course many other researchers studied this notion, and we mention just the latest results.

Another line of research that was pursued by some researchers, including us, is that of semi-random models. One possible weakness of random models is that they may simply not capture the space of “useful” instances. Random instances may show a very particular structure which need not reflect the “real-world” examples. To “deal” with this “problem” semirandom models were introduced. Blum and Spencer [11] studied a semi-random model in the context of planted  $k$ -colorable graphs and present a heuristic that *whp*  $k$ -colors such graphs with average degree  $n^{\alpha_k}$ ,  $\alpha_k \geq 2/5$  ( $n$  being the number of vertices). This result was improved by Feige and Kilian [24] using an SDP-based algorithm that works *whp* for average degree  $\Omega(k \ln n)$ . Finally, Coja-Oghlan [17] gave a simpler SDP-based heuristic that works *whp* for same density, and also provides a certificate for the optimality of the coloring. In two works (one with Julia Böttcher [13], and one with Michael Krivelevich [38]) we explored semi-random models for sparse instances of  $k$ -colorability (by sparse we means  $m/n = O(1)$ ). We obtained both algorithmic and hardness results.

## 1.4 Our Contribution

As we mentioned in the previous section, the average case complexity of the planted distribution is very well understood. On the other hand, the uniform distribution  $\mathcal{S}_{n,m}$  remained an enigma for many years. Ben-Sasson et al. [10] show that when the formulas are dense enough ( $m/n = \Omega(\log n)$ ), then a simple heuristic (Majority Vote) solves almost all of them. For that clause-density regime  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  are statistically close. The case where the number of clauses is linear in the number of variables remained open for many years, with only an exponential time algorithm due to Chen [16].

The main technical challenge in analyzing  $\mathcal{S}_{n,m}$  is the fact that the uniform distribution is not a product space (due to the conditioning on the contriving event of being satisfiable at that clause-density regime), clause appearances are dependent and it is unclear how to quantify this dependence. True,  $\mathcal{P}_{n,m}$  as we defined it is not a product space as well, but it is very “close” to the distribution  $\mathcal{P}_{n,p}$  in which every clause satisfied by the planted assignment is chosen independently at random with probability  $p = m / ((2^k - 1) \binom{n}{k})$ , which is already a product space. The same goes for  $\mathcal{Q}_{n,m}$ , which is “close” to a product space as well. One demonstration of this challenge is the difficulty of answering the following question: given a fixed assignment  $\psi$ , what is the probability that it satisfies a random  $F$ ? If  $F$  is drawn from  $\mathcal{Q}_{n,m}$  then the answer is simple,  $\Pr[\psi \models F] = (1 - 2^{-k})^m$ . Also if  $F$  belongs to  $\mathcal{P}_{n,m}$  then one can give an explicit closed formula for  $\Pr[\psi \models F]$ , which now also depends on the Hamming distance of  $\psi$  from the planted assignment. If  $F$  is drawn from  $\mathcal{S}_{n,m}$  then giving an explicit expression (as a function of  $m, n, k$ ) for  $\Pr[\psi \models F]$  is still an open question.

One consequence of the fact that in  $\mathcal{S}_{n,m}$  clauses are dependent is that the rich analytical methods that were developed to study random structures cannot be applied to  $\mathcal{S}_{n,m}$ , at least not immediately (as most of them assume that the underlying distribution is a product space,

for example the well known Chernoff bound).

Overcoming those obstacles, in a joint work with Amin Coja-Oghlan and Michael Krivelevich [19] we finally solved this long-standing open question. We describe a *polynomial* time algorithm that solves *whp*  $k$ -CNF formulas sampled according to  $\mathcal{S}_{n,m}$  with  $m/n \geq c$ ,  $c = c(k)$  some sufficiently large constant. Devising new ideas for analyzing  $\mathcal{S}_{n,m}$  we show that  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  actually share many structural properties. As a consequence, we can prove that a certain algorithm, designed with  $\mathcal{P}_{n,m}$  in mind, works for  $\mathcal{S}_{n,m}$  as well. In other words, by presenting new methods for analyzing heuristics on random instances, we can show that algorithmic techniques invented for the planted model extend to the more “canonical” uniform setting.

This result is described in Chapter 3. It appeared in the proceedings of Analysis of Algorithms 2007 [19], and a similar result (by the same authors) concerning  $k$ -colorable graphs appeared in the proceedings of STACS 2007 [18].

Having understood the computational complexity of the uniform and planted distributions quite deeply, we wanted to know if there was any a-priori ground to suspect that both distributions will be “easy” (in the regime we considered), and in fact that the same algorithmic ideas will be useful for both. Indeed, questions regarding the structure of the solution space of random  $k$ -CNF formulas guided the development of algorithms in similar contexts in the past (two such examples are algorithms that were developed for 3CNF formulas with a planted solution, and the intuition that served the development of the Survey Propagation algorithm). Inspired by this line of thought we proved a structural property concerning the structure of the solution space (that is, satisfying assignments) of a typical instance in  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$ . This might turn to be the key to answering the question we opened this paragraph with.

In a joint work with Uriel Feige and Abraham Flaxman [23] we showed that for all  $k \geq 20$  and  $m/n \geq (1 + 0.99^k)2^k \ln 2$ , all but a  $o(1)$ -fraction of satisfiable  $k$ -CNF formulas  $F$  with  $m$  clauses over  $n$  variables enjoy the following property: every two satisfying assignments of  $F$  are at Hamming distance at most  $O(k2^{-k}n)$  from each other (and the same is also true *whp* for a random formula from  $\mathcal{P}_{n,m}$ ). If  $m/n$  is sufficiently large (say  $m/n \geq 100 \cdot 2^k$ ), we showed in a joint work with Amin Coja-Oghlan and Michael Krivelevich [19] that *whp* not only are the satisfying assignments close, but they also agree on the assignment of most variables (such variables are said to be **frozen**). This latter result holds for all  $k \geq 3$ .

This line of research relates to one of the most surprising recent developments in satisfiability problems that comes from statistical physics. In their well-known work, Mezard, Parisi and Zecchina [14] designed the *Survey Propagation* algorithm for solving  $k$ -SAT instances. A particularly dramatic feature of this method is that it appears to remain effective in solving very large instances of random  $k$ -SAT even with densities very close to the conjectured satisfiability threshold, a regime where other algorithms (e.g., the WalkSAT method [51]) typically fail. The difficulty that *Survey Propagation* apparently overcomes lies in the complicated structure of the solution space of such formulas. That is, the *conjectured* picture, some supporting evidence of which were proved rigorously for  $k \geq 8$  [45, 4, 1], is that typically random  $k$ -CNF formulas in the near-threshold regime have an exponential number of **clusters** of satisfying assignments. While any two assignments in distinct clusters disagree on at least  $\varepsilon n$  variables, any two assignments within one cluster coincide on  $(1 - \varepsilon)n$  variables (for the scope of this work, this will be

the definition of a cluster). Furthermore, each cluster has a linear number of **frozen** variables (a variable is said to be *frozen* in some cluster if *all* satisfying assignments within that cluster assign it in the same way). The algorithmic difficulty with such a clustered solution space seems to be that most known algorithms do not “steer” into one cluster but try to find a “compromise” between the satisfying assignments in distinct clusters, which actually is impossible.

This, contrasted with our algorithmic results and the existence of a single cluster, suggests the following interesting connection between the structure of the solution space and the average-case complexity of the problem. When there exists a single cluster, whose volume might be exponential, the problem is “easy”. On the other hand, when the clustering is complicated, for example in the near threshold regime, experimental results predict that many “simple” heuristics fail, while “heavy machinery” such as *Survey Propagation* works. Stressing this last point, we prove that when the cluster structure is simple, then a simple message passing algorithm works (*Warning Propagation*), when the cluster structure is complicated, then possibly only a much more complicated message passing algorithm is known to work, *Survey Propagation* (and even this only experimentally).

This result is described in Chapter 2. Although “chronologically” this chapter should come after the chapter discussing the algorithmic ideas, we chose to reverse the order as we think this makes the presentation clearer and more cohesive. This work is based on a work which appeared in the proceedings of Analysis of Algorithms 2007 [19], and on a manuscript which was just submitted [23].

One of the key ingredients in the proof of the single-cluster phenomenon (and in particular the existence of frozen variables) is the notion of **support**. Given a CNF formula  $F$  and an assignment  $\psi$  to its variables we say that a literal  $x$  *supports* a clause  $C$  in  $F$  w.r.t.  $\psi$  if  $x$  is the *only* literal that evaluates to true in  $C$ . This notion, though not explicitly referred to, was used by Achlioptas and Ricci-Tersenghi [4] to rigorously establish the clustering phenomenon for below-threshold random  $k$ -CNF formulas. It was also used in the algorithm of Flaxman [28] for the planted distribution (for a completely different clause-density regime and distribution) – as far as we know, Flaxman was the one who actually coined the term “support” in this context. Also in experimental work, [50, 9] show that some simple variants of the well known RWalkSAT algorithm [46], which base their greedy rule on the support (although the notion of support is not referred to explicitly in any of these works), seem to be effective for solving random 3SAT formulas in the “hard” near-threshold regime. Specifically, the experimental results suggest that these algorithms may be efficient in finding satisfying assignments for random 3SAT instances in  $\mathcal{Q}_{n,m}$  with  $m/n$  as large as 4.21 (the conjectured satisfiability threshold for 3SAT is roughly 4.26). In contrast, the “original” RWalkSAT heuristic, which is not part of the “Support Paradigm”, seems to consume super-polynomial already for  $m/n = 2.65$  [47].

Observing that the notion of support is so beneficial in many different scenarios we asked ourselves whether one can design an algorithm which is based purely on that notion, and analyze it rigorously under some reasonable probabilistic assumptions. Specifically, our focus in this work is on heuristics that obey the following general template: start at some assignment to the variables, then iteratively, using some predefined (greedy) rule, try to minimize the number of unsatisfied clauses (or the distance from some satisfying assignment) until a satisfying

assignment is reached. We say that such a heuristic is part of the **Support Paradigm** if the greedy rule uses the support as its main criterion.

Indeed, we answered this question positively and gave a new algorithm in the Support Paradigm and rigorously proved its effectiveness for the planted distribution  $\mathcal{P}_{n,m}$ , with  $m/n$  sufficiently large (yet possibly constant). This result is discussed in Chapter 4. It appeared in the Journal of Satisfiability (JSAT) [52].

Another line of research that we pursued, favored by others as well ([5, 27, 15] for example) is rigorously analyzing the performance of *well known* heuristics when assuming a “reasonable” distribution over the inputs. The algorithm that we targeted is commonly known as *Warning Propagation*, and is part of a family of algorithms usually referred to as *message passing* algorithms.

Our starting point was experimental results showing that the aforementioned *Survey Propagation* (which is a message passing algorithm), seems very effective in finding satisfying assignments for random satisfiable 3CNF formulas in a regime that was observed to be hard for other heuristics [14]. Making a modest step towards providing a rigorous understanding of that phenomenon, we analyze the performance of another well-known message passing algorithm called *Warning Propagation*, which may be viewed as a primal ancestor of the *Survey Propagation*. We show that for  $k$ -CNF formulas generated under the planted distribution, running *Warning Propagation* in the standard way works *whp* when  $m/n$  is sufficiently large (yet possibly constant). We are not aware of previous *rigorous* analysis of message passing algorithms for satisfiability instances, though such analysis was performed for decoding of Low Density Parity Check (LDPC) Codes.

This work is described in Chapter 5. It appeared in the proceedings of Random 2006 [26].

## Chapter 2

# The Geometry of the Solution Space

We decided to begin the exposition with the description of the solution space of typical  $k$ -CNF formulas sampled from the uniform or the planted distributions. The reason is that in order to establish the structure of the typical solution space we go through an extensive discussion on fundamental structural properties of such formulas. This discussion is the key to understanding the algorithmic side presented in the next chapters.

### 2.1 Statement of Results

In this chapter we analyze the structure of the solution space of typical instances from the uniform and the planted distribution. We prove that when  $m/n$  is slightly above the satisfiability threshold, typically such formulas have a very simple structure of the solution space. The diameter of the set of satisfying assignments is small, and moreover, if  $m/n$  is greater than some sufficiently large constant, we are able to prove that most variables have the same assignment in all satisfying assignments (i.e. frozen). One can contrast this structure with the much richer cluster-structure in the below threshold regime proven in a series of works for  $k \geq 8$  [4, 45, 1]. Formally,

**Theorem 2.1.** ([23]) *For all  $k \geq 20$  and  $m/n \geq (1 + 0.99^k)2^k \ln 2$ , all but an  $o(1)$ -fraction of satisfiable  $k$ -CNF formulas  $F$  with  $m$  clauses over  $n$  variables enjoy the following property: every two satisfying assignments of  $F$  are at Hamming distance at most  $50k2^{-k}n$  from each other.*

If we allow  $m$  to be somewhat larger than  $2^k \ln 2$ , say  $m/n \geq 100 \cdot 2^k$ , then we can already prove a more detailed characterization of the typical solution space.

**Theorem 2.2.** ([19]) *For all  $k \geq 3$  and  $m/n \geq 100 \cdot 2^k$ , all but a  $o(1)$ -fraction of satisfiable  $k$ -CNF formulas  $F$  with  $m$  clauses over  $n$  variables enjoy the following property:*

1. All but  $e^{-\Theta(m/n)}n$  variables are frozen.
2. The formula induced by the non-frozen variables decomposes into connected components of at most logarithmic size.
3. Letting  $\beta(F)$  be the number of satisfying assignments of  $F$ , we have  $\frac{1}{n} \log \beta(F) = e^{-\Theta(m/n)}$ .

**Remark 2.3.** The assertions in both theorems are true also for instances from  $\mathcal{P}_{n,m}$  for the same  $m/n$  values. In fact, since our main technique to prove the theorems is first analyzing the planted distribution, one result of our discussion will be a proof of Theorems 2.1 and 2.2 in the planted setting (the phrase “all but  $o(1)$ -fraction of satisfiable  $k$ -CNf formulas” should be replaced with “*whp*” since  $\mathcal{P}_{n,m}$  is not the uniform distribution over satisfiable  $k$ -CNF formulas with  $m$  clauses).

We begin with the proof of Theorem 2.2. The proof of Theorem 2.1 uses somewhat different techniques and is given (in a self-contained manner) in the last section of the chapter – Section 2.8. Our algorithmic results (in the next chapters) concern only the setting of Theorem 2.2, that is  $m/n$  is sufficiently large. In fact for  $m/n = O(1)$  but not necessarily large, it is still an open problem to describe an algorithm that works on average and rigorously prove its efficiency.

## 2.2 Techniques and Outline (Theorem 2.2)

To prove Theorem 2.2 (and also 2.1) we consider the **uniform distribution** over satisfiable  $k$ -CNF formulas with  $m$  clauses over  $n$  variables, and study the structure of the solution space of a random instance in that distribution.

The major technical challenge that we face in this present work is the fact that the uniform distribution  $\mathcal{S}_{n,m}$  is not a product space (due to the conditioning on the contriving event of being satisfiable at that clause-density regime), clause appearances are dependent, and it is unclear how to quantify this dependence. To derive both our theorems we first analyze the **planted distribution**, which we denote by  $\mathcal{P}_{n,m}$ . To generate a formula according to  $\mathcal{P}_{n,m}$ , fix an assignment uniformly at random, then include  $m$  clauses uniformly at random out of  $(2^k - 1) \binom{n}{k}$  clauses that are consistent with the “planted” assignment. When working with  $\mathcal{P}_{n,m}$ , the dependency between the clauses is simple and calculation is much easier. We then relate the planted model and the uniform model to obtain the desired result.

To prove Theorem 2.2 we study some structural properties of a typical formula in  $\mathcal{S}_{n,m}$  (working through  $\mathcal{P}_{n,m}$ ). We are able to characterize such formulas in a manner which reveals the geometrical structure of the solution space. We describe a subset of the vertices, referred to as the *core* vertices, which plays a crucial role in understanding the structure of the solution space of a typical  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  formulas, and the algorithmic approach to solve such formulas. The notion of core relies on another basic notion which is the notion of support.

**Definition 2.4.** (*support*) Given a  $k$ -CNF formula  $F$  and some assignment  $\psi$  to the variables, we say that a literal  $x$  supports a clause  $C$  (in which it appears) w.r.t.  $\psi$  if  $x$  is the only literal that evaluates to true in  $C$  under  $\psi$ .

To get intuition, first consider the distribution  $\mathcal{P}_{n,m}$ , and the case  $k = 3$ . Every variable  $x$  is expected to support  $m \cdot \frac{3}{n} \cdot \frac{1}{7}$  clauses w.r.t. to the planted assignment ( $x$  appears in a clause with probability  $3/n$ , and supports it with probability  $1/7$ ). Suppose indeed that this is the case. Given a formula  $F$  we define the graph induced by  $F$  to be  $G_F = (V, E)$  where  $V$  is the set of variables of  $F$  and  $(x, y) \in E$  if there exists  $C \in F$  that contains both  $x$  and  $y$  (regardless of polarity).

To complete the discussion we need two extra facts.

**Fact 2.5.** Assume  $F$  is drawn from  $\mathcal{P}_{n,m}$ ,  $m/n \geq C_0$ ,  $C_0$  a sufficiently large constant. Then *whp* every subgraph of  $G_F$  containing at most  $n/4000$  vertices has average degree at most  $m/(2n)$ .

**Fact 2.6.** Assume  $F$  is drawn from  $\mathcal{P}_{n,m}$ ,  $m/n \geq C_0$ ,  $C_0$  a sufficiently large constant. Then *whp* there exists *no* two satisfying assignments of  $F$  at distances at least  $n/4000$  from each other.

Both facts are proven in the sequel (perhaps stated somewhat differently). Now suppose that these two facts are indeed true for  $F$  (which is typically the case), and further assume that every variable has the expected support in  $F$  with respect to the planted assignment (which is typically *not* the case when  $m/n$  is constant). Then we claim that  $F$  has only one satisfying assignment. If not, then let  $\psi$  be another satisfying assignment of  $F$ , not equal to the planted one  $\varphi$ . Let  $U$  be the set of variables that are assigned differently in  $\varphi$  and  $\psi$ . Every  $x \in U$ , say  $\psi(x) = F$ , must have at least  $m/n$  neighbors in  $G_F[U]$ :  $x$  supports  $3m/(7n)$  clauses w.r.t.  $\varphi$ , however under  $\psi$   $x$  is false, therefore every such clause, since it is satisfied by  $\psi$  must contain at least one more  $y \in U$ . Thus contributing an edge to  $G_F[U]$ . Thus the average degree in  $G_F[U]$  is at least  $2 \cdot 3m/7n = 6m/(7n) > m/(2n)$ . However,  $|U| \leq n/4000$  due to Fact 2.6, contradicting Fact 2.5.

In what we just described all variables in  $F$  are *frozen*. When  $m/n \geq c \log n$ ,  $c$  a sufficiently large constant, then *whp* every variable in  $F$  supports roughly  $3m/7n$  clauses, and combined with the two facts, one derives that typically such formulas in  $\mathcal{P}_{n,m}$  are uniquely satisfiable. However, when  $m/n = O(1)$  this is *whp* not the case. In particular, *whp*  $e^{-\Theta(m/n)}n$  variables will not appear at all in  $F$ . Nevertheless, in the case  $m/n = O(1)$  there exists a large subformula of  $F$  showing a very similar behavior to the aforementioned one, both in the planted and the uniform setting. The set of variables inducing this formula is called a *core*. A similar notion of core, though in a different context, was introduced by Alon and Kahale [6], and then adjusted to the SAT setting in [28].

Our techniques should be contrasted with the techniques used to analyze the solution space of near-threshold instances. In this context one can mention the work in [1, 2, 4, 45], where the structure of the solution space was analyzed directly (mainly using second moment calculations). This is possible due to the fair simpleness of the underlying probabilistic model (clauses are chosen uniformly at random, in  $\mathcal{Q}_{n,m}$ ). In our setting,  $\mathcal{S}_{n,m}$ , the clauses are far from being independent of each other, and therefore trying to characterize directly the relations between the different satisfying assignments may lead to a dead-end. The idea of relating the planted and uniform model was used in [1, 4, 44] for the below-threshold regime, and also in [16] but in a different context.

In the next sections we make this discussion formal and exact.

### 2.3 Properties of Random $\mathcal{S}_{n,m}$ and $\mathcal{P}_{n,m}$ Instances

In this section we establish the structure of a typical formula in  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  ( $m/n$  sufficiently large, that is the setting of Theorem 2.2). A direct consequence of the discussion in this section is a proof of Theorem 2.2. For the sake of clear and simple presentation we consider the case  $k = 3$ . The extension to general  $k$  is immediate. In this section we only state the propositions

themselves and defer the proofs to the following sections (unless the proof is simple, and then we give it inline).

Here and throughout we think of  $m$  as  $O(n \log n)$ . Otherwise, typically a formula in  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  with  $m \geq cn \log n$ ,  $c$  some sufficiently large constant, has only one satisfying assignment (see [10] for example). A rather simple probabilistic argument shows that the Majority Vote, discussed ahead, will reconstruct *whp* the satisfying assignment in that regime. The interesting case remains  $m = O(n \log n)$ .

As we mentioned above, the clauses in  $\mathcal{S}_{n,m}$  are not chosen independently (due to the conditioning). To analyze  $\mathcal{S}_{n,m}$  we use two main techniques.  $\mathcal{P}_{n,m}$  is already very well understood, and in particular the probability of many properties that interest us can be easily estimated for  $\mathcal{P}_{n,m}$  using standard probabilistic calculations (as these clauses are independent). It then remains to find a reasonable “exchange rate” between  $\mathcal{P}_{n,m}$  and  $\mathcal{Q}_{n,m}$ . We use this approach to estimate the probability of “complicated” properties, which hold with extremely high probability in  $\mathcal{P}_{n,m}$ . The other method is directly analyzing  $\mathcal{Q}_{n,m}$ , crucially overcoming clause-dependency issues. This method tends to be more complicated than the first one, and involves intricate counting arguments.

### 2.3.1 Setting the Exchange Rate

Let  $\mathcal{A}$  be some property of CNF formulas (it would be convenient for the reader to think of  $\mathcal{A}$  as a “bad” property). We start by determining the exchange rate for  $Pr[\mathcal{A}]$  when moving from the planted distribution to the uniform one. For a property  $\mathcal{A}$  we use  $Pr^{uniform}[\mathcal{A}]$  to denote the probability of  $\mathcal{A}$  occurring under  $\mathcal{S}_{n,m}$ , and  $Pr^{planted}[\mathcal{A}]$  for  $\mathcal{P}_{n,m}$ . The following lemma asserts the exchange rate  $\mathcal{P}_{n,m} \rightarrow \mathcal{S}_{n,m}$ .

**Lemma 2.7.** ( $\mathcal{P}_{n,m} \rightarrow \mathcal{S}_{n,m}$ ) *Let  $\mathcal{A}$  be some property of 3CNF formulas, then*

$$Pr^{uniform}[\mathcal{A}] \leq e^{ne^{-m/(3n)}} \cdot Pr^{planted}[\mathcal{A}].$$

The lemma is proven in Section 2.5.

**Remark 2.8.** The exchange rate between the planted distribution and the uniform is exponential in  $n$ . Thus Lemma 2.7 is useful whenever the “bad” event  $\mathcal{A}$  happens with exponentially small probability in  $\mathcal{P}_{n,m}$ .

### 2.3.2 The Discrepancy Property

A well known result in the theory of random graphs is that a random graph *whp* will not contain a small yet unexpectedly dense subgraph. This is also the case for  $\mathcal{Q}_{n,m}$  (when considering the graph induced by the formula). This property holds only with probability  $1 - 1/poly(n)$  (for example, with probability  $1/poly(n)$  a fixed clique on a constant number of vertices will appear). Thus the exchange-rate technique is of no use in this case (as the exchange rate factor is exponential in  $n$ ).

**Definition 2.9.** *A  $k$ -CNF  $F$  on  $n$  variables and  $m$  clauses is said to be proportional if for every subset  $U$  of size  $|U| \leq n/4000$ , there are at most  $|U| \cdot \frac{m}{100n}$  clauses in  $F$  that contain two variables from  $U$ .*

To see how Definition 2.9 corresponds to the random graph context, consider the graph induced by the formula  $F$  (the vertices are the variables, and two variables share an edge if there exists some clause containing them both) and observe that every clause that contains at least two variables from  $U$  contributes an edge to the subgraph induced by  $U$ . Thus if we have many such clauses, this subgraph will be prohibitively dense. Since  $F$  is random so is its induced graph (random graph in the sense that its source,  $F$ , is random), and therefore the latter will typically not occur (as we shall prove).

Overcoming the clause-dependency issue, using an intricate counting argument, we directly analyze  $\mathcal{S}_{n,m}$  to prove:

**Proposition 2.10.** *Assume  $F$  is drawn from  $\mathcal{S}_{n,m}$  with  $m \geq C_0 n$ ,  $C_0$  a sufficiently large constant. Then whp  $F$  is proportional. Further, the same is true if  $F$  is distributed according to  $\mathcal{P}_{n,m}$ .*

The full proof is given in Section 2.4 and is an example of the direct-analysis approach (for  $\mathcal{S}_{n,m}$ ). Another example of that approach, though more complicated, is the proof of Proposition 2.16.

### 2.3.3 The Core Variables

In this section we rigorously define the notion of core variables. Recall that a variable is said to be frozen in  $F$  if in every satisfying assignment it takes the same assignment. As we mentioned above, the notion of core captures this phenomenon. In addition, a core typically contains all but a small (though possibly constant) fraction of the variables. This implies that a large fraction of the variables is frozen, which forces the simple structure given by Theorem 2.2.

**Definition 2.11.** (*core*) *A set of variables  $\mathcal{H}$  is called a **core** of  $F$  w.r.t. a satisfying assignment  $\psi$  if the following two properties hold:*

- *Every variable  $x \in \mathcal{H}$  supports at least  $m/(6n)$  clauses in  $F[\mathcal{H}]$  w.r.t.  $\psi$  ( $F[\mathcal{H}]$  being the subformula containing the clauses where all three variables belong to  $\mathcal{H}$ ).*
- *$x$  appears in at most  $m/(10n)$  clauses in  $F \setminus F[\mathcal{H}]$ .*

**Remark 2.12.** The proof of Theorem 2.2 (structure of the solution space) uses only the first property in Definition 2.11. However, since the notion of core is also used for the algorithmic perspective, the second property is needed for the analysis of the algorithm.

**Remark 2.13.** The choice of  $m/(6n)$  corresponds to slightly less than the expected support of a variable w.r.t. the planted assignment (which is roughly  $3m/(7n)$ ), had the underlying probability space been  $\mathcal{P}_{n,m}$ .

We proceed by asserting some relevant properties that such a core typically possesses.

**Proposition 2.14.** *Assume  $F$  is drawn from  $\mathcal{S}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant. Then whp there exists a satisfying assignment  $\varphi$  of  $F$  w.r.t. which there exists a core  $\mathcal{H}$  and  $|\mathcal{H}| \geq (1 - e^{-\Theta(m/n)})n$ . Further, the same is true if  $F$  is distributed according to  $\mathcal{P}_{n,m}$ , and in that case  $\varphi$  is the planted assignment.*

The proof of Proposition 2.14 uses the exchange-rate technique. It is given in Section 2.6. The next proposition characterizes the structure of the formula induced by the non-core variables. The connected components of a formula  $F$  are the sub-formulas  $F[C_1], \dots, F[C_k]$ , where  $C_1, C_2, \dots, C_k$  are the the connected components in the graph induced by  $F$ . Given a core  $\mathcal{H}$  of  $F$  w.r.t. a satisfying assignment  $\varphi$ , we denote by  $F_{out}^\varphi(\mathcal{H})$  the subformula of  $F$  which is the outcome of the following procedure: set the variables  $\mathcal{H}$  in  $F$  according to  $\varphi$  and simplify  $F$ .

**Proposition 2.15.** *Assume  $F$  is drawn from  $\mathcal{P}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant. Let  $\mathcal{H}$  be the core promised in Proposition 2.14. The largest connected component in  $F_{out}^\varphi(\mathcal{H})$  is of size  $O(\log n)$  with probability  $n^{-\Omega(m/n)}$ .*

Observe that we prove that the connected components' size property holds in the planted setting only with probability  $1 - 1/\text{poly}(n)$  (when  $m/n$  is constant), which is not strong enough to be used in the exchange-rate lemma. Therefore to prove the same property in the uniform case we have to take the direct-analysis approach, similar to Proposition 2.5. As it will turn out, the same proof that works for  $\mathcal{S}_{n,m}$  also works for  $\mathcal{P}_{n,m}$  (as is the case in proposition 2.5 – see Section 2.4).

**Proposition 2.16.** *Assume  $F$  is drawn from  $\mathcal{S}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant. Let  $\mathcal{H}$  be the core promised in Proposition 2.14. The largest connected component in  $F_{out}^\varphi(\mathcal{H})$  is whp of size  $O(\log n)$ .*

Lastly, we establish the “frozenness” property of the core variables. Before doing so we need another basic notion.

**Definition 2.17.** *We say that a 3CNF formula  $F$  is concentrated w.r.t.  $\varphi$  if there is no satisfying assignment of  $F$  at Hamming distance greater than  $n/4000$  from  $\varphi$ .*

**Proposition 2.18.** *Assume  $F$  is drawn from  $\mathcal{S}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant, and let  $\varphi$  be the satisfying assignment w.r.t. which the core was defined.  $F$  is whp concentrated w.r.t.  $\varphi$ .*

To prove this proposition, we first prove that this property holds in  $\mathcal{P}_{n,m}$  with extremely high probability, and then use Lemma 2.7 to complete the proof.

**Proposition 2.19.** *Assume  $F$  is drawn from  $\mathcal{P}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant, and let  $\varphi$  be its planted assignment. With probability at least  $1 - 2^{-n}$ , every assignment  $\psi$  at distance at least  $n/4000$  from  $\varphi$  does not satisfy  $F$ .*

**Proof.** For this proof, it will be easier to first consider a slightly different version of  $\mathcal{P}_{n,m}$ . Instead of choosing  $m$  clauses uniformly at random, choose  $m$  clauses with repetitions. Observe that conditioned on the event that no clause appears twice, the two distributions are the same. The probability of some clause being chosen twice is at most

$$\binom{m}{2} \cdot n^{-3} \leq m^2 \cdot n^{-3}.$$

The setting we explore is  $m/n = O(\log n)$  (otherwise, it was already established that there is *whp* only one satisfying assignment in both  $\mathcal{P}_{n,m}$  and  $\mathcal{S}_{n,m}$  [5]). Therefore the latter probability is  $o(1)$ . Now by a simple averaging argument, every “bad” event  $A$  that occurs with probability  $q$  in  $\mathcal{P}_{n,m}$  with repetitions, occurs with probability at most  $2q$  in  $\mathcal{S}_{n,m}$ .

Let  $A_r$  be the event that an assignment at distance  $r$  from  $\varphi$  satisfies a clause which is also satisfied by  $\varphi$ . Straightforward counting arguments show that

$$\begin{aligned} Pr[A_r] &= \underbrace{1 \cdot \frac{\binom{n-r}{3}}{\binom{n}{3}}}_{\text{clause where both assignments agree on all}} + \underbrace{\frac{6}{7} \cdot \left(1 - \frac{\binom{n-r}{3}}{\binom{n}{3}}\right)}_{\text{clause in which disagree on at least one variable}} = \\ &= \frac{6}{7} + \frac{1}{7} \left( \frac{(n-r)(n-r-1)(n-r-2)}{n(n-1)(n-2)} \right) = \frac{6}{7} + \frac{1}{7} \left(1 - \frac{r}{n}\right)^3 + o(1) \leq 1 - \frac{r}{3n} \end{aligned}$$

$F$  contains  $m$  clauses, thus the probability that an assignment  $\psi$  at distance  $r$  also satisfies  $F$  is at most (observing that the events of satisfying a clause are independent in  $\mathcal{P}_{n,m}$  with repetitions)  $(Pr[A_r])^m = \left(1 - \frac{r}{3n}\right)^m \leq e^{-mr/(3n)}$ . Setting  $r_0 = n/4000$ , the probability that a fixed assignment at distance  $n/4000$  satisfies  $F$  is at most

$$e^{-mn/(12,000n)} = e^{-n \cdot m/(12,000n)} \leq 5^{-n}.$$

The last inequality comes from the fact that  $m/(12,000n) \gg 1$  for  $m/n$  sufficiently large. Observe that  $Pr[A_r]$  decreases as  $r$  grows, and therefore taking the union bound over all assignments at distance  $n/4000$  or more from  $\varphi$  (at most  $2^n$  such assignments) we obtain that with the required probability none satisfy  $F$ . ■

**Proposition 2.20.** *Let  $F$  be a  $k$ -CNF formula with some core  $\mathcal{H}$  w.r.t.  $\varphi$ . If  $F$  is concentrated (Definition 2.17) and proportional (Definition 2.9) then all variables of  $\mathcal{H}$  receive the same assignment in all satisfying assignments of  $F$ .*

**Proof.** By contradiction let  $\psi$  be a “bad” satisfying assignment of  $F$  – that is,  $\psi$  disagrees with  $\varphi$  on the assignment of at least one core variable. Let  $x$  be some variable on which they disagree. Now consider all the clauses that  $x$  supports w.r.t.  $\varphi$  where all variables belong to  $\mathcal{H}$ . It must be that every such clause contains another *core* variable on which  $\psi$  and  $\varphi$  disagree (since every such clause is satisfied by  $\psi$ , and the literal of  $x$  is false in  $\psi$ ). Put differently, let  $U$  be the set of core variables on which  $\psi$  and  $\varphi$  disagree. By the discussion above, there are  $|U| \cdot \frac{m}{6n}$  clauses each containing two variables from  $U$  (no clause was counted twice since the supporter of a clause is unique by definition). By the concentration property  $|U| \leq n/4000$ , this however contradicts the proportionality of  $F$ . ■

### 2.3.4 Proof of Theorem 2.2

Theorem 2.2 is an immediate corollary of Propositions 2.14, 2.16 and 2.20. Propositions 2.14 and 2.20 imply that all but a  $e^{-\Theta(m/n)}n$  of the variables are frozen. Therefore, there are at most  $2e^{-\Theta(m/n)}n$  possible ways to set the assignment of the remaining variables. Furthermore, every two satisfying assignments of  $F$  can differ on the assignment of at most  $e^{-\Theta(m/n)}n$  variables

(the non-core variables). Proposition 2.16 completes the proof with the characterization of the formula induced by the non-frozen variables (which are a subset of the non-core ones).

We shall now fill in the technical details to complete the discussion in this section.

## 2.4 Proof of Proposition 2.10

Take  $F$  distributed according to  $\mathcal{S}_{n,m}$ . Let  $V$  be the set of  $n$  variables, and let  $U$  be some fixed subset of  $V$ ,  $|U| = u$ . Let  $H$  be a fixed formula over  $V$  with exactly  $\frac{um}{100n}$  clauses s.t. each clause contains at least two variables from  $U$ . A formula  $F$  is said to be  $H$ -poor if it contains  $H$  as a sub-formula (‘‘poor’’ in the sense of an ‘‘unfortunate’’ event, and not the event of lacking something).

Furthermore, let  $\mathcal{P}_H$  signify the set of all  $H$ -poor satisfiable formulas with exactly  $m$  clauses, and  $\mathcal{A}$  the set of all satisfiable formulas with exactly  $m$  edges. Our first objective is to establish the following.

**Lemma 2.21.**  $|\mathcal{P}_H| \leq (em/n^3)^{um/(100n)} |\mathcal{A}|$ .

This immediately implies that the probability of an  $H$ -poor formula in  $\mathcal{S}_{n,m}$  is at most  $(em/n^3)^{um/(100n)}$ . Next take the union bound over all possible sub-formulas  $H$  (s.t.  $|U| \leq n/4000$  – as required by Proposition 2.10) to show that *whp* none is contained in a random  $\mathcal{S}_{n,m}$  formula.

To prove Lemma 2.21 we shall set up an auxiliary bipartite graph  $\mathcal{G}$  with vertex set  $V(\mathcal{G}) = \mathcal{P}_H \cup \mathcal{A}$ . This graph will have the property that the average degree of a vertex in  $\mathcal{P}_H$  is  $\Delta$ , while that of a vertex in  $\mathcal{A}$  is  $\Delta'$ , where in addition  $\Delta'/\Delta \leq (em/n^3)^{um/(100n)}$ . Since  $\Delta \#\mathcal{P}_H = \Delta' \#\mathcal{A}$ , by double counting, we thus obtain Lemma 2.21. We describe a nondeterministic procedure  $\mathbf{P}$  that receives a formula  $F \in \mathcal{P}_H$  and produces a new formula  $F' \in \mathcal{A}$ . In our auxiliary graph  $\mathcal{G}$ , we connect a right-side node  $F$  with a left-side one  $F'$ , if  $F'$  can be obtained from  $F$  by applying  $\mathbf{P}$  to  $F$ .  $\mathbf{P}$  is the following procedure:

given a  $H$ -poor formula  $F$  do:

- Choose a set  $\mathcal{C}$  of  $um/(100n)$  fresh clauses (that are not yet in  $F$ )
- Obtain  $F'$  from  $F$  by removing all  $um/(100n)$  clauses of  $H$  and adding  $\mathcal{C}$ .
- Output  $F'$  if it is satisfiable.

Therefore,

$$\Delta \geq \binom{n^3}{um/(100n)}.$$

This is because we have to choose  $um/(100n)$  clauses out of at least  $7\binom{n}{3} - m \geq n^3$  possible ones (since  $F$  was satisfiable to begin with, there are at least  $7\binom{n}{3}$  clauses that are satisfied by the assignment that satisfies  $F$ , and we can assume that  $m = O(n \log n)$ ). Conversely, consider the following nondeterministic procedure to recover a formula  $F$  from  $F'$ . Out of  $m$  possible clauses in  $F'$ , choose  $um/(100n)$ . Take them out, and reinstall the original clauses of  $H$ . Therefore,

$$\Delta' \leq \binom{m}{um/(100n)}.$$

Using standard bounds on the binomial coefficients, the required bound on  $\Delta'/\Delta$  is obtained and Lemma 2.21 follows.

We are now ready to bound the probability that a random formula  $F$  in  $\mathcal{S}_{n,m}$  violates the condition of Proposition 2.10. Using the union bound this probability is at most

$$\begin{aligned} \sum_{u=1}^{n/4000} \binom{n}{u} \binom{8n \binom{u}{2}}{um/(100n)} \cdot \left(\frac{em}{n^3}\right)^{um/(100n)} &\leq \sum_{u=1}^{n/4000} \left(\frac{en}{u}\right)^u \left(\frac{1200un^2}{m}\right)^{um/(100n)} \left(\frac{em}{n^3}\right)^{um/(100n)} \\ &\leq \sum_{u=1}^{n/4000} \left(\frac{en}{u}\right)^u \left(\frac{3600u}{n}\right)^{uC_0/100} \leq \sum_{u=1}^{n/4000} \left(\frac{en}{u} \cdot \frac{3600u}{n} \cdot \left(\frac{3600u}{n}\right)^{C_0/100-1}\right)^u \\ &\leq \sum_{u=1}^{n/4000} \left(11^4 \cdot \left(\frac{3600u}{n}\right)^{C_0/200}\right)^u = o(1) \end{aligned}$$

The last equality is due to  $(u/n) \leq 1/4000$ , so the last sum decreases faster than a geometric series with quotient and first element equal  $1/\text{poly}(n)$ , and therefore the whole sum is  $o(1)$ .

### 2.4.1 The Planted Setting

The proof of Proposition 2.10 does not imply (at least not directly) the same assertion for the planted setting. This is because the planted distribution is not uniform over all  $k$ -CNF formulas with  $m$  clauses over  $n$  variables, and hence the method we used with the auxiliary bi-partite graph will not work immediately. Nevertheless, let us consider all formulas in  $\mathcal{P}_{n,m}$  with exactly  $i$  satisfying assignments – let  $T_i$  be that set. It is easy to see that  $\mathcal{P}_{n,m}$  is uniform over  $T_i$  (recall Equation (1.1)). Therefore, we can repeat the argument above for  $T_i$  (that is, instead of  $\mathcal{P}_H$  we put  $\mathcal{P}_H \cap T_i$ ) to obtain that Proposition 2.5 holds *whp* for  $T_i$  for every  $i$ . Finally, let  $A_i$  be the “bad” event that we sampled a graph in  $\mathcal{P}_H$  given that the graph has exactly  $i$  satisfying assignments (that is, the graph belongs to  $T_i$ ), and  $B_i$  the event that we sampled a graph in  $T_i$ . Observe that in the last discussion we asserted that  $\Pr[A_i|B_i] = o(1)$ . Finally, the probability for some  $k$ -CNF  $F \in \mathcal{P}_H$  under the planted distribution is given by

$$\Pr[\mathcal{P}_H] = \sum_{i=1}^{2^n} \Pr^{\text{planted}}[A_i|B_i] \cdot \Pr[B_i] = \sum_i o(1) \cdot \Pr[B_i] = o(1) \sum_i \Pr[B_i] = o(1) \cdot 1 = o(1).$$

## 2.5 Proof of Lemma 2.7

Basically, what we show is that  $\Pr^{\text{uniform}}[\mathcal{A}]$  is at most the expected number of satisfying assignments that a formula in  $\mathcal{S}_{n,m}$  has times  $\Pr^{\text{planted}}[\mathcal{A}]$ . To make this bound useful we need to bound this expectation (which is possibly an interesting result on its own right).

More formally, let  $C_1(n)$  be the expected number of satisfying assignments of a formula in  $\mathcal{S}_{n,m}$ , and  $C_2(n)$  is defined similarly for  $\mathcal{P}_{n,m}$ . Let  $t_i$  be the number of formulas on  $n$  variables and  $m$  clauses which have exactly  $i$  satisfying assignments. Let  $p_i$  be the probability that a formula with exactly  $i$  satisfying assignments is sampled from  $\mathcal{S}_{n,m}$ , and let  $q_i$  be defined similarly for  $\mathcal{P}_{n,m}$ . For a satisfying assignment  $\varphi$ , let  $\Delta_{n,m,\varphi}$  be the number of formulas on  $n$

variables with  $m$  clauses that are satisfied by  $\varphi$ . Observe that due to symmetry  $\Delta_{n,m,\varphi}$  is the same for every  $\varphi$  – thus we omit the  $\varphi$  subscript. In the above notation

$$p_i = \frac{t_i}{\sum_{j=1}^{2^n} t_j},$$

$$q_i = t_i \cdot \frac{i}{2^n} \cdot \frac{1}{\Delta_{n,m}}.$$

Further observe that

$$2^n \cdot \Delta_{n,m} = \sum_{i=1}^{2^n} i \cdot t_i.$$

This is because every formula with  $j$  satisfying assignments is counted exactly  $j$  times in the product  $2^n \cdot \Delta_{n,m}$ . Lastly,

$$C_1(n) = \sum_{i=1}^{2^n} i \cdot p_i = \frac{\sum_{i=1}^{2^n} i \cdot t_i}{\sum_{i=1}^{2^n} t_i},$$

$$C_2(n) = \sum_{i=1}^{2^n} i \cdot q_i = \frac{\sum_{i=1}^{2^n} i^2 \cdot t_i}{2^n \cdot \Delta_{n,m}} = \frac{\sum_{i=1}^{2^n} i^2 \cdot t_i}{\sum_{i=1}^{2^n} i \cdot t_i}.$$

**Lemma 2.22.** *Let  $\mathcal{A}$  be some property of 3CNF formulas, then*

$$P_{r^{uniform}}[\mathcal{A}] \leq C_1(n) \cdot P_{r^{planted}}[\mathcal{A}].$$

**Proof.** First we obtain the following bound.

$$\frac{P_{r^{uniform}}[\mathcal{A}]}{P_{r^{planted}}[\mathcal{A}]} \leq \max_i \frac{p_i}{q_i}.$$

This follows from the following discussion. Let  $T_{\mathcal{A}}$  be the set of satisfiable formulas for which property  $\mathcal{A}$  holds.

$$P_{r^{uniform}}[\mathcal{A}] = \sum_{F \in T_{\mathcal{A}}} P_{r^{uniform}}[F], \quad P_{r^{planted}}[\mathcal{A}] = \sum_{F \in T_{\mathcal{A}}} P_{r^{planted}}[F].$$

Now let  $b = \max_i \frac{p_i}{q_i}$ . For every satisfiable 3CNF  $F$  it holds that  $P_{r^{uniform}}[F] \leq b \cdot P_{r^{planted}}[F]$ . Therefore,

$$\sum_{F \in T_{\mathcal{A}}} P_{r^{uniform}}[F] \leq \sum_{F \in T_{\mathcal{A}}} b \cdot P_{r^{planted}}[F] = b \cdot \sum_{F \in T_{\mathcal{A}}} P_{r^{planted}}[F].$$

It now remains to estimate  $\max_i \frac{p_i}{q_i}$ .

$$\begin{aligned} \frac{P_{r^{uniform}}[\mathcal{A}]}{P_{r^{planted}}[\mathcal{A}]} &\leq \max_i \frac{p_i}{q_i} = \max_i \left( \frac{t_i}{\sum_{j=1}^{2^n} t_j} \right) \cdot \left( \frac{2^n \cdot \Delta_{n,m}}{i \cdot t_i} \right) \\ &= \max_i \left( \frac{1}{i} \cdot \frac{\sum_{j=1}^{2^n} j \cdot t_j}{\sum_{j=1}^{2^n} t_j} \right) = \left( \frac{\sum_{j=1}^{2^n} j \cdot t_j}{\sum_{j=1}^{2^n} t_j} \right) \cdot \left( \max_i \frac{1}{i} \right) = C_1(n). \end{aligned}$$

■

Since directly estimating  $C_1(n)$  seems an intricate task, the following lemma is very useful.

**Lemma 2.23.**  $C_1(n) \leq C_2(n)$

**Proof.** To prove  $C_1(n) \leq C_2(n)$ , one needs to prove that

$$\left( \sum_{i=1}^{2^n} i \cdot t_i \right)^2 \leq \left( \sum_{i=1}^{2^n} t_i \right) \cdot \left( \sum_{i=1}^{2^n} i^2 \cdot t_i \right).$$

This is just Cauchy-Schwartz,  $(\sum a_i \cdot b_i)^2 \leq (\sum a_i^2) \cdot (\sum b_i^2)$ , with  $a_i = \sqrt{t_i}$  and  $b_i = i \cdot \sqrt{t_i}$ . ■

**Corollary 2.24.**  $(\mathcal{P}_{n,m} \rightarrow \mathcal{S}_{n,m})$  Let  $\mathcal{A}$  be some property of 3CNF formulas, then

$$Pr^{uniform}[\mathcal{A}] \leq C_2(n) \cdot Pr^{planted}[\mathcal{A}].$$

**Proposition 2.25.** Assume  $F$  is drawn from  $\mathcal{P}_{n,m}$ . Then  $C_2(n) \leq e^{ne^{-m/(3n)}}$  (for every ratio  $m/n$ ).

**Proof.** Let  $F$  be a planted instance with  $\varphi$  its planted assignment. Fix some assignment  $\psi$  at distance  $r$  from  $\varphi$ , and recall the event  $A_r$  (the proof of Proposition 2.19):  $\psi$  satisfies a random clause that is also satisfied by  $\varphi$ . As we established in the proof of Proposition 2.19,

$$Pr[\psi \text{ satisfies } F] \leq (Pr[A_r])^m \leq e^{-mr/(3n)}.$$

Therefore,

$$\begin{aligned} C_2(n) &\leq \sum_{r=0}^n \binom{n}{r} \cdot e^{-mr/(3n)} = \sum_{r=0}^n \binom{n}{r} \left( e^{-m/(3n)} \right)^r \cdot 1^{n-r} \\ &= \left( 1 + e^{-m/(3n)} \right)^n \leq e^{ne^{-m/(3n)}}. \end{aligned}$$

■

**Remark 2.26.** Observe that if for example  $m/n \geq 4 \log n$ , then  $C_2(n) = 1 + o(1)$ . That is, besides the planted assignment, one expects additional  $o(1)$  satisfying assignments. Put differently, (using the Markov inequality) when  $m/n \geq 4 \log n$  then *whp* there is only one satisfying assignment. This is then the regime where  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$  are statistically close (see Equation (1.1)).

## 2.6 Proof of Proposition 2.14

To prove Proposition 2.14 we use the exchange-rate technique, that is Lemma 2.7. We start by proving the proposition in the planted setting and then switch to the uniform. First let us estimate the number of variables that have too small a support.

**Proposition 2.27.** Assume  $F$  is drawn from  $\mathcal{P}_{n,m}$ ,  $m/n$  bigger than some sufficiently large constant. Let  $B$  be the number of variables whose support w.r.t.  $\varphi$  is smaller than  $m/(4n)$ . Then

$$Pr[|B| > e^{-m/(200n)} n] \leq e^{-ne^{-m/(50n)}}.$$

**Proof.** A variable  $x$  supports a random clause with probability  $3/(7n)$  (with probability  $3/n$  the variable  $x$  appears in the clause and once it appears, it supports the clause with probability  $1/7$ ). Therefore  $x$  supports in expectation  $3m/(7n)$  clauses. To claim concentration it would be useful if clauses were chosen independently. One standard “trick” is to consider the following distribution to generate  $F$ , denoted by  $\mathcal{P}_{n,p}$ : pick every clause consistent with the planted assignment with probability  $p$ . By setting  $p = m / \binom{n}{3}$ , it follows from standard calculations that  $F$  will have  $m$  clauses with probability  $\Omega(m^{-\frac{1}{2}})$ . It is easily verified that if some property holds with probability  $1 - \alpha$  in  $\mathcal{P}_{n,p}$ , it holds with probability  $1 - O(\alpha \cdot m^{\frac{1}{2}})$  in  $\mathcal{P}_{n,m}$ .

Now in  $\mathcal{P}_{n,p}$  the clauses are chosen independently, and  $x$  supports in expectation  $3m/(7n)$  clauses. Applying the Chernoff bound one obtains that

$$\Pr[x \text{ supports less than } m/(4n) \text{ clauses}] \leq e^{-m/150n}.$$

By the linearity of expectation, the expected number of variables that support less than  $m/(4n)$  clauses, call  $D$  the random variable that counts these variables, is  $e^{-m/150n}n$ . Now observe that the support of  $x$  and  $y$  are two independent events (as they concern disjoint sets of clauses that are chosen independently). Therefore we can use the Chernoff bound to the deviation of  $D$  for its expectation. Namely,

$$\Pr[D \geq e^{-m/(200n)}n] \leq \Pr[D \geq 2e^{-m/(150n)}n] \leq e^{-e^{-m/(150n)}n/3} \leq e^{-e^{-m/(100n)}n}.$$

Respectively, for  $\mathcal{P}_{n,m}$ ,

$$\Pr[D \geq e^{-m/(200n)}n] \leq O(m^{\frac{1}{2}}) \cdot e^{-ne^{-m/(100n)}}.$$

As we said, the interesting regime for us is  $m/n \leq 4 \log n$  (see Remark 2.26), and therefore the  $m^{\frac{1}{2}}$  factor is dominated by the exponentially decreasing part  $e^{-ne^{-m/(100n)}}$ . To conclude,

$$\Pr[D \geq e^{-m/(200n)}n] \leq e^{-ne^{-m/(50n)}}. \quad (2.1)$$

■

We now use Lemma 2.7 to obtain the same property in the uniform setting.

**Proposition 2.28.** *Assume  $F$  is drawn from  $\mathcal{S}_{n,m}$ ,  $m/n$  bigger than some sufficiently large constant. Let  $B_\varphi$  be the number of variables whose support w.r.t.  $\varphi$  is smaller than  $m/(4n)$ . There whp exists an assignment  $\varphi$  w.r.t. which*

$$|B_\varphi| \leq e^{-m/(200n)}n.$$

**Proof.** We call a satisfying assignment *good* if most variables have a good support w.r.t. to it, or formally: at most  $e^{-m/(200n)}n$  variables support less than  $m/(4n)$  clauses w.r.t. to it. Let  $\mathcal{A}$  be the following property: “there exists no good satisfying assignments”. Using the exchange-rate technique we obtain:

$$\begin{aligned} \Pr^{\text{uniform}}[\mathcal{A}] &\stackrel{\text{Lemma 2.7}}{\leq} e^{ne^{-m/(3n)}} \cdot \Pr^{\text{planted}}[\mathcal{A}] \stackrel{\text{Proposition 2.27}}{\leq} e^{ne^{-m/(3n)}} \cdot e^{-ne^{-m/(50n)}} \\ &= e^{n(e^{-m/(3n)} - e^{-m/(50n)})} = o(1). \end{aligned}$$

■

Finally, consider the following iterative procedure for defining a core w.r.t. some assignment  $\varphi$ . Set  $H^{(0)}$  to be all vertices that support at least  $m/(4n)$  clauses w.r.t.  $\varphi$ . Iteratively, remove a variable  $x$  from  $H^{(i)}$  if either  $x$  supports less than  $m/(6n)$  clauses in  $F[H^{(i)}]$ , or  $x$  appears in more than  $m/(10n)$  clauses where not all variables belong to  $H^{(i)}$ ;  $H^{(i+1)} = H^{(i)} \setminus \{x\}$ . Let  $t$  be the iteration where  $H^{(t)} = H^{(t+1)}$ , and set  $\mathcal{H} = H^{(t)}$ .

Set  $\alpha = 2e^{-m/(50n)}n$ , and let  $\varphi$  be the assignment promised in Proposition 2.27 (or 2.28). Let  $B$  be the variables that support less than  $m/(4n)$  clauses. Partition the variables not in  $\mathcal{H}$  into variables that belong to  $B$ , and variables that were removed in the iterative step –  $C$ . If  $|V \setminus \mathcal{H}| \geq \alpha n$ , then at least one of  $B, C$  has cardinality at least  $\alpha n/2$ . Consequently,

$$\Pr[|V \setminus \mathcal{H}| \geq \alpha n] \leq \underbrace{\Pr[|B| \geq \alpha n/2]}_{(a)} + \underbrace{\Pr[|C| \geq \alpha n/2 \mid |B| \leq \alpha n/2]}_{(b)}.$$

By the choice of  $\varphi$  and Proposition 2.27 (or 2.28)

$$\Pr[|B| \geq \alpha n/2] = \Pr[|B| \geq e^{-m/(50n)}n] = o(1).$$

To bound (b), observe that every variable that is removed in iteration  $i$  of the iterative step supports at least  $m/(4n) - m/(6n) \geq m/(20n)$  clauses in which at least another variable belongs to  $U = \{a_1, a_2, \dots, a_{i-1}\} \cup B$ . Consider iteration  $\alpha n/2$ . Assuming  $|B| \leq \alpha n/2$ , by the end of this iteration there exists a set containing at most  $|U| = \alpha n$  variables, and there are at least  $m/(20n) \cdot \alpha n/2 = m/(40n) \cdot |U|$  clauses containing at least two variables from it (note that no clause was counted twice as the supporter of a clause is unique). However,  $|U| \leq n/4000$  (for sufficiently large  $m/n$ ) and thus Proposition 2.5 can be used to bound (b).

Proposition 2.14 follows from combining these results.

## 2.7 Proof of Proposition 2.16

Let  $d = \frac{m}{n}$ . Let us say that a 3CNF  $F$  is *bounded* if the following conditions hold.

- B1.** For all  $X \subset V$  such that  $|X| \leq n/d^2$  there are at most  $10|X|$  clauses containing more than two variables from  $X$ .
- B2.** Every variable appears in at most  $\ln^2 n$  clauses.
- B3.** Let  $H$  be a subformula of  $F$  on  $|V(H)| \geq (1 - d^{-10})n$  variables, so that every variable in  $H$  supports at least  $d/50$  clauses in  $H$ . Then  $H$  is uniquely satisfiable.

Moreover, we call  $F$   $\varepsilon$ -feasible if  $F$  has an induced subformula  $H$  with the following properties.

- F1.**  $|V(H)| \geq (1 - \varepsilon \exp(-\sqrt{d}))n$  and  $|H|$  contains at least  $(1 - \varepsilon)m$  clauses.
- F2.** There exists a satisfying assignment  $\varphi$  of  $F$  so that every variable  $x \in H$  supports at least  $(1 - \varepsilon)d/5$  clauses in  $H$  w.r.t.  $\varphi$ .
- F3.** Every variable in  $H$  appears in at most  $\varepsilon d$  clauses where not all variables belong to  $H$ .

**F4.**  $H$  is uniquely satisfiable.

If  $H, K$  are two induced subformulas of  $F$  that satisfy F1–F4, then the same is true for  $H \cup K$ . Therefore,  $F$  has a unique maximal induced subformula that enjoys F1–F4; this subformula will be denoted by  $F_\varepsilon$  in the sequel. Also observe that if  $F$  is  $\varepsilon$ -feasible then it is also  $\varepsilon'$ -feasible for  $\varepsilon' \leq \varepsilon$  and it holds that  $V(F_\varepsilon) \subseteq V(F_{\varepsilon'})$  (by  $V(F)$  we denote the variable set of a formula  $F$ ).

**Lemma 2.29.** *For any fixed  $\varepsilon > 0$   $\mathcal{S}_{n,m}$  is bounded and  $\varepsilon$ -feasible whp.*

The Lemma follows from a discussion very similar to the one in Section 2.3. Also observe that  $F_{0.02}$  for example is contained in the core defined according to Definition 2.11. Therefore when reading the proof of Proposition 2.16, it will be easiest to think of the core  $\mathcal{H}$  as  $F_{0.02}$ .

Let  $T \subset V$  be a set of size  $t = \lceil \log n \rceil$ , and let  $\tau$  be a tree with vertex set  $T$ . Let  $F_\tau$  be a fixed collection of clauses such that each edge of  $\tau$  is induced by some clause of  $F_\tau$ . We say that a clause set  $F_\tau$  is *minimal* w.r.t.  $\tau$  if by deleting a clause from  $F_\tau$ ,  $\tau$  is not induced by  $F_\tau$  anymore. By the definition of minimality,  $|F_\tau| \leq |E(\tau)| = |V(\tau)| - 1$  (as  $\tau$  is a tree). Moreover, let us call  $F$   $(T, \tau, F_\tau)$ -poor if

- $F$  is bounded,
- $F$  is 0.01-feasible,
- $F$  contains  $F_\tau$  as a subformula,
- $V(\tau)$  does not intersect  $F_{0.02}$ .

Denote by  $\mathcal{G}$  the set of all satisfiable 3CNF formulas with variable set  $V = \{x_1, \dots, x_n\}$  and exactly  $m$  clauses, and let  $\mathcal{P}(T, \tau, F_\tau)$  signify the set of all  $(T, \tau, F_\tau)$ -poor formulas  $F \in \mathcal{G}$ . Below we shall establish the following.

**Lemma 2.30.** *We have  $\#\mathcal{P}(T, \tau, F_\tau) \leq \left( e^{-\Theta(tm/n)} \cdot \left(\frac{m}{n^3}\right)^{\#F_\tau} \right) \#\mathcal{G}$ .*

Where by  $\#F_\tau$  we denote the number of clauses in  $F_\tau$ . Before we prove Lemma 2.30, let us note that it implies Proposition 2.16 immediately (noticing that the core  $\mathcal{H}$  is contained in  $F_{0.02}$ ).

### 2.7.1 Proof of Proposition 2.16

In this section we prove Proposition 2.16 for  $\mathcal{S}_{n,m}$ . Nevertheless, the same discussion as in Section 2.4.1 provides the correctness of the statement for  $\mathcal{P}_{n,m}$ .

Given a tree  $T$  of size  $b$  (in our case  $b = \log n$ ), we estimate the number of ways to extend  $T$  to a minimal inducing set  $F_T$ . Every clause in  $F_T$  can cover either one or two edges of  $T$  (if cannot cover three edges or we have a cycle in  $T$ ). Following the argument in [28], let  $N_{T,s}$  be the number of ways to pair  $2s$  edges of  $T$  to form  $s$  clauses in  $F_T$  which cover two edges. There are 8 ways to set the negations in every clause of  $F_T$  (and there are  $b - 1 - s$  such clauses), and  $n^{b-1-2s}$  ways to choose the third variable in the  $b - 1 - 2s$  clauses that cover exactly one edge.

Using this terminology, the probability that a random formula  $F$  contains a set of clauses which induce a  $b$ -tree whose vertices do not intersect  $\mathcal{H}$  is at most:

$$\sum_{T:|V(T)|=b} \sum_{s=0}^{b/2} N_{T,s} 8^{b-1-s} n^{b-1-2s} \cdot \Pr[F_T \subseteq F \text{ and } V(T) \cap \mathcal{H} = \emptyset],$$

and using Lemma 2.30, the latter is at most

$$\sum_{b\text{-trees}} \sum_{s=0}^{b/2} N_{T,s} 8^{b-1-s} n^{b-1-2s} \left(\frac{d}{n^2}\right)^{b-1-s} e^{-\Omega(bd)} \leq \sum_{b\text{-trees}} \left(\sum_{s=0}^{b/2} N_{T,s}\right) (8d)^b n^{1-b} e^{-\Omega(bd)}, \quad (2.2)$$

where  $d = 3m/(4n)$ . Our last task is to obtain useful upper bounds on the sum  $\left(\sum_{s=0}^{b/2} N_{T,s}\right)$ . To this end let us fix a degree sequence  $(d_1, \dots, d_b)$  for  $T$ , and consider the following procedure for *properly* pairing edges. By proper we mean that every pair of edges can be covered by a 3CNF clause; for example, we cannot pair the edges  $(x_1, x_2)$  and  $(x_3, x_4)$  as they result in a 4CNF clause. For each vertex, we specify a permutation of the edges incident to that vertex. Then we iterate through the vertices, and for each vertex, we iterate through the edges and pair up each unpaired edge with the edge given by the permutation associated with the current vertex (and leave the edge unpaired if the permutation sends the edge to itself). Any pairing of edges which can be covered by clauses can be generated this way by choosing the permutations to transpose each pair of edges to be covered by a single clause and to leave fixed all the other edges. Since there are  $d_i!$  different permutations for vertex  $i$ , we have

$$\sum_{s=0}^{b/2} N_{T,s} \leq \prod_{i=1}^b d_i!.$$

A classical result by Prüfer is that the number of  $b$ -trees with degree sequence  $(d_1, \dots, d_b)$  equals  $\binom{b-2}{d_1-1, \dots, d_b-1}$  (see, for example, [41], Section 4.1, p. 33). There are  $\binom{n}{b}$  ways to choose the  $b$  vertices of the tree. So (2.2) is at most

$$\begin{aligned} & \sum_{d_1+\dots+d_b=2(b-1)} \binom{n}{b} \binom{b-2}{d_1-1, \dots, d_b-1} \left(\prod_{i=1}^b d_i!\right) (8d)^b n^{1-b} e^{-\Omega(db)} \\ & \leq \sum_{d_1+\dots+d_b=2(b-1)} \frac{1}{b(b-1)} \left(\prod_{i=1}^b d_i\right) (8d)^b n e^{-\Omega(db)}. \end{aligned}$$

By convexity, for  $(d_1, \dots, d_b)$  with  $d_1 + \dots + d_b = 2(b-1)$ , the product  $\left(\prod_{i=1}^b d_i\right)$  is maximized when  $d_1 = \dots = d_b$ , and so  $\left(\prod_{i=1}^b d_i\right) \leq 2^b$ . The number of ways to choose positive integers  $(d_1, \dots, d_b)$  so that  $d_1 + \dots + d_b = 2(b-1)$  is  $\binom{2(b-1)+b-1}{2(b-1)}$  which is less than  $2^{3b}$ . Thus, the probability that  $F$  contains a set of clauses which cover a  $b$ -tree whose vertex set is disjoint from  $\mathcal{H}$  is at most

$$n 2^{3b} 2^b (8d)^b e^{-\Omega(db)}.$$

Finally, recall that we are only interested in trees of size  $\log n$ , therefore setting  $b = \log n$  we obtain:

$$n2^{3\log n}2^{\log n}(8d)^{\log n}e^{-\Omega(d\log n)} = n^{-\Omega(d)} = o(1),$$

assuming we have chosen  $d = \Theta(m/n)$  sufficiently large.

### 2.7.2 Proof of Lemma 2.30

Thus, the remaining task is to prove Lemma 2.30. To this end we fix a set  $T$  of variables, a tree  $\tau$  on the variables  $T$ , and a minimal inducing set of clauses  $F_\tau$ . We set up a bipartite auxiliary graph  $\mathcal{A} = \mathcal{A}(T, F_\tau, \tau)$  with vertex set  $V(\mathcal{A}) = \mathcal{P}(T, F_\tau, \tau) \oplus \mathcal{G}$ ; for brevity we set  $\mathcal{P} = \mathcal{P}(T, F_\tau, \tau)$ . The auxiliary graph will enjoy the following property. In  $\mathcal{A}$  every vertex  $F \in \mathcal{P}$  has degree at least  $\Delta$ , while every vertex  $F' \in \mathcal{G}$  has degree at most  $\Delta'$  s.t.

$$\Delta' \leq \left( e^{-\Theta(tm/n)} \cdot \left( \frac{m}{n} \right)^{\#F_\tau} \right) \Delta \quad (2.3)$$

Since  $\Delta\#\mathcal{P}(T, \tau) \leq \#E(\mathcal{A}) \leq \Delta'\#\mathcal{G}$ , Lemma 2.30 follows directly from (2.3).

To describe the construction of  $\mathcal{A}$  we let  $I$  be the set of all  $x \in T$  that appear in at most 6 clauses in  $F_\tau$ ; then  $\#I \geq t/2$ , because  $\#F_\tau \leq \#V(\tau) - 1$  (as  $\tau$  is a tree and  $F_\tau$  is minimal). Let  $\varphi$  be the unique satisfying assignment of  $F_{0.02}$ . We define the following partition of  $I$ :

$$\begin{aligned} I_1(F) &= \{x \in I : x \text{ appears in at least } 0.02d \text{ clauses where all variables belong to } V \setminus F_{0.02}\}, \\ I_2(F) &= \{x \in I : x \text{ supports at most } (1 - 0.02)d/5 \text{ clauses w.r.t. } \varphi \text{ where the other two variables} \\ &\quad \text{belong to } F_{0.02}\} \setminus I_1(F). \end{aligned}$$

If  $F$  is  $(T, F_\tau, \tau)$ -poor, then all variables  $x \in I$  are outside of the 0.02-core  $F_{0.02}$ ; hence, due to F1–F4 we have  $I = I_1(F) \cup I_2(F)$ . Thus, we decompose  $\mathcal{P}$  into two parts  $\mathcal{P}_1 = \{F \in \mathcal{P} : \#I_1(F) \geq 0.15t\}$ ,  $\mathcal{P}_2 = \mathcal{P} \setminus \mathcal{P}_1$  (that is,  $\#I_2(F)$  is at least  $0.85t$ ).

As a next step, we will construct two subgraphs  $\mathcal{A}_1, \mathcal{A}_2$  of  $\mathcal{A}$ , both of which consist of the  $\mathcal{P}_i$ - $\mathcal{G}$ -edges of  $\mathcal{A}$ . Thus,  $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ , so that (2.3) will be a consequence of the following statement. In  $\mathcal{A}_j$  every vertex  $F \in \mathcal{P}_j$  has degree at least  $\Delta_j$ , while every vertex  $F' \in \mathcal{G}$  has degree at most  $\Delta'_j$  where

$$\Delta'_j \leq \left( e^{-\Theta(tm/n)} \cdot \left( \frac{m}{n^3} \right)^{\#F_\tau} \right) \Delta_j \quad (j = 1, 2). \quad (2.4)$$

In the remainder of this section we present the constructions of  $\mathcal{A}_1, \mathcal{A}_2$  and establish (2.4). To facilitate these constructions we say that a triplet  $\{x, y, z\}$  of variables is *compatible* if there is no clause in  $F$  involving all three variables, and  $x, y, z$  lie in  $F_{0.01}$ . Moreover, we say that a set  $F$  of triplets of variables is compatible if every triplet in  $F$  is compatible and no variable  $x$  occurs in more than one triplet.

**Lemma 2.31.** *Let  $F \in \mathcal{P}$  and let  $1 \leq s \leq n^{0.1}$ . There exist  $\binom{n^{3/4}}{s}$  compatible sets  $F'$  of size  $s$ .*

**Proof.** Let  $\varphi$  signify the unique satisfying assignment of  $F_{0.01}$ , and let  $\mathcal{C}$  be all the clauses over the variables of  $F_{0.01}$  that are satisfied by  $\varphi$ . Since  $F$  satisfies F1,  $\mathcal{C}$  has at least  $7\binom{0.99n}{3} \geq n^3/2$  clauses. Furthermore, let  $S$  be a set of  $s$  clauses of  $\mathcal{C}$  chosen uniformly at random. Then the probability that  $S$  does not contain a clause of  $F$  is

$$\binom{\#\mathcal{C} - m}{s} \binom{\#\mathcal{C}}{s}^{-1} = \prod_{j=0}^{s-1} 1 - \frac{m}{\#\mathcal{C} - j} = 1 - o(1),$$

because  $\#\mathcal{C} = \Omega(n^3)$ , while  $ms = o(n^3)$ . Moreover, the probability that a specific variable  $x$  occurs twice in  $S$  is at most

$$\binom{7n^2}{2} \binom{\#\mathcal{C}}{s-2} \binom{\#\mathcal{C}}{s}^{-1} \leq O(s^2 n^{-2}) = o(n^{-1}).$$

Hence, by the union bound with probability  $1 - o(1)$  a randomly chosen  $S$  will touch no variable  $x$  more than once. Thus, with probability  $1 - o(1)$  a randomly chosen  $S$  is compatible, so that the number of compatible sets is at least  $(1 - o(1))\binom{\#\mathcal{C}}{s} > \binom{n^3/4}{s}$ . ■

**Construction of  $\mathcal{A}_1$ .** The construction of  $\mathcal{A}_1$  is based on the following observation.

**Lemma 2.32.** *Suppose that  $F \in \mathcal{P}_1$ . There exist sets  $U \subset I_1(F)$ ,  $\#U = \lceil 0.1t \rceil$ , and  $W \subset V \setminus (\tau \cup F_{0.02})$  such that for every  $x \in U$ ,  $x$  appears in at least  $d/1000$  clauses where the other two variables belong to  $W$ , and for every  $y \in W$ ,  $y$  appears in at most 1000 clauses where the other two variables belong to  $U$ .*

**Proof.** Let  $J \subset I_1(G)$  be a set of size  $0.15t$ , and let  $K$  be the set of all variables  $w \in V \setminus (F_{0.02} \cup \tau)$  s.t. there exists a clause in  $F$  containing  $w$  and some variable from  $J$ . Moreover, let  $L \subset K$  be the set of all  $w \in K$  such that the number of clauses containing  $w$  and at least one more variables in  $J$  is at least 1000. Then the boundedness property of  $F$  implies that  $\#L \leq 0.01t$ . Furthermore, letting  $Q$  be all the variables  $w \in J$  such that the number of clauses containing  $w$  and at least one more variable from  $L$  is at least 1000. Then we have  $\#Q \leq 0.01t$  (once more due to the boundedness of  $F$ ). Now, let  $U = J \setminus Q$  and  $W = K \setminus L$ . Then each  $w \in W$  appears in at most 1000 clauses where the other two variables are in  $U$ . Moreover, if  $v \in U$ , then the number of clauses that contain  $v$  and two variables in  $K$  is at least  $0.02d - 6 - 10^4 \geq 0.015d$ . Furthermore,  $U$  has the required size. ■

Our objective is to associate with each  $F \in \mathcal{P}_1$  a large number of “target graphs”  $F' \in \mathcal{G}$  such that no  $F'$  occurs as a target graph too frequently. To this end, we consider the following nondeterministic procedure that maps  $F$  to a target graph  $F'$ . For each possible outcome  $F'$  we include the edge  $\{F, F'\}$  into  $\mathcal{A}_1$ .

Set  $\gamma = \lceil d/1000 \rceil$  and  $u = \lceil 0.1t \rceil$ .

**C1.** Choose a compatible set  $C$  of size  $\#F_\tau + \gamma u$ .

**C2.** Choose sets  $U$  and  $W$  as in Lemma 2.32.

**C3.** For each  $x \in U$  choose a set  $N_x$  of  $\gamma$  clauses of the form  $(\ell_x \vee \ell_y \vee \ell_z)$ ,  $y, z \in W$ .

**C4.** Obtain  $F'$  from  $F$  by removing the clauses of  $F_\tau$  along with the clauses of C3 and adding the clauses from  $C$ .

Lemmas 2.31 entails that the number of formulas  $F'$  that can be obtained from each  $F$  via the above procedure is at least

$$\Delta_1 = \binom{n^3/4}{\#F_\tau + \gamma u} \quad (2.5)$$

(because there are at least this many choices in step C1, and one can always set the signs of variables in each new clause so that the obtained formula remains satisfiable – as we started with a satisfiable one). Conversely, to recover  $F$  from  $F'$ , we consider the following nondeterministic procedure.

**R1.** Choose a set  $C'$  of  $\#F_\tau + \gamma u$  clauses of  $F'$ .

**R2.** Choose a set  $U' \subset T$  of size  $u$ .

**R3.** For each such  $x \in U'$  choose a set  $N'_x$  of  $\gamma$  pairs of variables outside of  $F_{0.015}$ , and a way to set the signs of the variables in each clause.

**R4.** Output the formula  $F''$  obtained from  $F'$  by removing the clauses  $C'$  and adding the ones  $(\ell_x \vee \ell_y \vee \ell_z)$ ,  $x \in U'$ ,  $y, z \in N'_x$  along with the clauses of  $F_\tau$ .

**Lemma 2.33.** *If  $\{F, F'\}$  is an edge of  $\mathcal{A}_1$ , then  $F'$  is 0.015-feasible and the process R1–R4 applied to  $F'$  can yield the output  $F'' = F$ .*

**Proof.** Let  $C, U, W$ , and  $(N_x)_{x \in U}$  be the sets chosen by C1–C4 to obtain  $F'$  from  $F$ . If R1–R4 chooses  $F' = F$ ,  $U' = U$ ,  $N'_x = N_x$  for all  $x \in U$ , then the outcome will be  $F'' = F$ . Thus, we just need to show that it is feasible for R1–R4 to choose  $N'_x = N_x$ , i.e., that  $F'$  is 0.015-feasible and the vertices in  $N'_x$  do not belong to  $F'_{0.015}$ .

It suffices to show that  $V(F'_{0.015}) \subseteq V(F_{0.02})$  (since all the variables in  $N_x$  lie outside  $F_{0.02}$ , and in particular outside  $F'_{0.015}$ ).

To see that  $F'$  is 0.015-feasible, let  $X$  be the variable set of  $F_{0.01}$ . We claim that  $X$  satisfies F1–F4 with respect to  $F'$  with  $\varepsilon = 0.01$ . F1 is an immediate consequence of the fact that  $F$  is 0.01-feasible. Moreover, as C4 adds a compatible set  $C$  and only removes clauses that contain variables outside of  $X$ , the unique satisfying assignment of  $F_{0.01}$  remains then unique for the set  $X$  in  $F'$ , whence F2–F4 follow. Thus,  $F'$  is indeed 0.01-feasible, and hence 0.015-feasible as well.

Finally, to show that the variable set  $Y$  of  $F'_{0.015}$  is contained in that of  $F_{0.02}$ , we show that  $Y$  is 0.02-feasible in  $F$ . Requirement  $F_1$  is satisfied since  $F'$  is 0.015-feasible (and  $0.015 > 0.02$ ). Further observe that if  $F'[Y]$  is uniquely satisfiable then so is  $F[Y]$ , this is because when moving from  $F'$  to  $F$  one can either add clauses, or remove clauses that are uniquely satisfied in  $F$  to begin with (therefore removing them incurs no addition of satisfying assignments) – thus requirement F4 follows. Now consider the  $(1 - 0.015)d/5$  clauses that every  $y \in Y$  supports in  $F'[Y]$ , then since  $F' \setminus F$  contains at most 1 clause involving  $y$  (as  $C$  is a compatible set), it holds that  $y$  supports at least  $(1 - 0.015)d/5 - 1 \geq (1 - 0.02)d/5$  clauses w.r.t.  $F[Y]$  (requirement  $F_2$ ). Lastly, observe that if  $y$  appears in at most  $0.015d$  clauses in  $F'$  where not all variables

belong to  $Y$ , then in  $F$  this number could have been at most  $0.015d + \gamma \leq 0.02d$ , by the choice of  $\gamma$  (this establishes requirement  $F_3$ ). ■

**Lemma 2.34.** *If  $F'$  is an outcome of C1–C4 for some  $F \in \mathcal{P}_1$ , then the number of possible nondeterministic choices in the R1–R4 is at most  $\Delta'_1 = 2^{\#T} \binom{m}{\#F_\tau + \gamma u} \left( \frac{\exp(-\sqrt{d})n^2}{\gamma} \right)^u$ .*

**Proof.** The first factor accounts for the number of ways to choose  $F'$ . Moreover, there are clearly at most  $2^{\#T}$  ways to choose  $U'$  (recall that in our setting  $\#T = \log n$ ). To bound the number of choices of R3, note that for each  $x \in U'$  there are at most  $\left( \binom{n - |V(F'_{0.015})|}{\gamma} \right)^2$  ways to choose the set  $N'_x$ . By the definition of  $F'_{0.015}$  (requirement  $F_1$ ) it holds that  $|V(F'_{0.015})| \geq n(1 - \exp(-\sqrt{d}))$ . ■

Finally, combining (2.5) with Lemma 2.33 and 2.34, and using standard estimates for the binomial coefficients, one obtains (2.4) for  $j = 1$ .

**Construction of  $\mathcal{A}_2$ .** As in the construction of  $\mathcal{A}_1$  we consider a nondeterministic procedure that maps  $F \in \mathcal{P}_2$  to  $F' \in \mathcal{G}$ . Let  $u = \lceil 0.1t \rceil$  and  $\gamma = \lceil 10^{-9}d \rceil$ . Let  $\varphi$  be the unique satisfying assignment of  $F_{0.01}$ .

**C1.** Choose a compatible set  $C$  of size  $\#F_\tau$ .

**C2.** Choose a subset  $U \subset I_2(F)$  of size  $u$ .

**C3.** Choose a set of clauses  $M \subset F_{0.01}$  of size  $\gamma u$  s.t. every two clauses in  $M$  are variable-disjoint, and no variable  $x$  appears in more than 100 clauses with a variable from  $M$ . Moreover, for each  $x \in U$  choose a set  $N_x$  of clauses  $C = (\ell_x \vee \ell_y \vee \ell_z)$  s.t.  $y, z \in F_{0.02}$ , there in no clause in  $F$  that contains both  $x, z$  or both  $x, y$ , and no variable in  $N_x$  occurs in  $M$ . Set the negation in every such  $C$  so that  $x$  supports  $C$  w.r.t.  $\varphi$ . Moreover, the sets  $(N_x)_{x \in U}$  should be pairwise disjoint.

**C4.** Obtain  $F'$  from  $F$  by removing the clauses of  $F_\tau$  and the set  $M$ , adding the clauses of  $C$ , and adding all the clauses  $(N_x)_{x \in U}$ .

For each  $F \in \mathcal{P}_2$  and each possible outcome  $F'$  of C1–C4 we include the edges  $\{F, F'\}$  into  $\mathcal{A}_2$ . The following lemma provides a lower bound on the degree of  $F \in \mathcal{P}_2$  in  $\mathcal{A}_2$ .

**Lemma 2.35.** *Each  $F \in \mathcal{P}_2$  has at least  $\Delta_2 = \frac{1}{2} \binom{n^3/4}{\#F_\tau} \cdot \binom{m/2}{\gamma u} \cdot \left( n^2 \frac{1 - e^{-\sqrt{d}}}{\gamma} \right)^u$  images  $F'$ .*

**Proof.** By Lemma 2.31 there are  $\binom{n^3/4}{\#F_\tau}$  ways to choose  $C$ . Furthermore, property F1 implies that  $F_{0.01}$  contains at least  $m/2$  clauses. Moreover, since every variable appears in at most  $\ln^2 n$  clauses in  $F$ , property B2, and the boundedness of  $F$  (property B1 – which assures that for every  $M$  of size  $\gamma u$  there are not too many variables that appear in at least 100 clauses with some other variable from  $M$ ) then  $F_{0.01}$  has at least  $(1 - o(1)) \binom{m/2}{\gamma u}$  sets  $M$  of size  $\gamma u$ . Finally, since  $\#V(F_{0.02}) \geq (1 - 0.02e^{-\sqrt{d}})n$  by F2, there are at least  $\left( n^2 \frac{1 - e^{-\sqrt{d}}}{\gamma} \right)^u$  ways to choose the sets  $(N_x)_{x \in U}$ . ■

Conversely, we consider the following nondeterministic procedure for obtaining a formula  $F''$  from an outcome  $F'$  of C1–C4.

- R1.** Choose a set of clauses  $C'$  from  $F'$  of size  $\#F_\tau$ .
- R2.** Determine the unique satisfying assignment  $\varphi$  of  $F'_{0.015}$ . Then, choose a set  $U' \subset T$  of size  $u$ . Moreover, choose a set  $M'$  of  $\gamma u$  clauses out of all possible ones.
- R3.** For each  $x \in U'$  out of the clauses that  $x$  supports in  $F'_{0.015}$ , choose  $\gamma$  such clauses and set  $N'_x$  to be that set of these clauses.
- R4.** Obtain a formula  $F''$  from  $F'$  by removing  $C'$  and all clauses in  $N'_x$ , and adding the clauses of  $F_\tau$  and  $M'$ .

**Lemma 2.36.** *If  $\{F, F'\}$  is an edge of  $\mathcal{A}_2$ , then  $F'$  is 0.015-feasible and the process R1–R4 applied to  $F'$  can yield the output  $F'' = F$ .*

**Proof.** Suppose that  $F'$  has been obtained from  $F$  by choosing the sets  $M, U$ , the sets  $(N_x)_{x \in U}$ , and the compatible set  $C$ . To recover  $F'' = F$ , we shall prove that  $F'$  is 0.015 feasible and that the process R1–R4 can choose  $M' = M, C' = C$ , and  $N'_x = N_x$ .

To see that  $F'$  is 0.015-feasible, let  $X$  be the variable set of  $F_{0.01}$ . We claim that  $X$  satisfies F1–F4 with respect to  $F'$  with  $\varepsilon = 0.015$ . For F1 is an immediate consequence of the fact that  $F$  is 0.01-feasible. Moreover, as C1 adds a compatible set  $C$ , and the  $N_x$ 's are variable disjoint, and of size at most  $\gamma$ , then every  $x \in X$  appears in at most  $0.01d + \gamma + 1 \leq 0.015d$  clauses where not all variables belong to  $X$ . Every  $x \in X$  supports at least  $(1 - 0.01)d/5$  clauses where all other variables belong to  $X$ , and since  $M$  removes at most 100 clauses per variable, and  $C$  is compatible, then  $x$  supports at least  $(1 - 0.01)d/5 - 100 - 1 \geq (1 - 0.015)d/5$  clauses (w.r.t. the same satisfying assignment, which remains satisfying for  $F'$  as well). Lastly,  $F'[X]$  is uniquely satisfiable by property B3. This in turn implies that the same unique assignment satisfies  $F[X]$  (otherwise, if  $\varphi'$  is the different assignment that satisfies  $F[X]$  then  $\varphi'$  also satisfies  $F'[X]$ , as in  $F'[X]$  we either removed clauses or added clauses which are satisfied by  $\varphi'$  – contradicting the uniqueness).

It is clear that R1–R4 can choose  $C' = C$  and  $M' = M$  since we had no restrictions in step R2. It only remains to show that it is feasible for the reconstruction procedure to have chosen  $N'_x = N_x$ . To this end, it suffices to show that  $V(F'_{0.015}) \subseteq V(F_{0.02})$  (since all the variables in  $N_x$  lie outside  $F_{0.02}$ , and then in particular outside  $F'_{0.015}$ , and the set of clauses that  $x$  supports in  $F'_{0.015}$  will then contain the set of clauses that it supports in  $F_{0.02}$  – as the unique satisfying assignment is the same). Let  $Y$  be the set of variables of  $F'_{0.015}$ . We show that  $Y$  is 0.02-feasible in  $F$ . Requirement  $F_1$  is satisfied since  $F'$  is 0.015-feasible (and  $0.015 > 0.02$ ). Further observe that if  $F'[Y]$  is uniquely satisfiable then so is  $F[Y]$ , this is because when moving from  $F'[Y]$  to  $F[Y]$  one can either add clauses, or remove clauses that are uniquely satisfied in  $F$  to begin with (therefore removing them incurs no addition of satisfying assignments) – thus requirement F4 follows. Now consider the  $(1 - 0.015)d/5$  clauses that every  $y \in Y$  supports in  $F'[Y]$ , then since  $F' \setminus F$  contains at most 1 clause involving  $y$  (as  $C$  is a compatible set), it holds that  $y$  supports at least  $(1 - 0.015)d/5 - 1 \geq (1 - 0.02)d/5$  clauses w.r.t.  $F[Y]$  (requirement  $F_2$ ).

Lastly, observe that if  $y$  appears in at most  $0.015d$  clauses in  $F'$  where not all variables belong to  $Y$ , then since the  $N_x$  are of size  $\gamma$  and variable-disjoint – in  $F$  this number could have been at most  $0.015d + \gamma \leq 0.02d$  (by the choice of  $\gamma$ ) – this establishes requirement  $F_3$ . ■

In the light of Lemma 2.36 we can bound the degrees of  $F' \in \mathcal{G}$  in  $\mathcal{A}_2$  as follows.

**Lemma 2.37.** *If  $G'$  has been obtained from  $G$  via C1–C4, then during R1–R4 there are at most  $\Delta'_2 = \binom{m}{\#F_\tau} 2^t \binom{\binom{n}{3}}{\gamma u} \binom{d/5}{\gamma}^u$  ways to choose  $F'$ , the sets  $N'_x$ , and  $M'$ .*

**Proof.** There are exactly  $\binom{m}{\#F_\tau}$  ways to choose  $C'$  and at most  $2^t$  ways of choosing  $U'$ . Furthermore, there are at most  $\binom{\binom{n}{3}}{\gamma u}$  ways to choose  $M'$ . Finally, since each  $x \in U'$  has at most  $(1 - 0.02)d/5 + \gamma u \leq d/5$  clauses that it supports w.r.t.  $F'_{0.015}$ , therefore there are at most  $\binom{d/2}{\gamma}$  ways to choose  $N'_x$ . ■

Combining the bounds from Lemmas 2.35 and 2.37 establishes (2.4) for  $j = 2$ .

## 2.8 Proof of Theorem 2.1

In this section we prove Theorem 2.1. Like in the proof of Theorem 2.2, we work our way through the planted distribution. Unlike the proof of Theorem 2.2, we do not derive our result by establishing structural properties like the existence of a core, but rather analyze directly relations between satisfying assignments.

### 2.8.1 Relating the uniform and the planted distributions

For a formula drawn from  $\mathcal{S}_{n,m}$ , let  $u_x$  count the pairs of satisfying assignments at distance  $xn$  from each other. Similarly for  $\mathcal{P}_{n,m}$ , let  $f_x$  count the number of satisfying assignments at distance  $xn$  from the planted assignment. Also recall the definition of  $C_1, C_2$  from Section 2.5 ( $C_1$  is the expected number of satisfying assignments for  $\mathcal{S}_{n,m}$ , and  $C_2$  for  $\mathcal{P}_{n,m}$ ). In our new terminology  $C_2 = \sum_x E[f_x]$ . The following proposition allows us to upper bound  $E[u_x]$  via the more accessible quantity  $E[f_x]$ . A similar version of this proposition appears in [44].

**Proposition 2.38.** *Assume  $F$  is drawn from  $\mathcal{S}_{n,m}$ , then*

$$E[u_x] = C_1 \cdot E[f_x]/2.$$

**Proof.** For two satisfying assignments  $\varphi_i, \varphi_j$  we use  $\delta(\varphi_i, \varphi_j)$  to denote their Hamming distance. Consider some ordering on the  $2^n$  possible assignments, and let  $A_i$  be an indicator variable which is 1 if  $\varphi_i$  satisfies  $F$ . Using this terminology,

$$u_x = \frac{1}{2} \sum_{i,j:\delta(\varphi_i,\varphi_j)=xn} A_i \cdot A_j.$$

Linearity of expectation gives

$$E[u_x] = \frac{1}{2} \sum_{i,j:\delta(\varphi_i,\varphi_j)=xn} Pr[A_i \wedge A_j] = \frac{1}{2} \sum_{\delta(\varphi_i,\varphi_j)=xn} Pr[A_i|A_j] Pr[A_j].$$

By symmetry, the latter equals

$$2^n \cdot \frac{\Pr[A_j]}{2} \cdot \sum_{\varphi_i} \Pr[A_i|A_j].$$

It remains to estimate  $\Pr[A_i|A_j]$ . Conditioning on the event  $A_j$  means conditioning on the fixed assignment  $\varphi_j$  to be satisfying. In turn,  $\mathcal{S}_{n,m}$  conditioned on  $\varphi_j$  being a satisfying assignment means that only clauses which are satisfied by  $\varphi_j$  can be included, and by symmetry, every set of  $t$  clauses satisfied by  $\varphi_j$  has the same probability of being included. Observe that for  $t = m$  this is exactly the definition of the planted distribution  $\mathcal{P}_{n,p}$ . Therefore  $\sum_i \Pr[A_i|A_j] = E[f_x]$ , when summing over all assignments  $\varphi_i$  at distance  $xn$  from  $\varphi_j$ . Furthermore,  $C_1 = \sum_i \Pr[A_i]$  (now we are summing over all  $2^n$  assignments), and hence  $\Pr[A_i] = C_1/2^n$ . Putting everything together we derive

$$E[u_x] = C_1 \cdot E[f_x]/2.$$

■

In [44] this sort of proposition was already enough to estimate  $E[u_x]$  since  $C_1$  can be easily calculated when  $m/n$  is below the satisfiability threshold. However in  $\mathcal{S}_{n,m}$ ,  $m/n$  above the satisfiability threshold, it is not clear how to calculate  $C_1$ . Recall Lemma 2.23 which establishes the following:

**Lemma 2.39.** *Always  $C_1 \leq C_2$ .*

Therefore we can use the more convenient relation:

$$E[u_x] = C_2 \cdot E[f_x]/2.$$

## 2.8.2 The Planted Setting

In this section we analyze  $C_2$  and  $E[f_x]$ . Recall that we use  $C_2$  to denote the expected number of satisfying assignments that a random formula in  $\mathcal{P}_{n,p}$  has, and  $f_x$  counts the number of satisfying assignments at distance  $xn$  from the planted assignment, had  $F$  belonged to  $\mathcal{P}_{n,p}$ .

Our analysis of  $E[f_x]$  is composed of two regimes. The first is the case  $x \in [0, 1/k]$ . In this regime  $E[f_x]$  changes from  $\omega(1)$  to  $o(1)$ .

To translate our results to the uniform setting, it turns out that we need to have a more precise control on the rate in which  $E[f_x]$  decreases once changing to  $o(1)$ . Therefore the analysis of that regime is more careful (Proposition 2.43). Then we analyze the case  $x \in [1/k, 1]$ . In this regime, for a suitable choice of  $\varepsilon$  (recall  $m/n = (1 + \varepsilon)2^k \ln 2$ ),  $E[f_x]$  is constantly  $o(1)$  (in fact, exponentially small in  $n$ ). Therefore a more crude analysis will suffice (Proposition 2.42).

In this section we consider a slight modification of  $\mathcal{P}_{n,p}$ . Instead of choosing  $m$  clauses u.a.r., we choose  $m$  clauses with repetitions. However, for  $m/n = O(1)$ , the expected number of pairs of identical clauses in  $F$  (in the modified model) is  $O(m^2/n^k)$ . Thus, for  $k \geq 3$  this quantity is  $o(1)$ . Therefore, as standard calculations show, every property that holds with probability  $q$  in the modified model holds with probability  $q(1 + O(1))$  in  $\mathcal{P}_{n,p}$ . Somewhat abusing notation, we will denote the modification also by  $\mathcal{P}_{n,p}$ .

Let us start with formulating  $E[f_x]$  in a way which is convenient to work with.

**Lemma 2.40.**

$$E[f_x] \leq \binom{n}{xn} \cdot \left(1 - \frac{1 - (1-x)^k}{2^k - 1}\right)^m.$$

**Proof.** Fix an assignment  $\psi$  at distance  $xn$  from the planted assignment  $\varphi$ . The probability that  $\psi$  also satisfies  $F$  can be calculated in the following manner. Let  $A$  be the set of variables on which both  $\psi$  and  $\varphi$  agree.  $|A| = (1-x)n$ . Consider a random clause  $C$  satisfied by  $\varphi$ ; if all  $k$  variables in that clause fall in  $A$ , then  $C$  is surely satisfied by  $\psi$ . The probability for that is  $q = \binom{(1-x)n}{k} / \binom{n}{k}$ . If at least one variable falls out of  $A$ , which happens with probability  $1-q$ , then the clause is satisfied only with probability  $\frac{2^k-2}{2^k-1}$ . This is because there is one way to complement the variables which is not consistent with  $\psi$  but is consistent with  $\varphi$ . There are  $\binom{n}{xn}$  ways to fix  $\psi$ , and therefore

$$E[f_x] = \binom{n}{xn} \left( q \cdot 1 + (1-q) \cdot \frac{2^k-2}{2^k-1} \right)^m = \binom{n}{xn} \left( \frac{2^k-2+q}{2^k-1} \right)^m = \binom{n}{xn} \left( 1 - \frac{1-q}{2^k-1} \right)^m.$$

Finally, observing that  $q \leq (1-x)^k$  proves the lemma.  $\blacksquare$

**Remark 2.41.** Although Lemma 2.40 establishes an upper bound on  $E[f_x]$ , in fact for  $x$  bounded away from 0 equality holds (up to a  $o(1)$  additive factor inside the parenthesis). Also, the bound on  $E[f_x]$  (and therefore also the bound that we derive on  $E[u_x]$ ) is monotonically decreasing and continuous in  $m/n$ .

It will be more convenient to work with the following quantity:

$$f^*(x) \equiv \frac{\ln E[f_x]}{n}. \quad (2.6)$$

One can verify that

$$f^*(x) \leq H(x) \ln 2 + c \ln \left( 1 - \frac{1 - (1-x)^k}{2^k - 1} \right), \quad (2.7)$$

where  $H(x)$  denotes the binary entropy measure,

$$H(x) = -(1-x) \log_2(1-x) - x \log_2 x,$$

and  $c = m/n = (1+\varepsilon)2^k \ln 2$ .

To make use of Proposition 2.38 we need to obtain tight bounds on  $C_2$  and  $E[f_x]$ . In terms of  $f^*(x)$ ,  $E[f_x] = e^{f^*(x)n}$ , therefore to prove  $E[f_x] = o(1)$  it suffices to prove  $f^*(x) < 0$ . This is exactly what the following two propositions formally establish.

**Proposition 2.42.** *For any  $k \geq 20$ ,  $\varepsilon \geq 0.99^k$  and  $x \in [1/k, 1]$ ,*

$$f^*(x) \leq -50k2^{-k}$$

**Proof.** Throughout, we use the following useful upper bound on  $\log(1-x)$  (for any log base).

$$\log(1-x) \leq -x.$$

We break the interval  $[1/k, 1]$  into two subintervals. Let us first consider  $x \in [0.3, 1]$ . Always  $H(x) \ln 2 \leq \ln 2$ , and on the other hand, using  $\log(1-x) \leq -x$ ,

$$c \ln \left( 1 - \frac{1 - (1-x)^k}{2^k - 1} \right) \leq -\frac{(1+\varepsilon)2^k \ln 2}{2^k - 1} (1 - (1-x)^k).$$

Therefore it suffices to prove that  $(1+\varepsilon)(1 - (1-x)^k) \geq 1 + (50k2^{-k}/\ln 2)$  for every  $x \in [0.3, 1]$ . Indeed,

$$(1 - (1-x)^k) \geq (1 - 0.7^k), \quad (1+\varepsilon) \geq (1 + 0.99^k).$$

One can verify that for  $k \geq 20$ , multiplying these two quantities is always greater than  $1 + (50k2^{-k}/\ln 2)$ .

Let us now move the case  $x \in [1/k, 0.3]$ .  $H(x)$  is monotonically increasing until  $x = 0.5$ , therefore it takes its maximal value in this interval at  $x = 0.3$ , which gives  $H(0.3) \leq 0.266$ . On the other hand  $(1 - (1-x)^k)$  takes its minimal value at  $1/k$ . Observe that  $(1 - 1/k)^k \leq e^{-1}$ , and therefore

$$(1 - (1-x)^k) \geq 1 - 1/e \geq 0.6 > 0.266 > H(0.3).$$

In this case we have  $a \leq 0.266 - 0.6 \leq -0.3 < 50k2^{-k}$  for every  $k \geq 20$ . ■

**Proposition 2.43.** *For any  $k \geq 20$ ,  $\varepsilon \geq 0$  and  $\lambda \in [20, 2^k/k]$ , if  $x = \lambda 2^{-k}$  then  $f^*(x) \leq -\lambda 2^{-k}$ .*

**Proof.** For any  $x$ , we have

$$\log(1-x) \leq -x,$$

and, for  $0 \leq x \leq 1$ ,

$$1 - (1-x)^k \geq kx - \frac{k^2 x^2}{2}.$$

Thus,

$$\begin{aligned} H(x) \ln 2 + c \ln \left( 1 - \frac{1 - (1-x)^k}{2^k - 1} \right) &= -x \ln x - (1-x) \ln(1-x) + (1+\varepsilon)2^k (\ln 2) \ln \left( 1 - \frac{1 - (1-x)^k}{2^k - 1} \right) \\ &\leq -x \ln x + x(1-x) - (1+\varepsilon)2^k (\ln 2) \left( \frac{1 - (1-x)^k}{2^k - 1} \right) \\ &\leq -x \ln x + x - (1+\varepsilon)(\ln 2) \left( kx - \frac{k^2 x^2}{2} \right). \end{aligned}$$

Substituting  $\lambda 2^{-k}$  for  $x$ , this upper-bound becomes

$$\begin{aligned} &-x \ln x + x - (1+\varepsilon)(\ln 2) \left( kx - \frac{k^2 x^2}{2} \right) \\ &= \lambda 2^{-k} (k(\ln 2) - \ln \lambda) + \lambda 2^{-k} - (1+\varepsilon)(\ln 2) (k\lambda 2^{-k} - k^2 \lambda^2 2^{-2k-1}) \\ &= -(\lambda \ln \lambda) 2^{-k} + \lambda 2^{-k} - \varepsilon (\ln 2) (k\lambda 2^{-k} - k^2 \lambda^2 2^{-2k-1}) + (\ln 2) k^2 \lambda^2 2^{-2k-1} \\ &= -\lambda 2^{-k} \left( (\ln \lambda) - 1 + \varepsilon (\ln 2) (k - k^2 \lambda 2^{-k-1}) - (\ln 2) k^2 \lambda 2^{-k-1} \right) \end{aligned}$$

$$= -\lambda 2^{-k} \left( (\ln \lambda) \left( 1 - (\ln 2) k^2 \frac{\lambda}{\ln \lambda} 2^{-k-1} \right) - 1 + \left( \varepsilon (\ln 2) \left( k - k^2 \lambda 2^{-k-1} \right) \right) \right).$$

Observe that  $\lambda \leq 2^k/k$  and thus,

$$k - k^2 \lambda 2^{-k-1} \geq 0,$$

and since  $\varepsilon \geq 0$  it suffices to prove that

$$(\ln \lambda) \left( 1 - (\ln 2) k^2 \frac{\lambda}{\ln \lambda} 2^{-k-1} \right) - 1 \geq 1.$$

Since  $\lambda \leq 2^k/k$ , and  $k \geq 5$ , we have

$$(\ln 2) k^2 \frac{\lambda}{\ln \lambda} 2^{-k-1} \leq (\ln 2) k^2 \frac{2^k/k}{k(\ln 2) - \ln k} 2^{-k-1} = (\ln 2) \frac{1}{2((\ln 2) - (\ln k)/k)} \leq 0.65,$$

and so it suffices to verify that

$$\ln \lambda \geq 2/(1 - 0.65),$$

which is always true for  $\lambda \in [20, 2^k/k]$  (for  $k \geq 20$ ,  $2^k/k \gg 20$ ). ■

### 2.8.3 Proof of Theorem 2.1

Recall Proposition 2.38 and Lemma 2.39 which establish together

$$E[u_x] \leq C_2 \cdot E[f_x]/2.$$

$C_2$  is the expected number of satisfying assignment in the planted model,  $C_2 = \sum_x E[f_x]$ .

The idea of the proof is to use Propositions 2.42 and 2.43 to upper bound  $C_2$  by looking at the largest  $x$  s.t.  $E[f_x]$  contributes to  $C_2$  (that is,  $E[f_x]$  is not vanishing with  $n$ ). We shall use  $x_0$  to denote this number (regardless, observe that  $x_0$  is an upper bound on the diameter of the cluster region in the *planted* setting). Then, to beat  $C_2$ , we take  $x_1 > x_0$ , so that for every  $x \geq x_1$ ,  $E[f_x] \cdot C_2 \ll 1$ . Respectively,  $x_1$  upper bounds the diameter of the cluster region in the *uniform* setting. It turns out that  $x_1/x_0 = O(k)$ , and since  $x_0$  scales down with  $2^{-k}$ , this additional factor is manageable.

Formally, propositions 2.42 and 2.43 assert that only  $x \leq 20 \cdot 2^{-k}$  may contribute to the value of  $C_2$ . Indeed, take  $x_0 = 20 \cdot 2^{-k}$ , then  $E[f_x] = o(n^{-1})$  for every  $x \geq x_0$ . For  $x \leq x_0$ , the total number of possible assignments (which obviously bounds the expected number of satisfying assignments) at distance  $xn$  from the planted is

$$\binom{n}{xn} \leq \left( \frac{en}{xn} \right)^{xn} \leq e^{(1-\ln x)xn}.$$

This quantity is maximized for  $x \leq x_0$  at  $x_0$ , which gives  $e^{(k \ln 2 + 1)2^{-k}n}$ . Therefore, for sufficiently large  $n$ ,

$$C_2 \leq o(1) + \sum_{x \leq x_0} \binom{n}{xn} \leq n e^{(k \ln 2 + 1)20 \cdot 2^{-k}n} \leq e^{40k2^{-k}n}.$$

Now take  $x_1 = 50k2^{-k}$  (for  $k \geq 20$ ,  $50k \leq 2^k/k$ , which is the maximal  $\lambda$  allowed), applying Propositions 2.42 and 2.43 once more gives that for  $x \geq x_1$ ,

$$E[f_x] \leq e^{-50k2^{-k}n}.$$

In turn, for  $x \geq x_1$

$$E[u_x] \leq C_2 \cdot E[f_x]/2 \leq e^{40k2^{-k}n} \cdot e^{-50k2^{-k}n} = e^{-10k2^{-k}n}.$$

Using Markov's inequality, for  $x \geq x_1$ ,

$$Pr[u_x > 0] \leq e^{-10k2^{-k}n}.$$

Applying the union bound,

$$Pr[\exists x \geq 50k2^{-k}, u_x > 0] \leq n \cdot e^{-10k2^{-k}n} = o(1).$$

## Chapter 3

# Why almost all satisfiable $k$ -CNF instances are easy

In the previous chapter we discussed the structure of the solution space of typical instances from both  $\mathcal{P}_{n,m}$  and  $\mathcal{S}_{n,m}$ . In this chapter we will study the average case complexity of  $\mathcal{S}_{n,m}$ . As we mentioned in the introduction chapter, the complexity of the planted distribution is well understood (at least for the clause-density regimes that interest us).

We prove that for all but a vanishing fraction of satisfiable  $k$ -CNF formulas over  $n$  variables with  $m$  clauses, one can efficiently find a satisfying assignment (whenever  $m/n$  is greater than some constant). This answers an open problem posed by several researchers, including Ben-Sasson et. al. in [10], asking whether one can characterize  $\mathcal{S}_{n,m}$  for  $m/n = o(\log n)$ . Formally we prove,

**Theorem 3.1.** *There exists a deterministic polynomial time algorithm that finds a satisfying assignment for all but  $o(1)$ -fraction of satisfiable  $k$ -CNF formulas with  $m$  clauses over  $n$  variables when  $m/n \geq c2^k$ ,  $c$  a sufficiently large constant.*

**Remark 3.2.** Observe that since  $\mathcal{S}_{n,m}$  is the uniform distribution, then having an algorithm that works *whp* over  $\mathcal{S}_{n,m}$  is equivalent to having an algorithm that solves all but  $o(1)$ -fraction of satisfiable  $k$ -CNF formulas with  $m$  clauses (which is how we chose to formulate Theorem 3.1).

For the sake of clarity of presentation we prove Theorem 3.1 for the case  $k = 3$  and note that the proof readily generalizes to any constant  $k$ . The dependency of  $m/n$  on  $k$  is given by  $c2^k$  where  $c$  is some universal constant. This dependency is not surprising, as the satisfiability threshold itself scales with  $2^k$ .

Our proof of Theorem 3.1 is constructive – that is, we present an algorithm that meets the requirements of Theorem 3.1. In fact, the algorithm that we present is a variation of the algorithm given in [28], which was analyzed for the planted distribution. Thus, another merit of our work is that by presenting new methods for analyzing heuristics on random instances, we can show that algorithmic techniques invented for the “easy” planted model extend to the more canonical uniform setting.

The notion of core which was presented in the previous chapter (Definition 2.11) is crucial for the analysis of the algorithm. One algorithmic ingredient which we didn't discuss in the previous chapter is that of the Majority Vote.

### 3.1 The Majority Vote

For a 3CNF formula  $F$  and a variable  $x$  we let  $N^+(x)$  be the set of clauses in  $F$  in which  $x$  appears positively (namely, as the literal  $x$ ), and  $N^-(x)$  be the set of clauses in which  $x$  appears negatively (that is, as  $\bar{x}$ ). The Majority Vote assignment over  $F$ , which we denote by MAJ, assigns every  $x$  according to the sign of  $|N^+(x)| - |N^-(x)|$  (TRUE if the difference is positive and FALSE otherwise).

To show the usefulness of the Majority Vote in  $\mathcal{S}_{n,m}$  we work our way through  $\mathcal{P}_{n,m}$ , and use the exchange-rate technique (recall Lemma 2.7). Consider  $F$  in  $\mathcal{P}_{n,m}$ , and let  $\varphi$  be its planted assignment. Consider a variable  $x$  whose assignment is w.l.o.g.  $\varphi(x) = TRUE$ . In every clause of  $F$  that contains  $x$ ,  $x$  appears positively with probability  $4/7$ , and negatively with probability  $3/7$ . Therefore in expectation the sign of  $|N^+(x)| - |N^-(x)|$  agrees with  $\varphi(x)$ . More formally, we prove the following fact:

**Lemma 3.3.** *Let  $F$  be distributed according to  $\mathcal{P}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant. Let  $F_{MAJ}$  be a random variable counting the number of variables in  $F$  on which MAJ disagrees with the planted assignment.*

$$Pr[F_{MAJ} \geq e^{-m/(50n)}n] \leq e^{-ne^{-m/(20n)}}.$$

**Proof.** To simplify our proof we consider the following version of the planted distribution,  $\mathcal{P}_{n,p}$ . To generate  $F$ : pick every clause consistent with the planted assignment with probability  $p$ . By setting  $p = m / \binom{n}{3}$ , it follows from standard calculations that  $F$  will have exactly  $m$  clauses with probability  $\Omega(m^{-\frac{1}{2}})$ . It is easily verified that if some property holds with probability  $1 - \alpha$  in  $\mathcal{P}_{n,p}$ , it holds with probability  $1 - O(\alpha \cdot m^{\frac{1}{2}})$  in  $\mathcal{P}_{n,m}$ .

Let  $P_i$  be a random variable which counts the positive appearances of  $x_i$  in  $F$ , and similarly  $N_i$  for negative ones. Since in  $\mathcal{P}_{n,p}$  the clauses are chosen independently (assuming w.l.o.g. that  $\varphi(x) = TRUE$ )

$$P_i \sim Binom\left(4\binom{n}{3}, p\right), \quad N_i \sim Binom\left(3\binom{n}{3}, p\right).$$

Therefore,

$$Pr[MAJ(x_i) \neq \varphi(x_i)] \leq Pr\left[Binom\left(4\binom{n}{3}, p\right) \leq Binom\left(3\binom{n}{3}, p\right)\right].$$

Set  $a = p \cdot (4\binom{n}{3} + 3\binom{n}{3})/2$ , and then

$$Pr[MAJ(x_i) \neq \varphi(x_i)] \leq Pr[P_i \leq a] + Pr[N_i \geq a].$$

To bound  $Pr[P_i \leq a]$ ,  $Pr[N_i \geq a]$  we use the Chernoff bound, and obtain

$$Pr[P_i \leq a] + Pr[N_i \geq a] \leq e^{-m/(300n)}.$$

Using the linearity of expectation,

$$E[F_{MAJ}] = e^{-m/(300n)}n.$$

To obtain concentration around this value, observe that  $F_{MAJ}$  satisfies the Lipschitz condition with difference 3: let  $M = 7\binom{n}{3}$  and  $X_i$  be an indicator random variable which is 1 iff the  $i^{\text{th}}$  clause was selected to the formula ( $i = 1, \dots, M$ ). For every  $i$  and every two assignments  $a = (a_1, \dots, a_{i-1}, a_i, a_{i+1}, \dots, a_M)$  and  $a' = (a_1, \dots, a_{i-1}, a'_i, a_{i+1}, \dots, a_M)$  of values to  $X_1, \dots, X_M$  (that possibly differ on the  $i^{\text{th}}$  coordinate), it holds that  $|f(a) - f(a')| \leq 3$  (every clause contains three variables). Using the method of bounded differences (see e.g. [8]) one can prove that

$$Pr[F_{MAJ} \geq e^{-m/(200n)}n] \leq e^{-ne^{-m/(100n)}}.$$

Respectively, for  $\mathcal{P}_{n,m}$ ,

$$Pr[F_{MAJ} \geq e^{-m/(200n)}n] \leq O(m^{\frac{1}{2}}) \cdot e^{-ne^{-m/(100n)}}.$$

As we said, the interesting regime for us is  $m/n \leq 4 \log n$  (see Remark 2.26), and therefore the  $m^{\frac{1}{2}}$  factor is dominated by the exponentially decreasing part  $e^{-ne^{-m/(100n)}}$ . To conclude,

$$Pr[F_{MAJ} \geq e^{-m/(200n)}n] \leq e^{-ne^{-m/(50n)}}.$$

■

**Proposition 3.4.** *Let  $F$  be distributed according to  $\mathcal{S}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant. Then whp there exists a satisfying assignment  $\varphi$  of  $F$  that differs from MAJ on at most  $e^{-\Theta(m/n)}n$  variables.*

**Proof.** Let  $\mathcal{A}$  be the following property: “there exists no satisfying assignment s.t. MAJ is at distance at most  $e^{-m/(300n)}n$  from it” (by distance we mean the Hamming distance). Using the exchange-rate technique we obtain:

$$\begin{aligned} Pr^{uniform,m}[\mathcal{A}] &\stackrel{\text{Lemma 2.7}}{\leq} e^{ne^{-m/(3n)}} \cdot Pr^{\text{planted},m}[\mathcal{A}] \stackrel{\text{Lemma 3.3}}{\leq} e^{ne^{-m/(3n)}} \cdot e^{-ne^{-m/(50n)}} \\ &= e^{n(e^{-m/(3n)} - e^{-m/(50n)})} = o(1). \end{aligned}$$

The last equality is due to  $e^{-m/(3n)} - e^{-m/(50n)} < 0$ . ■

The next proposition relates the core to the Majority Vote, and is crucial to the analysis of the algorithm.

**Proposition 3.5.** *Let  $F$  be distributed according to  $\mathcal{S}_{n,m}$  with  $m \geq cn$ ,  $c$  a sufficiently large constant. Then whp there exists a satisfying assignment  $\varphi$  s.t. the following two properties hold:*

- MAJ differs from  $\varphi$  on at most  $e^{-\Theta(m/n)}n$  variables
- There exists a core  $\mathcal{H}$  w.r.t.  $\varphi$  as promised in Proposition 2.14.

The proposition follows by noticing that the exchange rate technique used to prove Lemma 3.3 and Proposition 2.14 (the existence whp of a large core in  $\mathcal{S}_{n,m}$ ) refers to the same assignment in the planted setting – the planted assignment. Thus, one can apply the exchange-rate technique on the combined property.

### 3.2 Proof of Theorem 3.1

We start by presenting the algorithm which we claim meets the requirements of Theorem 3.1. The algorithm is basically the one given in [28]. The only difference (besides maybe different constants) is the first step: we use the Majority Vote and not a spectral step. The reason is that analyzing the spectrum of the adjacency matrix of an  $\mathcal{S}_{n,m}$  instance is a complicated task (as the entries of this matrix are not independent 0/1-random-variables). For an example of such analysis (for  $k$ -colorable graphs) the reader is referred to [18].

#### **SAT**( $F$ )

##### Step 1: Majority Vote

1.  $\pi_1 \leftarrow$  Majority Vote over  $F$ .

##### Step 2: Reassignment

2. **for**  $i = 1$  **to**  $\log n$

3.   **for all**  $x \in V$

4.     **if**  $x$  supports fewer than  $m/(8n)$  clauses w.r.t.  $\pi_i$  **then**  $\pi_{i+1} \leftarrow \pi_i$  with  $x$  flipped.

5.   **end for.**

6. **end for.**

##### Step 3: Unassignment

7. **set**  $\psi_1 = \pi_{\log n}$ ,  $i = 1$ .

8. **while**  $\exists x$  s.t.  $x$  supports fewer than  $m/(10n)$  clauses w.r.t.  $\psi_i$

9.   **set**  $\psi_{i+1} \leftarrow \psi_i$  with  $x$  unassigned.

10.  $i \leftarrow i + 1$ .

11. **end while.**

##### Step 4: Exhaustive Search

12. Let  $\xi$  be the final partial assignment.

13. **let**  $A$  be the set of assigned variables in  $\xi$ .

14. exhaustively search  $F_{out}^\xi(A)$ , component by component.

We now prove that the algorithm SAT meets the requirements of Theorem 3.1. We say that  $F$  is *typical* in  $\mathcal{S}_{n,m}$  if Propositions 2.10, 2.16 and 3.5 hold for it. The discussion in Section 2.3 guarantees that *whp*  $F$  is typical. Therefore, to prove Theorem 3.1 it suffices to consider a typical  $F$  and prove that SAT (always) finds a satisfying assignment for  $F$ .

We let  $\mathcal{H}$  be the core promised in Proposition 2.14, and  $\varphi$  – the satisfying assignment w.r.t. which  $\mathcal{H}$  is defined. In all the following propositions we assume  $F$  is typical (we don't explicitly state it every time for the sake of brevity).

**Proposition 3.6.** *Let  $\psi_1$  be the assignment defined in line 7 of SAT. Then  $\psi_1$  agrees with  $\varphi$  on the assignment of all variables in  $\mathcal{H}$ .*

**Proof.** Let  $B_i$  be the set of core variables whose assignment in  $\psi_i$  disagrees with  $\varphi$  at the beginning of the  $i^{th}$  iteration of the main for-loop – line 2 in SAT. It suffices to prove that  $|B_{i+1}| \leq |B_i|/2$  (if this is true, then after  $\log n$  iterations  $B_{\log n} = \emptyset$ ). Observe that by Proposition 3.5  $|B_0| \leq n/4000$ . By contradiction, assume that not in very iteration  $|B_{i+1}| \leq |B_i|/2$ ,

and let  $j$  be the first iteration violating the inequality  $|B_{j+1}| \geq |B_j|/2$ . Consider a variable  $x \in B_{j+1}$ . If also  $x \in B_j$ , this means that  $x$ 's assignment was not flipped in the  $j^{\text{th}}$  iteration, and therefore,  $x$  supports at least  $m/(8n)$  clauses w.r.t.  $\psi_j$ . By the second item in the definition of a core, at least  $m/(8n) - m/(10n) = m/(40n)$  of these clauses contain only core variables. Since the literal of  $x$  is true in all these clauses, but in fact should be false under  $\varphi$ , each such clause must contain another variable on which  $\varphi$  and  $\psi_j$  disagree, that is another variable from  $B_j$ . If  $x \notin B_j$ , this means that  $x$ 's assignment was flipped in the  $j^{\text{th}}$  iteration. This is because  $x$  supports fewer than  $m/(8n)$  clauses w.r.t.  $\psi_j$ . Since  $x$  supports at least  $m/(6n)$  clauses w.r.t.  $\varphi$ , it must be that in at least  $m/(6n) - m/(8n) = m/(24n)$  of them, the literal of some other core variable evaluates to true (rather than false, as it should be w.r.t.  $\varphi$ ). For conclusion, let  $U = B_j \cup B_{j+1}$ . Then there are at least  $m/(40n) \cdot |B_{j+1}|$  clauses containing at least two variables from  $U$ . Now if  $|B_{j+1}| \geq |B_j|/2$ , then  $m/(40n) \cdot |B_{j+1}| \geq m/(80n) \cdot |U|$ , contradicting Proposition 2.10. ■

**Proposition 3.7.** *Let  $\xi$  be the partial assignment defined in line 12 of SAT. Then all assigned variables in  $\xi$  are assigned according to  $\varphi$ , and all the variables in  $\mathcal{H}$  are assigned.*

**Proof.** By the definition of core – the Majority Vote assignment sets the core variables correctly in  $\psi_1$  (the assignment defined in line 2 in the algorithm SAT) – that is according to  $\varphi$  ( $\varphi$  is the satisfying assignment w.r.t. which  $\mathcal{H}$  is defined). Furthermore, by the definition of core, every core variable supports at least  $m/(5n)$  clauses w.r.t.  $\varphi$ , and also w.r.t.  $\psi_1$  (the assignment at hand before the unassignment step begins). Hence all core variables survive the first round of unassignment. By induction it follows that the core variables survive all rounds. Now suppose by contradiction that not all assigned variables are assigned according to  $\varphi$  when the unassignment step ends. Let  $U$  be the set of variables that remain assigned when the unassignment step ends, and whose assignment disagrees with  $\varphi$ . Every  $x \in U$  supports at least  $m/(10n)$  clauses w.r.t. to  $\xi$  (the partial assignment defined in line 7 of SAT), but each such clause must contain another variable on which  $\psi$  and  $\varphi$  disagree (since the clause is satisfied by  $\varphi$ , and  $\varphi$  falsifies the literal of  $x$  in each such clause). Thus, we have  $|U| \cdot \frac{m}{10n}$  clauses each containing at least two variables from  $U$ . Since  $U \cap \mathcal{H} = \emptyset$  (by the first part of this argument) and  $|\mathcal{H}| \geq (1 - e^{-\Theta(m/n)})n$  (by Proposition 3.5) it follows that  $|U| \leq e^{-\Theta(m/n)}n < n/4000$ , contradicting Proposition 2.10. ■

**Proposition 3.8.** *The exhaustive search, Step 4 of SAT, completes in polynomial time with a satisfying assignment of  $F$ .*

**Proof.** By Proposition 3.7, the partial assignment at the beginning of the exhaustive search step is partial to some satisfying assignment of the entire formula. Therefore the exhaustive search will succeed. Further observe that the unassigned variables are a subset of the non-core variables (Proposition 3.7). Proposition 2.16 then guarantees that the running time of the exhaustive search will be at most polynomial. ■

Theorem 3.1 follows from Propositions 3.6–3.8, and the discussion in Chapter 2 which guarantees that *whp*  $F$  is typical (in the sense that we described above).



## Chapter 4

# The Support Paradigm

In the previous two chapters we saw that the notion of support (Definition 2.4) plays an important role in the definition of a core (Definition 2.11), and also in the analysis of the algorithm in Chapter 3. As it turns out the notion of support plays an important role in other settings as well. A rather similar notion of core was defined in [4] to establish the existence of frozen variables for random  $k$ -CNF formulas below the threshold. Another rather surprising motivation for this work comes from recent experimental results [50, 9] showing that some simple variants of the well known RWalkSAT algorithm [46], which base their greedy rule on the support (although the notion of support is not referred to explicitly in any of these works), seem to be effective for solving random 3SAT formulas in the “hard” near-threshold regime. Specifically, the experimental results suggest that these algorithms may be *whp* efficient in finding satisfying assignments for random 3SAT instances in  $\mathcal{Q}_{n,m}$  with  $m/n$  as large as 4.21 (the conjectured satisfiability threshold for 3SAT is roughly 4.26). In contrast, the “original” RWalkSAT heuristic, which does not consider the support, seems to require typically super-polynomial time already for  $m/n = 2.65$  [47].

Motivated by a search of a unifying rule that may contribute to the understanding of this phenomenon we define the Support Paradigm. Specifically, our focus in this work is on heuristics that obey the following general template: start at some assignment to the variables, then iteratively, using some predefined (greedy) rule, try to minimize the number of unsatisfied clauses (or the distance from some satisfying assignment) until a satisfying assignment is reached. We say that such a heuristic is part of the **Support Paradigm** if the greedy rule uses the support as its main criterion.

We present a new algorithm which is part of the Support Paradigm and rigorously show its effectiveness for the planted  $k$ -SAT distribution with clause-variable ratio greater than some constant. Our results are thus in line with the experimentally-observed advantage of the algorithms in [50, 9] over RWalkSAT.

To keep the presentation simple we shall confine ourselves to the “canonical” case  $k = 3$ , and just point out that our result extends to any fixed  $k$ .

One disclaimer is due before we proceed. We do not claim that our result provides a direct explanation as to why certain algorithms seem to perform well in the below-threshold regime. For one, we deal with higher densities in which the instances are typically more structured.

The notion of support also has a constructive interpretation when considering things from the statistical-mechanics point of view. In this discipline, the combinatorial object 3CNF is a diluted 3-spin spin glass system. Every assignment to the variables corresponds to an energy level of the system, where the free energy of the system in a certain state is the number of clauses not satisfied by the given assignment. Thus, the question of whether the 3CNF is satisfiable or not is equivalent to the question whether the ground state energy of this diluted 3-spin spin glass system is zero. One of the main theoretical bases, at least from a physical point of view, underlying Survey Propagation [14] is the structure of the energy states of near-threshold random 3CNF formulas.

Having said that, one immediately notices that the notion of support is tightly connected to the notion of free energy. For example, flipping the assignment of the variable with the lowest (maybe zero) support corresponds to making a move which incurs the least increase in the free energy of the system; or, lowering the energy of the system (by flipping the assignment of a variable which appears in at least one unsatisfied clause) corresponds to increasing the number of clauses that belong to the support of some variable, and so forth.

As we proved in Chapter 2, the notion of support also plays a crucial role in explaining the existence of frozen variables, at least for the distributions we consider (the uniform distribution over satisfiable  $k$ -CNF instances –  $\mathcal{S}_{n,m}$ , and the planted distribution  $\mathcal{P}_{n,m}$ ). For example, if a variable has zero-support with respect to some satisfying assignment then it cannot be frozen – flip its assignment and the new assignment, which lies in the same cluster, remains satisfying. The other direction is less obvious (that is, what happens when a variable has a large support w.r.t. some satisfying assignment) – and the argument is more involved.

## 4.1 Our Result

We now formally state our result. We state it for 3SAT although it generalizes to  $k$ -SAT for any fixed  $k$ . To somewhat simplify the presentation we consider the following version of the planted distribution which we denote by  $\mathcal{P}_{n,p}$ . To generate  $F$  according to  $\mathcal{P}_{n,p}$ : fix an assignment  $\varphi$ , and pick every clause consistent with  $\varphi$  with probability  $p$ . By setting  $p = m / \binom{n}{3}$ , it follows from standard calculations that  $F$  will have  $m$  clauses with probability  $\Omega(m^{-\frac{1}{2}})$ . It is easily verified that if some property holds with probability  $1 - \alpha$  in  $\mathcal{P}_{n,p}$ , it holds with probability  $1 - O(\alpha \cdot m^{\frac{1}{2}})$  in  $\mathcal{P}_{n,m}$ . Since all our properties hold with very high probability in  $\mathcal{P}_{n,p}$ , they also hold in  $\mathcal{P}_{n,m}$ .

**Theorem 4.1.** *Let  $F$  be a random formula distributed according to  $\mathcal{P}_{n,p}$  with  $n^2p \geq c$ ,  $c$  a sufficiently large constant. Then whp the algorithm  $\text{SupportSAT}(F)$  finds a satisfying assignment of  $F$  using polynomial time.*

The algorithm  $\text{SupportSAT}(F)$ , which belongs to the Support Paradigm, is described in Figure 4.2 in Section 4.2. The proof of Theorem 4.1 also reveals an interesting connection between the notion of support and the notion of frozen variables. Details in Section 4.5.

Combining our result with the work in [5] draws the following interesting picture. We show that  $\text{SupportSAT}$ , which can be viewed as a variation on the classical  $\text{RWalkSAT}$ , succeeds whp in

finding a satisfying assignment for sufficiently dense  $\mathcal{P}_{n,p}$  formulas. For the same clause-density regime it is shown in [5] that RWalkSAT, which disregards the support, fails<sup>1</sup> *whp* to find a satisfying assignment, and not even an assignment which is closer than say  $n/3$  to the planted one. This mirrors nicely the near-threshold picture: experiments predict that RWalkSAT fails to find a satisfying assignment for random 3SAT instances with clause-variable ratio greater than 2.65 [47], while variants of RWalkSAT which take the support into account succeed as far as 4.21 [50, 9].

## 4.2 The Algorithm

We start with a high-level description of our algorithm. Given a formula  $F$  and an assignment  $\psi$  to its variables, we say that the assignment of  $x$  is *suspicious* in  $\psi$  if it supports very few clauses w.r.t.  $\psi$  (we soon quantify “very few” in an exact sense). Our algorithm is composed of two parts. The main part is a simple greedy procedure in which iteratively the assignment of suspicious variables is flipped. From a physical point of view this part can be viewed as a fast cool-down process. When reaching low temperature, a large portion of the formula is already satisfied. If the remaining part of the formula is “simple” then one can find a satisfying assignment for it using some of-the-shelf heuristic.

The fast cool-down process is done mainly via a procedure that we call a **Directed Walk** (inspired by the work of [16]), and then another procedure with a more refined flipping criterion. This corresponds to Steps 1 and 2 in the description of **SupportSAT** below (Figure 4.2). Step 2 typically ends up with an assignment which is very close to a satisfying one. Step 3 completes the job using a simple exhaustive search. As typically the unsatisfied part left at the end of Step 2 is “simple” the exhaustive search takes polynomial time (by typically we mean *whp* over formulas from the planted distribution).

**Remark 4.2.** The reader may wonder at this point if one can do without Step 3 of **SupportSAT**. Namely, can one push the greedy part of the algorithm (Steps 1 and 2) to typically find a satisfying assignment? The answer is probably no, at least not in our planted setting. To see this observe that every variable is expected to appear in  $7\binom{n}{2}p = \Theta(n^2p)$  clauses (which we think of as constant in our analysis). Further, the number of clauses in which a variable appears is binomially distributed. Thus, with constant probability some variables never appear, or appear very scarcely (say once or twice). Therefore in some sense such variables don’t show enough structure to allow a greedy procedure of the sort we use to set them correctly. On the other hand, these variables induce a “simple” formula for which exhaustive search is efficient for example.

### 4.2.1 The Directed Walk

We now introduce the sub-procedure **Directed Walk** which is possibly of its own interest. The input to **Directed Walk** is a 3CNF formula  $F$  and a number  $\varepsilon \in [0, 1]$ .

<sup>1</sup>Throughout when we say that a heuristic fails to produce an optimal solution we always mean that it fails when spending polynomial time (and similarly for success).

**Directed Walk**( $F, \varepsilon$ )

1.  $\psi_0 \leftarrow$  an arbitrary assignment to the variables.
2. **for**  $i = 1$  to  $3/\varepsilon$ 
  - $\psi_i \leftarrow \psi_{i-1}$  with the assignment of the  $\varepsilon n$  variables with the lowest support in  $F$  w.r.t.  $\psi_{i-1}$  – flipped.
3. **return**  $\psi_{3/\varepsilon}$ .

Figure 4.1: Directed Walk

The name “Directed Walk” comes from the fact that as opposed to RWalkSat (‘R’ stands for Random), where the choice of which variables to flip blends randomness, here one employs a decisive deterministic rule, thus the walk is in some sense directed. Directed Walk can be used with other measurements – e.g. flip the assignment of the  $\varepsilon n$  variables whose flipping will gain the maximal number of satisfied clauses, and so forth. In fact, using the last rule with  $\varepsilon = 1/n$  is exactly the algorithm in [36]. Actually, one can generalize Directed Walk to receive the “directing rule” as an argument, and then have a general template (and analysis) for such algorithms. More details in Section 4.4.

We are now ready to present our main algorithm (the notations we use are clarified right after the following figure). In the algorithm we use the parameter  $m$ , where  $m$  satisfies  $m = p \cdot 7 \binom{n}{3}$ . We do this to be consistent with the notations of the algorithm in Figure 3.2.

**Notations.**  $Support_F(x, \psi)$  is the support of a variable  $x$  in  $F$  w.r.t. an assignment  $\psi$ ;  $\psi^{(x)}$  is the assignment  $\psi$  with the assignment of  $x$  flipped. For a formula  $F$  and a subset  $U$  of the variables we denote by  $F[U]$  the subformula containing all clauses with some variable in  $U$ . By *partial assignment* we mean an assignment where some variables may take the value UNASSIGNED. For a partial assignment  $\psi$ ,  $Support_F(x, \psi)$  counts only clauses where all variables are assigned.

The last step of SupportSAT (Step 3) is an exhaustive search. The variables that we want to exhaustively search are those which are still suspicious after the greedy step ends (Steps 1 and 2). To separate the suspicious variables from the reliable ones we employ a careful unassignment step (lines 8-12) which leaves assigned only variables with large support.

### 4.3 Properties of a Random Instance from $\mathcal{P}_{n,p}$

In this section we remind the reader some structural properties of a random instance from the planted distribution. These properties we discussed extensively (and proved) in Section 2. Here we only repeat them briefly (restated in the  $\mathcal{P}_{n,p}$  “terminology”; observe that  $m/n = \Theta(n^2 p)$ ).

The following holds *whp* for random  $F$  in  $\mathcal{P}_{n,p}$  when  $n^2 p \geq c$ ,  $c$  some sufficiently large constant.

**P1.** There exists no set of variables  $U$ ,  $|U| \leq n/10^4$  and there are  $n^2 p |U|/100$  clauses in  $F$  that contain two variables from  $U$ .

<p><b>SupportSAT</b>(<math>F</math>)</p> <p><i>Step 1: Directed Walk</i></p> <ol style="list-style-type: none"> <li>1. <math>\psi_1 \leftarrow \text{Directed Walk}(F, 10^{-5})</math>.</li> </ol> <p><i>Step 2: refining the assignment</i></p> <ol style="list-style-type: none"> <li>2. <b>for</b> <math>i = 1</math> <b>to</b> <math>\log n</math></li> <li>3.   <b>for all</b> <math>x \in V</math></li> <li>4.     <b>if</b> <math>\text{Support}_F(x, \psi_i) \leq m/(8n)</math> <b>then</b> <math>\psi_{i+1} \leftarrow \psi_i^{(x)}</math></li> <li>5.     <b>end for.</b></li> <li>6. <b>end for.</b></li> <li>7. <b>let</b> <math>\tau</math> be the final assignment.</li> </ol> <p><i>Step 3: the exhaustive search</i></p> <ol style="list-style-type: none"> <li>8. <b>set</b> <math>\tau_1 = \tau, j = 1</math>.</li> <li>9. <b>while</b> <math>\exists x</math> s.t. <math>\text{Support}_F(x, \tau_j) \leq m/(10n)</math></li> <li>10.   <b>set</b> <math>\tau_{j+1} \leftarrow \tau_j</math> with <math>x</math> unassigned.</li> <li>11.   <math>j \leftarrow j + 1</math>.</li> <li>12. <b>end while.</b></li> <li>13. <b>let</b> <math>\xi</math> be the final partial assignment.</li> <li>14. <b>let</b> <math>U</math> be the set of unassigned variables in <math>\xi</math>.</li> <li>15. exhaustively search <math>F[U]</math>, separately in every connected component.</li> </ol>
---

Figure 4.2: SupportSAT

**P2.** There exists a core  $\mathcal{H}$ , according to Definition 2.11. This core contains all but  $e^{-\Omega(n^2p)}n$  variables.

**P3.** Let  $F[V \setminus \mathcal{H}]$  be the formula induced by the non-core variables. Then *whp* the graph induced by  $F[V \setminus \mathcal{H}]$  contains no connected component of size greater than  $\log n$ .

#### 4.4 Analysis of the Directed Walk

In this section we analyze a typical execution of Directed Walk for  $\mathcal{P}_{n,p}$ ,  $n^2p$  greater than some sufficiently large constant. Directed Walk, as defined in Figure 4.1, uses the measure of support to determine which variables flip their assignment in every round. Nevertheless, one can use other measures such as the number of unsatisfied clauses in which a variable appears, the number of satisfied clauses gained by flipping the variable, and so on. Our analysis can be easily fit to other measures that satisfy some sufficient conditions which are implicit in Lemma 4.8 (and stated explicitly in Remark 4.10). In fact, our analysis implies the main result in [36]. The following proposition summarizes the main quality of Directed Walk.

**Proposition 4.3.** *Let  $F$  be a random formula distributed according to  $\mathcal{P}_{n,p}$ ,  $n^2p \geq c$ ,  $c$  some sufficiently large constant. Then *whp*  $\text{Directed Walk}(F, 10^{-5})$  enjoys the following property: after at most  $3 \cdot 10^5$  rounds the output assignment differs from the planted one on at most  $n/10^5$  variables.*

**Remark 4.4.** The constant  $10^5$  is arbitrary. In fact, one can show that *whp* Directed Walk( $F, \varepsilon$ ) finds after  $3/\varepsilon$  rounds an assignment at distance at most  $\varepsilon n$  from the planted one for  $\varepsilon$  as small as  $e^{-\Theta(n^2 p)}$ . This is (up to a constant in the exponent that does not depend on  $n, p$ ) exactly the approximation ratio of the Majority Vote [27, 10]. Therefore, if one considers  $\mathcal{P}_{n,p}$  with  $n^2 p \geq c \log n$  then *whp* Directed Walk finds the planted assignment. Indeed, this is what's implicitly proved in [36], though the directing measure is not the support.

Before proving the proposition we make some further observations.

**Definition 4.5.** (*misleading assignments*) Let  $F$  be a satisfiable 3CNF formula and  $\varphi$  a satisfying assignment of  $F$ . We call an assignment  $\psi$   $k$ -misleading w.r.t.  $\varphi$  if there exists a set of  $2k$  variables  $t_1, \dots, t_k, f_1, \dots, f_k$  s.t. for every  $i, j = 1, \dots, k$ :

- $\varphi(t_i) = \psi(t_i), \varphi(f_i) \neq \psi(f_i)$ ,
- $\text{Support}(t_i, \psi) \leq \text{Support}(f_j, \psi)$ .

**Definition 4.6.** ( $\varepsilon$ -directable) We say that  $F$  is  $\varepsilon$ -directable w.r.t. a satisfying assignment  $\varphi$  of  $F$  if there exists no  $\varepsilon n/3$ -misleading assignment w.r.t.  $\varphi$  at Hamming distance greater than  $\varepsilon n$  from  $\varphi$ .

**Proposition 4.7.** Let  $F$  be a random formula distributed according to  $\mathcal{P}_{n,p}$ ,  $n^2 p \geq c$ ,  $c$  some sufficiently large constant, and let  $\varphi$  be its planted assignment. Then *whp*  $F$  is  $10^{-5}$ -directable w.r.t.  $\varphi$ .

The proof of Proposition 4.7 is given by the following lemma.

**Lemma 4.8.** Fix  $\varepsilon \in (0, 1)$  and let  $F$  be a random formula distributed according to  $\mathcal{P}_{n,p}$ ,  $n^2 p \geq c$ ,  $c$  some sufficiently large constant with  $\varphi$  its planted assignment. Let  $\psi$  be an assignment at distance  $\geq \varepsilon n$  from  $\varphi$ . Then the probability that  $\psi$  is  $\varepsilon n/3$ -misleading w.r.t.  $\varphi$  is at most  $3^{-n}$ .

The union bound then guarantees that *whp* no misleading  $\psi$  exists as there are at most  $2^n$  possible ways to choose  $\psi$ . To prove Lemma 4.8 we need the following easy fact whose proof consists of standard probabilistic arguments.

**Lemma 4.9.** Let  $\beta \in (0, 1)$ . Let  $B_1 = \text{Binom}(p, \binom{N}{2}), B_2 = \text{Binom}(p, \binom{N}{2} - \binom{\beta N}{2})$ . Then  $\Pr[B_1 \leq B_2] \leq e^{-g(\beta)pN^2}$ , where  $g : (0, 1) \rightarrow (0, \infty)$  is a monotonically increasing function.

**Proof.** (Lemma 4.8) Consider some assignment  $\psi$  at distance  $\beta n$  from  $\varphi$ ,  $\beta \geq \varepsilon$ . Let  $t, f$  be two variables s.t.  $\psi(t) = \varphi(t), \psi(f) \neq \varphi(f)$ .  $t$  supports  $\binom{n}{2}$  clauses w.r.t.  $\psi$ , and every one of them could have been included in  $F$  (since  $\varphi$  satisfies all of them). On the other hand,  $f$  supports  $\binom{n}{2}$  clauses w.r.t.  $\psi$ , but clauses where both  $\varphi$  and  $\psi$  agree on the assignment of the other two variables cannot be included in  $\varphi$  as they are not satisfied by  $\varphi$  – and there are  $\binom{(1-\beta)n}{2}$  of them. Therefore we get:

$$\Pr[\text{Sup}(t, \psi) \leq \text{Sup}(f, \psi)] \leq \Pr \left[ \text{Bin} \left( p, \binom{n}{2} \right) \leq \text{Bin} \left( p, \binom{n}{2} - \binom{(1-\beta)n}{2} \right) \right] \leq e^{-g(1-\beta)n^2 p}. \quad (4.1)$$

Further observe that the sets of clauses that any two variables support w.r.t. to some assignment are always disjoint (since the supporting variable is unique by definition). If  $\psi$  is  $k$ -misleading w.r.t.  $\varphi$  then in particular there exist  $k$  pairs of variables  $(t_1, f_1), \dots, (t_k, f_k)$  s.t.  $\text{Support}(t_i, \psi) \leq \text{Support}(f_i, \psi)$ . The probability for this is at most

$$e^{-g(1-\beta)n^2p\cdot\epsilon n/3} \leq e^{-(g(1-\beta)c\cdot\epsilon/3)n} = \left(e^{-g(1-\beta)c\cdot\epsilon/3}\right)^n \leq 12^{-n}.$$

The last inequality is due to  $1 - \beta \leq 1 - \epsilon$  and therefore  $g(1 - \beta) \leq g(1 - \epsilon)$ , where  $\epsilon$  is some fixed number, while  $c$  can be an arbitrarily large. Finally observe that there are at most  $2^n \cdot 2^n = 4^n$  ways to choose the sets of  $t_i$ 's and  $f_j$ 's. The lemma follows by applying the union bound.  $\blacksquare$

**Remark 4.10.** In order for a measure function  $M$  to fit the proof of Proposition 4.3 it suffices for  $M$  to obey Equation (4.1) (maybe with some other function  $g'$ ), and also

$$\Pr[M(t) \leq M(f) | M(t_{i_1}) \leq M(f_{i_1}), \dots, M(t_{i_r}) \leq M(f_{i_r})] \leq \Pr[M(t) \leq M(f)],$$

for every  $r$ -subset of the variables  $(i, r \leq k)$ .

**Proof.** (Proposition 4.3) The proof we give here shares some ideas with the analysis in [16]. We prove that Proposition 4.3 holds with probability 1 for  $F$  s.t.  $F$  is  $10^{-5}$ -directable w.r.t.  $\varphi$ . Since this is the case *whp*, as asserted by Proposition 4.7, Proposition 4.3 follows. For two assignments  $\psi, \varphi$ , define  $T(\psi, \varphi)$  to be the set of variables on which  $\psi$  and  $\varphi$  agree, and  $F(\psi, \varphi)$  the set of variables on which they disagree. Let  $E_\epsilon(\psi)$  be the set of  $\epsilon n$  variables with lowest support w.r.t.  $\psi$ . Observe that if  $\varphi$  is some satisfying assignment of  $F$ , and  $\psi$  is the current assignment that Directed Walk( $F, \epsilon$ ) considers, then the variables in  $T(\psi, \varphi) \cap E_\epsilon(\psi)$  will be “wrongly” flipped. Our goal is then to show that  $|T(\psi, \varphi) \cap E_\epsilon(\psi)|$  cannot be too large.

Set  $\epsilon = 10^{-5}$  (as required by Proposition 4.3). Suppose at first that for every  $\psi$  at distance  $\geq \epsilon n$  from  $\varphi$ ,  $|T(\psi, \varphi) \cap E_\epsilon(\psi)| \leq \epsilon n/3$ . If so, then  $|F(\psi, \varphi) \cap E_\epsilon(\psi)| \geq 2\epsilon n/3$ . Thus in every iteration of Directed Walk the distance from  $\varphi$  is decreased by at least  $2\epsilon n/3 - \epsilon n/3 = \epsilon n/3$ . The initial distance is at most  $n$ ; hence, after at most  $n/(\epsilon n/3) = 3/\epsilon = 3 \cdot 10^5$  rounds an assignment  $\psi'$  at distance at most  $\epsilon n$  from  $\varphi$  is reached.

It remains to prove that the above picture is indeed the case. To this end, consider a “bad” assignment  $\psi$  at distance  $> \epsilon n$  from  $\varphi$  but for which  $|T(\psi, \varphi) \cap E_\epsilon(\psi)| \geq \epsilon n/3$ . This implies that  $|F(\psi, \varphi) \cap E_\epsilon(\psi)| \leq 2\epsilon n/3$ . Since the distance between  $\psi$  and  $\varphi$  is  $\geq \epsilon n$ , it holds that  $|F(\psi, \varphi)| \geq \epsilon n$ . The two last observations imply that  $|F(\psi, \varphi) \setminus E_\epsilon(\psi)| \geq \epsilon n/3$ .

Set  $k = \epsilon n/3$ . Let  $f_1, f_2, \dots, f_k$  be variables in  $F(\psi, \varphi) \setminus E_\epsilon(\psi)$ , and  $t_1, t_2, \dots, t_k$  be variables in  $T(\psi, \varphi) \cap E_\epsilon(\psi)$ . For every  $t_i, f_j$ ,  $\text{Support}(t_i, \psi) \leq \text{Support}(f_j, \psi)$  (by the definition of  $E_\epsilon(\psi)$  and the choice of the  $t_i$ 's and the  $f_j$ 's). However this means that  $\psi$  is  $k$ -misleading w.r.t.  $\varphi$  (as the Hamming distance between  $\psi$  and  $\varphi$  is greater than  $\epsilon n$ ), which contradicts that fact that  $F$  is  $\epsilon$ -directable w.r.t.  $\varphi$ .  $\blacksquare$

## 4.5 Algorithm's Analysis – Proof of Theorem 4.1

We say that a formula  $F$  is *typical* if it satisfies the properties mentioned in Section 4.3, and indeed as we saw a formula is typical *whp* (when  $n^2p$  or  $m/n$  is sufficiently large). Thus proving Theorem 4.1 reduces to proving that **SupportSAT** (always) finds a satisfying assignment in polynomial time for typical formulas. In all the propositions below we assume that  $F$  is typical; we let  $\mathcal{H}$  be the core promised in Proposition 2.14 and  $\varphi$  the planted assignment of  $F$ . The following three properties were proven for the algorithm described in Chapter 3, for completeness we just recap them.

- P1.** Let  $\tau$  be the assignment defined in line 7 of **SupportSAT**. Then  $\tau$  agrees with  $\varphi$  on the assignment of all variables in  $\mathcal{H}$  (Proposition 3.6).
- P2.** Let  $\xi$  be the partial assignment defined in line 13 of **SupportSAT**. Then all assigned variables in  $\xi$  are assigned according to  $\varphi$ , and all the variables in  $\mathcal{H}$  are assigned (Proposition 3.7).
- P3.** The exhaustive search in Step 3 of **SupportSAT** completes in polynomial time with a satisfying assignment of  $F$  (Proposition 3.8).

P1 was proven in Chapter 3 conditioned on the fact that the reassignment step starts with an assignment at distance at most  $n/4000$  from the planted one (which was supplied in that setting by the Majority Vote). In our case, Proposition 4.3 guarantees that property.

Theorem 4.1 then follows from this discussion combined with the discussion in Section 4.4.

Taking a closer look at the proof of Theorem 4.1 it turns out that we actually prove the following:

- The support-based greedy step of **SupportSAT** (Steps 1 and 2) set (almost all) **frozen** variables in  $F$  correctly (that is, according to the planted assignment).
- The exhaustive search completes the assignment of the rest.

The latter implies that the frozen variables embed enough “support structure” to allow a support-based greedy heuristic to set their assignment correctly. This asserts an interesting connection between the clustering phenomenon, the notion of frozen variables and the success of support-based greedy heuristics.

## Chapter 5

# The Message Passing Paradigm: Warning Propagation

In Chapter 2 we discussed the geometry of the solution space of typical instances from  $\mathcal{S}_{n,m}$  and  $\mathcal{P}_{n,m}$ . We proved that such formulas, for  $m/n$  above the threshold, typically have a very degenerated structure of the solution space. We then saw two algorithms that can *whp* solve such formulas (for  $m/n$  greater than some sufficiently large constant). In Chapter 3 we described an algorithm which is an adaptation of the “classical” algorithm for the planted distribution (given by Flaxman [28]). In Chapter 4 we described another algorithm which is part of the Support Paradigm. In this chapter we describe a third algorithmic approach, commonly known as *message passing*. Message passing algorithms were studied in several contexts (not only SAT). One striking application of this approach is the **Survey Propagation** algorithm [14]. It was experimentally shown to be effective in solving “hard”  $k$ -CNF formulas with clause-variable just below the conjectured satisfiability threshold. Unfortunately, no rigorous analysis has yet backed up these experimental observations.

In this chapter we analyze the performance of **Warning Propagation** (WP for brevity), a simple popular message passing algorithm, when applied to random satisfiable formulas generated under the planted distribution. Specifically, we show that the standard way of running message passing algorithms – run message passing until convergence, simplify the formula according to the resulting assignment, and satisfy the remaining subformula (if nonempty), if possible, using a simple “off the shelf” heuristic – works for planted random satisfiable formulas with a sufficiently large yet constant clause-variable ratio. We are not aware of previous rigorous analysis of message passing algorithms for non-trivial SAT distributions.

### 5.1 Warning Propagation

Warning Propagation (WP) is a simple iterative message passing algorithm, and serves as an excellent intuitive introduction to more involved message passing algorithms such as **Belief Propagation** [48] and **Survey Propagation** [14]. These algorithms are based on the *cavity method* in which the messages that a clause (or a variable) receives are meant to reflect a situation in which a “cavity” is formed, namely, the receiving clause (or variable) is no longer part of the

formula. Messages in the WP algorithm can be interpreted as "warnings", telling a clause the values that variables will have if the clause "keeps quiet" and does not announce its wishes, and telling a variable which clauses will not be satisfied if the variable does not commit to satisfying them. We now present the algorithm in a formal way.

Let  $F$  be a CNF formula. For a variable  $x$ , let  $N^+(x)$  be the set of clauses in  $F$  in which  $x$  appears positively (namely, as the literal  $x$ ), and  $N^-(x)$  be the set of clauses in which  $x$  appears negatively. For a clause  $C$ , let  $N^+(C)$  be the set of variables that appear positively in  $C$ , and respectively  $N^-(C)$  for negative ones. There are two types of messages involved in the WP algorithm. The first type is messages that a **variable**  $x_i$  sends to a **clause**  $C_j$  in which it appears:

$$x_i \rightarrow C_j = \sum_{C_k \in N^+(x_i), k \neq j} C_k \rightarrow x_i - \sum_{C_k \in N^-(x_i), k \neq j} C_k \rightarrow x_i.$$

If  $x_i$  appears only in  $C_j$  then we set the message to 0. The intuitive interpretation of this message should be  $x_i$  signals  $C_j$  what is currently its favorable assignment by the other clauses it appears in (a positive message means TRUE, negative one means FALSE and a 0 message means UNASSIGNED). The second type are messages sent from a **clause**  $C_j$  to a **variable**  $x_i$  appearing in  $C_j$ :

$$C_j \rightarrow x_i = \prod_{x_k \in N^+(C_j), k \neq i} I_{<0}(x_k \rightarrow C_j) \prod_{x_k \in N^-(C_j), k \neq i} I_{>0}(x_k \rightarrow C_j),$$

where  $I_{<0}(b)$  is an indicator function which is '1' iff  $b < 0$  (respectively  $I_{>0}$ ). If  $C_j$  contains only  $x_i$  (which cannot be the case in 3CNF formulas) then the message is set to 1.  $C_j \rightarrow x_i = 1$  can be intuitively interpreted as  $C_j$  sending a *warning* to  $x_i$  asking it to commit to satisfying  $C_j$  (as all other literals signaled  $C_j$  that currently they evaluate to FALSE). Lastly, we define the current assignment of a variable  $x_i$  to be

$$B_i = \sum_{C_j \in N^+(x_i)} C_j \rightarrow x_i - \sum_{C_j \in N^-(x_i)} C_j \rightarrow x_i.$$

If  $B_i > 0$  then  $x$  is assigned TRUE, if  $B_i < 0$  then  $x_i$  is assigned FALSE, otherwise  $x_i$  is UNASSIGNED. Assume some order on the clause-variable messages (e.g. the lexicographical order on pairs of the form  $(j, i)$  representing the message  $C_j \rightarrow x_i$ ). Given a vector  $\alpha \in \{0, 1\}^{3m}$  in which every entry is the value of the corresponding  $C_j \rightarrow x_i$  message, a partial assignment  $\psi \in \{TRUE, FALSE, UNASSIGNED\}^n$  can be generated according to the corresponding  $B_i$  values (as previously explained).

### 5.1.1 3SAT and Factor Graphs

Given a 3CNF formula  $F$  on  $n$  variables and  $m$  clauses, the *factor graph* of  $F$ , denoted by  $FG(F)$ , is the following graph representation of  $F$ . The factor graph is a bipartite graph,  $FG(F) = (V_1 \cup V_2, E)$  where  $V_1 = \{x_1, x_2, \dots, x_n\}$  (the set of variables) and  $V_2 = \{C_1, C_2, \dots, C_m\}$  (the set of clauses).  $(x_i, C_j) \in E$  iff  $x_i$  appears in  $C_j$ . For a 3CNF  $F$  with  $m$  clauses it holds that  $\#E = 3m$  (as every clause contains exactly 3 different variables). To make presentation clearer, we denote by  $\#A$  the size of a set  $A$  and by  $|a|$  the absolute value of a real number  $a$ .

It would be convenient to think of the messages in terms of the corresponding factor graph. Every undirected edge  $(x_i, C_j)$  of the factor graph is replaced with 2 anti-parallel directed edges,  $(x_i \rightarrow C_j)$  associated with the message  $x_i \rightarrow C_j$  and respectively the edge  $(C_j \rightarrow x_i)$ .

### 5.1.2 The Warning Propagation Algorithm

Warning Propagation(CNF formula  $F$ ):

1. construct the corresponding factor graph  $FG(F)$ .
2. randomly initialize the clause-variable messages to 0 or 1.
3. repeat until no clause-variable message changed from the previous iteration:
  - 3.a randomly order the edges of  $FG(F)$ .
  - 3.b update all clause-variable messages  $C_j \rightarrow x_i$  according to the random edge order.
4. compute a partial assignment  $\psi$  according to the  $B_i$  messages.
5. return  $\psi$ .

In the above description it seems like no update of variable-clause messages is carried out. This however is implicit in line 3.b. Namely, when evaluating the clause-variable message along the edge  $C \rightarrow x$ ,  $C = (x \vee y \vee z)$ , the variable-clause messages concerning this calculation ( $z, y \rightarrow C$ ) are evaluated on-the-fly using the last updated values  $C_i \rightarrow y$ ,  $C_j \rightarrow z$  (allowing feedback from the same iteration). We allow the algorithm not to end up at all (the clause-variable messages may keep changing every iteration). If the algorithm does return an assignment  $\psi$  then we say that it converged. In practice it is common to limit in advance the number of iterations, and if the algorithm didn't converge by that limit, to return "failure".

## 5.2 Related Work

Currently, the Survey Propagation [14] algorithm experimentally outperforms all known algorithms in finding satisfying assignments to uniformly random 3CNF formulas with clause-variable ratio  $\rho$  close to the satisfiability threshold ( $4 \leq \rho \leq 4.25$ ). However, theoretical understanding of Survey Propagation and other message passing algorithms for random SAT problems is still lacking. This should be compared with the success of message passing algorithms for decoding low-density-parity-check (LDPC) codes [31]. Here, the experimental success of message passing algorithms [31] was recently complemented rigorously by a large body of theoretical work, see e.g. [42, 49, 43]. Some important insights emerge from this theoretical work. In particular, it is shown that the quality of decoding improves exponentially with the number of iterations – thus all but a small constant fraction of the received codeword can be decoded correctly using a constant number of iterations. Our analysis of WP on  $\mathcal{P}_{n,p}$  shows that much of the coding picture is valid also for  $\mathcal{P}_{n,p}$  thus providing important insights as to the success of message passing algorithms for random satisfiability problems. The planted 3SAT model is similar to LDPC in many ways. Both constructions are based on random factor graphs. In codes, the received corrupted codeword provides noisy information on a single bit or

on the parity of a small number of bits of the original codeword. In  $\mathcal{P}_{n,p}$ ,  $\varphi$  being the planted assignment, the clauses containing a variable  $x_i$  contain noisy information on the polarity of  $\varphi(x_i)$  in the following sense – each clause contains  $x_i$  in a polarity coinciding with  $\varphi(x_i)$  with probability  $4/7$ . The SAT setting is however more involved than its coding counterpart; for example a SAT instance may have many satisfying assignments (which is *whp* the case in  $\mathcal{P}_{n,p}$  with clause-variable ratio of order  $o(\log n)$ ) whereas a transmitted codeword has a unique decoding. More discussion follows in Section 5.3.

As for relevant results in random graph theory, the seminal work of Alon and Kahale [6] paved the road towards dealing with large-constant-degree planted distributions. [6] present an algorithm that *whp*  $k$ -colors planted  $k$ -colorable graphs (the distribution of graphs generated by partitioning the  $n$  vertices into  $k$  equally-sized color classes, and including every edge connecting two different color classes with probability  $p$ ; commonly denoted  $G_{n,p,k}$ ) with a sufficiently large constant expected degree. Building upon the techniques introduced in [6], Chen and Frieze [34] present an algorithm that 2-colors large constant degree planted 3-uniform bipartite hypergraphs, and Flaxman [28] presents an algorithm for satisfying large-constant clause-variable ratio planted 3SAT instances.

Though in our analysis we use similar techniques to the aforementioned works, our result is conceptually different in the following sense. In [6, 34, 28] the starting point is the planted distribution, and then one designs an algorithm that works well under this distribution. The algorithm may be designed in such a way that makes its analysis easier. In contrast, our starting point is a given message passing algorithm (WP), and then we ask for which input distributions it works well. We cannot change the algorithm in ways that would simplify the analysis. This is similar in spirit to the work of [5] who showed that RWalkSat works on very sparse uniformly random 3CNF instances (for which other simple heuristics are also known to work), and to the work in [27], where a certain version of the  $k$ -opt heuristic is shown to work on  $\mathcal{P}_{n,p}$ . Another kind of interplay between algorithms and random distributions is involved in the work on the lower end of the satisfiability threshold. Much of it is based on the analysis of simple heuristics, often too simple to be of practical value (e.g., in [15] the pure-literal heuristic is used for very sparse uniformly random 3CNF instances).

Another difference between our work and that of [6, 34, 28] is that unlike the algorithms analyzed in those other papers, WP is a randomized algorithm, a fact which makes its analysis more difficult. We could have simplified our analysis had we changed WP to be deterministic (for example, by initializing all clause-variable messages to 1 in step 2 of the algorithm), but there are good reasons why WP is randomized. For example, it can be shown that (the randomized version of) WP converges with probability 1 on 2CNF formulas that form one cycle of implications, but might not converge if step 4 does not introduce fresh randomness in every iteration of the algorithm (details omitted).

### 5.3 Our Results

We state our results with respect to a somewhat different notion of planted distribution. Fix an assignment  $\varphi$  and include every clause consistent with  $\varphi$  with probability  $p$ . Standard

calculations show that when  $m = p \cdot 7 \binom{n}{3}$  then for the “interesting” properties,  $\mathcal{P}_{n,m}$  and  $\mathcal{P}_{n,p}$  behave similarly.

Given a 3CNF  $F$ , **simplify**  $F$  according to  $\psi$ , when  $\psi$  is a partial assignment, means: in every clause substitute every assigned variable with the value given to it by  $\psi$ . Next, if a clause contains a literal which evaluates to true, remove the clause. Otherwise, remove all literals which evaluate to false. The resulting instance is not necessarily in 3CNF form. Denote by  $F|_\psi$  the 3CNF  $F$  simplified according to  $\psi$ . For a set of variables  $A \subseteq V$ , denote by  $F[A]$  the set of clauses in which all variables belong to  $A$ .

To better understand our results it would be convenient to have the somewhat informal notion of a *simple formula* in mind. We call a 3CNF formula simple, if it can be satisfied using simple well-known heuristics (examples include very sparse random 3CNF formulas which are solvable *whp* using the pure-literal heuristic [15], formulas with small weight terminators – to use the terminology of [5] – solvable *whp* using RWalkSat, etc).

**Theorem 5.1.** *Let  $F$  be a 3CNF formula randomly sampled according to  $\mathcal{P}_{n,p}$ , where  $p \geq d/n^2$ ,  $d$  a sufficiently large constant. Then the following holds whp (the probability taken over the choice of  $F$ , and the random choices in lines 2 and 4 of the WP algorithm). There exists a satisfying assignment  $\varphi^*$  (not necessarily the planted one) such that:*

1.  $WP(F)$  converges after at most  $O(\log n)$  iterations.
2. Let  $\psi$  be the partial assignment returned by  $WP(F)$ , let  $V_A$  denote the variables assigned to either *TRUE* or *FALSE* in  $\psi$ , and  $V_U$  the variables left *UNASSIGNED*. Then for every variable  $x \in V_A$ ,  $\psi(x) = \varphi^*(x)$ . Moreover,  $\#V_A \geq (1 - e^{-\Theta(d)})n$ .
3.  $F|_\psi$  is a simple formula which can be satisfied in time  $O(n)$ .

**Proposition 5.2.** *Let  $F$  be a 3CNF formula randomly sampled according to  $\mathcal{P}_{n,p}$ , where  $p \geq c \log n/n^2$ , with  $c$  a sufficiently large constant, and let  $\varphi$  be its planted assignment. Then whp after at most 2 iterations,  $WP(F)$  converges, and the returned  $\psi$  equals  $\varphi$ .*

It is worth noting that formulas in  $\mathcal{P}_{n,p}$ ,  $n^2 p$  some large constant, are not known to be simple (in the sense that we alluded to above). For example, it is shown in [5] that RWalkSat is very unlikely to hit a satisfying assignment in polynomial time when running on a random  $\mathcal{P}_{n,p}$  instance in the setting of Theorem 5.1.

Comparing our results with the coding setting, the effectiveness of message passing algorithms for amplifying local information in order to decode codes close to channel capacity was recently established in a number of papers, e.g. [42, 49]. Our results are similar in flavor, however the combinatorial analysis provided here allows to recover an assignment satisfying *all* clauses, whereas in the random LDPC codes setting, message passing allows to recover only  $1 - o(1)$  fraction of the codeword correctly. In [43] it is shown that for the erasure channel, all bits may be recovered correctly using a message passing algorithm, however in this case the LDPC code is designed so that message passing works for it. We on the other hand take a well known SAT distribution and analyze the performance of a message passing algorithm on it, without changing either of them to ease-up the analysis.

The remainder of the chapter is structured as follows. Section 5.4 provides an overview that may help the reader follow the more technical parts of the proofs. In Section 5.5 we discuss some properties that a typical instance in  $\mathcal{P}_{n,p}$  possesses. Using these properties, we prove in Section 5.6 Theorem 5.1 and Proposition 5.2. In Section 5.8 we summarize our results and discuss potentially interesting lines for further research.

## 5.4 An overview

Let us first consider some possible fixed points of the Warning Propagation (WP) algorithm. The *trivial* fixed point is the one in which all messages are  $\mathbf{0}$ . One may verify that this is the unique fixed point in some cases when the underlying 3CNF formula is very easy to satisfy, such as when all variables appear only positively, or when every clause contains at least two variables that do not appear in any other clause. A *local maximum* fixed point is one that corresponds to a strict local maximum of the underlying MAX-3SAT instance, namely to an assignment  $\tau$  to the variables in which flipping the truth assignment of any single variable causes the number of satisfied clauses to strictly decrease. The reader may verify that if every clause  $C$  sends a  $\mathbf{1}$  message to a variable if no other variable satisfies  $C$  under  $\tau$ , and a  $\mathbf{0}$  message otherwise, then this is indeed a fixed point of the WP algorithm. Needless to say, the WP algorithm may have other fixed points, and might not converge to a fixed point at all.

Recall the definition of  $\mathcal{P}_{n,p}$ . First a truth assignment  $\varphi$  to the variables  $V = \{x_1, x_2, \dots, x_n\}$  is picked uniformly at random. Next, every clause satisfied by  $\varphi$  is included in the formula with probability  $p$  (in our case  $p \geq d/n^2$ ,  $d$  a sufficiently large constant). There are  $(2^3 - 1) \cdot \binom{n}{3}$  clauses satisfied by  $\varphi$ , hence the expected size of  $F$  is  $p \cdot 7 \cdot \binom{n}{3} = 7dn/6 + o(n)$  (when  $d$  is constant, then this is linear in  $n$ , and therefore such instances are sometimes referred to as *sparse* 3CNF formulas). To simplify the presentation, we assume w.l.o.g. (due to symmetry) that the planted assignment  $\varphi$  is the all-one vector.

To aid intuition, we list some (incorrect) assumptions and analyze the performance of WP on a  $\mathcal{P}_{n,p}$  instance under these assumptions.

1. In expectation, a variable appears in  $4\binom{n}{2}p = 2d + o(1)$  clauses positively, and in  $3d/2 + o(1)$  clauses negatively. Our first assumption is that for every variable, its number of positive and negative appearances is equal to these expectations.
2. We say that a variable *supports* a clause with respect to the planted assignment (which was assumed without loss of generality to be the all  $\mathbf{1}$  assignment) if it appears positively in the clause, and the other variables in the clause appear negatively. Hence the variable is the only one to satisfy the clause under the planted assignment. For every variable in expectation there are roughly  $d/2$  clauses that it supports. Our second assumption is that for every variable, the number of clauses that it supports is equal to this expectation.
3. Recall that in the initialization of the WP algorithm, every clause-variable message  $C \rightarrow x$  is 1 w.p.  $\frac{1}{2}$ , and 0 otherwise. Our third assumption is that for every variable, half the messages it receives from clauses in which it is positive are initialized to 1, and half the messages from clauses in which it is negative are initialized to 1.

4. Recall that in step 3b of WP, clause-variable messages are updated in a random order. Our fourth assumption is that in each iteration of step 3, the updates are based on the values of the other messages from the previous iteration, rather than on the last updated values of the messages (that may correspond either to the previous iteration or the current iteration, depending on the order in which clause-variable messages are visited). Put differently, we assume that in step 3b all clause-variable messages are evaluated in *parallel*.

Observe that under the first two assumptions, the planted assignment is a local maximum of the underlying MAX-3SAT instance. We show that under the third and fourth assumption, WP converges to that local maximum, which is also a fixed point, in two iterations. Recalling the initial messages in our third assumption, the messages that variables send to clauses are all roughly  $(2d - 3d/2)/2 = d/4$ . Following the initialization, in the first iteration of step 3 every clause  $C$  that  $x$  supports will send  $x$  the message 1, and all other messages will be 0. Here we used our fourth assumption. (Without our fourth assumption, WP may run into trouble as follows. The random ordering of the edges in step 3 may place for some variable  $x$  all messages from clauses in which it appears positively before those messages from clauses in which it appears negatively. During the iteration, some of the messages from the positive clauses may change from 1 to 0. Without our fourth assumption, this may at some point cause  $x$  to signal to some clauses a negative rather than positive value.) The set of clause-variable messages as above will become a fixed point and repeat itself in the second iteration of step 3. (For the second iteration, the fourth assumption is no longer needed.) Hence the algorithm will terminate after the second iteration.

Unfortunately, none of the four assumptions that we made are correct. Let us first see to what extent they are violated in the context of Proposition 5.2, namely, when  $d$  is very large, significantly above  $\log n$ . Standard concentration results for independent random variables then imply that the first, second and third assumptions simultaneously hold for all variables, up to small error terms that do not effect the analysis. Our fourth assumption is of course never true, simply because we defined WP differently. This complicates the analysis to some extent and makes the outcome depend on the order chosen in the first iteration of step 3a of the algorithm. However, it can be shown that for most such orders, the algorithm indeed converges to the fixed point that corresponds to the planted assignment.

The more difficult part of our work is the case when  $d$  is constant (though a sufficiently large constant), as in the case of Theorem 5.1. In this case, already our first two assumptions are incorrect. Random fluctuations with respect to expected values will *whp* cause a linear fraction of the variables to appear negatively more often than positively, or not to support any clause (with respect to the planted assignment). In particular, the planted assignment would no longer be a local maximum with respect to the underlying MAX-3SAT instance. Nevertheless, as is known from previous work [28], a large fraction of the variables will behave sufficiently close to expectation so that the planted assignment is a local maximum with respect to these variables. Slightly abusing notation, these set of variables are often called the *core* of the 3CNF formula. Our proof plan is to show that WP does converge, and that the partial assignment in step 4 assigns all core variables their correct planted value. Moreover, for non-core variables, we wish to show that the partial assignment does not make any unrecoverable error – whatever value

it assigns to some of them, it is always possible to assign values to those variables that are left unassigned by the partial assignment so that the input formula is satisfied. The reason why we can expect such a proof plan to succeed is that it is known to work if one obtains a good initial partial assignment by means other than WP, as was already done in [28, 27].

Let us turn now to our third assumption. It too is violated for a linear fraction of the variables, but is nearly satisfied for most variables. This fact marks one point of departure for our work compared to previous work [28, 27]. Our definition of the core variables will no longer depend only on the input formula, but also on the random choice of initialization messages. This adds some technical complexity to our proofs.

The violation of the fourth assumption is perhaps the technical part in which our work is most interesting. It relates to the analysis of WP on factor graphs that contain cycles, which is often a stumbling point when one analyzes message passing algorithms. Recall that when  $d = \Omega(\log n)$  (Proposition 5.2), making the fourth assumption simplifies the proof of convergence of WP. Hence removing this assumption in that case becomes a nuisance. On the other hand, when  $d$  is smaller (as in Theorem 5.1), removing this assumption becomes a necessity. This will become apparent when we analyze convergence of WP on what we call *free cycles*. If messages in step 3b of WP are updated based on the value of other messages in the *previous* iteration (as in our fourth assumption), then the random choice of order in step 3a of WP does not matter, and one can design examples in which the messages in a free cycle never converge. In contrast, if messages in step 3b of WP are updated based on the latest value of other messages, (either from the previous iteration or from the current iteration, whichever one is applicable), in the *random order* of the current iteration, then free cycles converge with probability 1 (as we shall later show).

To complete the proof plan, we still need to show that simplifying the input formula according to the partial assignment returned by WP results in a formula that is satisfiable, and moreover, that a satisfying assignment for this sub-formula can easily be found. The existential part (the sub-formula being satisfiable) will follow from a careful analysis of the partial assignment returned by WP. The algorithmic part (easily finding an assignment that satisfies the sub-formula) is based on the same principles used in [6, 28], showing that the sub-formula breaks into small connected components.

## 5.5 Properties of a Random $\mathcal{P}_{n,p}$ Instance

In the previous chapters we discussed some properties of typical planted instances. However since the algorithm WP itself tosses coins, things become more complicated. One difference from previous chapters is that we incorporate into the notion of core also properties of the algorithm (the random order in which the messages are evaluated). Thus, in this chapter we prove anew all the properties that we require for the analysis, but usually as an elaborated outline of a proof (the fine details can be easily completed by the reader, or can be found in previous chapters). In what follows, for simplicity of presentation, we assume w.l.o.g. that the planted assignment is the all TRUE assignment.

### 5.5.1 Stable Variables

**Definition 5.3.** A variable  $x$  **supports** a clause  $C$  with respect to a partial assignment  $\psi$ , if it is the only variable to satisfy  $C$  under  $\psi$ , and the other two variables are assigned by  $\psi$ .

**Proposition 5.4.** Let  $F$  be a 3CNF formula randomly sampled according to  $\mathcal{P}_{n,p}$ , where  $p \geq d/n^2$ ,  $d$  a sufficiently large constant. Let  $F_{SUPP}$  be a random variable counting the number of variables in  $F$  whose support w.r.t.  $\varphi$  is less than  $d/3$ . Then whp  $F_{SUPP} \leq e^{-\Theta(d)}n$ .

**Proof.** (Outline) The proposition follows from simple concentration arguments. Every variable is expected to support  $\frac{d}{n^2} \cdot \binom{n}{2} = \frac{d}{2} + O(\frac{1}{n})$  clauses, thus using e.g. Chernoff's bound and linearity of expectation, one obtains  $E[F_{SUPP}] \leq e^{-\Theta(d)}n$ . To prove concentration around the expected value one can use the Chernoff bound once more as the support of one variable is independent of the others (since it concerns different clauses which are included independently of each other).  
■

Following the definitions in Section 5.1, given a CNF  $F$  and a variable  $x$ , we let  $N^{++}(x)$  be the set of clauses in  $F$  in which  $x$  appears positively but doesn't support w.r.t.  $\varphi$ . Let  $N^s(x)$  be the set of clause in  $F$  which  $x$  supports w.r.t.  $\varphi$ . Let  $\pi = \pi(F)$  be some ordering of the clause-variable message edges in the factor graph of  $F$ . For an index  $i$  and a literal  $\ell_x$  (by  $\ell_x$  we denote a literal over the variable  $x$ ) let  $\pi^{-i}(\ell_x)$  be the set of clause-variable edges ( $C \rightarrow x$ ) that appear before index  $i$  in the order  $\pi$  and in which  $x$  appears in  $C$  as  $\ell_x$ . For a set of clause-variable edges  $\mathcal{E}$  and a set of clauses  $\mathcal{C}$  we denote by  $\mathcal{E} \cap \mathcal{C}$  the subset of edges containing a clause from  $\mathcal{C}$  as one endpoint.

**Definition 5.5.** A variable  $x$  is **stable** in  $F$  w.r.t. an edge order  $\pi$  if the following holds for every clause-variable edge  $C \rightarrow x$  (w.l.o.g. assume  $C = (\ell_x \vee \ell_y \vee \ell_z)$ ,  $C \rightarrow x$  is the  $i$ 'th message in  $\pi$ ):

1.  $|\#\pi^{-i}(y) \cap N^{++}(y) - \#\pi^{-i}(\bar{y}) \cap N^-(y)| \leq d/30$ .
2.  $|\#N^{++}(y) - \#N^-(y)| \leq d/30$ .
3.  $\#N^s(y) \geq d/3$

and the same holds for  $z$ .

**Proposition 5.6.** Let  $F$  be a 3CNF formula randomly sampled according to  $\mathcal{P}_{n,p}$ , where  $p \geq d/n^2$ ,  $d$  a sufficiently large constant. Let  $\pi$  be a random ordering of the clause-variable messages, and  $F_{UNSTAB}$  be a random variable counting the number of variables in  $F$  which are not stable. Then whp  $F_{UNSTAB} \leq e^{-\Theta(d)}n$ .

**Proof.** We start by bounding  $E[F_{UNSTAB}]$ . Consider a clause-variable message edge  $C \rightarrow x$  in location  $i$  in  $\pi$ ,  $C = (\ell_x \vee \ell_y \vee \ell_z)$ . Now consider location  $j \leq i$ . The probability of an edge  $C' \rightarrow \bar{y}$  in location  $j$  is  $(3\binom{n}{2}) / (7\binom{n}{3}) = \frac{3}{7n} + O(\frac{1}{n})$  which is exactly the probability of an edge  $C'' \rightarrow y$ ,  $C'' \in N^{++}(y)$ . This implies

$$E[|\#\pi^{-i}(y) \cap N^{++}(y) - \#\pi^{-i}(\bar{y}) \cap N^-(y)|] = 0.$$

If however

$$|\#\pi^{-i}(y) \cap N^{++}(y) - \#\pi^{-i}(\bar{y}) \cap N^{-}(y)| > d/30$$

then at least one of the quantities deviates from its expectation by  $d/60$ .

Look at  $\#\pi^{-i}(y) \cap N^{++}(y)$  – this is the number of success in draws without replacement. It is known that this quantity is more concentrated than the corresponding quantity if the draws were made with replacement [33]. In particular, since the expectation of  $\#\pi^{-i}(y) \cap N^{++}(y)$  is  $O(d)$  it follows from Chernoff's bound that the probability that it deviates from its expectation by more than  $d/60$  is  $e^{-\Theta(d)}$ . A similar statement holds for  $\#\pi^{-i}(\bar{y}) \cap N^{-}(y)$ . Properties (b) and (c) are bounded similarly using concentration results.

The calculations above hold in particular for the first  $5d$  appearances of messages involving  $x$ . As for message  $5d + 1$ , the probability of this message causing  $x$  to become unstable is bounded by the event that  $x$  appears in more than  $5d$  clauses. As  $x$  is expected to appear in  $3.5d$  clauses, the latter event happens w.p.  $e^{-\Theta(d)}$  (again using standard concentration results). To sum up,

$$Pr[x \text{ is unstable}] \leq 5d \cdot e^{-\Theta(d)} + e^{-\Theta(d)} = e^{-\Theta(d)}.$$

The bound on  $E[F_{UNSTAB}]$  follows by linearity of expectation.

We are now left with proving that  $F_{UNSTAB}$  is concentrated around its expectation, we do so using a martingale argument. Define two new random variables,  $F_1$  counting the number of unstable variables  $x$  s.t. there exists a clause  $C$ , containing  $x$ , and another variable  $y$ , s.t.  $y$  appears in more than  $\log n$  clauses, and  $F_2$  to be the unstable variables s.t. in all clauses in which they appear, all the other variables appear in at most  $\log n$  clauses. Observe that  $F_{UNSTAB} = F_1 + F_2$ . To bound  $F_1$ , observe that if  $F_1 \geq 1$ , then in particular this implies that there exists a variable which appears in more than  $\log n$  clauses in  $F$ . This however can be shown not to happen *whp* (since every variable is expected to appear only in  $O(d)$  clauses). To bound  $F_2$  we use a martingale argument in the framework of [8], page 101. We use the clause-exposure martingale (the clause-exposure martingale implicitly includes the random ordering  $\pi$ , since one can think of the following way to generate the random instance – first randomly shuffle all possible clauses, and then toss the coins). The exposure of a new clause  $C$  can change  $F_2$  by at most  $6 \log n$  since every variable in  $C$  appears in at most  $\log n$  clauses, namely with at most  $2 \log n$  other variables that might become (un)stable due to the new clause. The martingale's total variance, to use the terminology in [8], is  $\sigma^2 = \Theta(dn \log^2 n)$ . Using inequality (7.1) in [8] page 101, with  $\alpha = e^{-\Theta(d)} \sqrt{n} / \log n$ , and the fact that  $E[F_2] \leq E[F_{UNSTAB}]$ , concentration around the expectation of  $F_2$  is obtained. ■

Let  $\alpha \in \{0, 1\}^{3\#F}$  be a clause-variable message vector. For a set of clause-variable message edges  $\mathcal{E}$  let  $\mathbf{1}_\alpha(\mathcal{E})$  be the set of edges along which the value is 1 according to  $\alpha$ . For a set of clauses  $\mathcal{C}$ ,  $\mathbf{1}_\alpha(\mathcal{C})$  denotes the set of clause-variable message edges in the factor graph of  $F$  containing a clause from  $\mathcal{C}$  as one endpoint and along which the value is 1 in  $\alpha$ .

**Definition 5.7.** *A variable  $x$  is **violated** by  $\alpha$  in  $\pi$  if there exists a message  $C \rightarrow x$ ,  $C = (\ell_x \vee \ell_y \vee \ell_z)$ , in place  $i$  in  $\pi$  s.t. one of the following holds:*

1.  $|\#\mathbf{1}_\alpha(\pi^{-i}(y) \cap N^{++}(y)) - \#\mathbf{1}_\alpha(\pi^{-i}(\bar{y}) \cap N^{-}(y))| > d/30$

2.  $|\#\mathbf{1}_\alpha(N^{++}(y)) - \#\mathbf{1}_\alpha(N^-(y))| > d/30$
3.  $\#\mathbf{1}_\alpha(N^s(y)) < d/7$ .

Or one of the above holds for  $z$ .

**Proposition 5.8.** *Let  $F$  be as in the setting of Theorem 5.1, and let  $X$  be a set of stable variables w.r.t. an arbitrary ordering  $\pi$ . Let  $\alpha$  be a random clause-variable message vector. Let  $F_{VIO}$  be a random variable counting the number of violated variables in  $X$ . Then,  $F_{VIO} \leq e^{-\Theta(d)}\#X$ .*

**Proof.** As in the proof of Proposition 5.6, we first bound  $E[F_{VIO}]$ , and then prove concentration using a martingale argument. Since the martingale argument is the same as Proposition 5.6 (instead of a clause-exposure martingale, we have a clause-variable message values exposure martingale), we just show how to bound the expectation.

Consider a stable variable  $x$  in  $F$  w.r.t. to an ordering  $\pi$  of the clause-variable messages. Let  $\alpha$  be a random assignment to the clause-variable messages. Consider a clause-variable message edge  $C \rightarrow x$  at location  $i$  in  $\pi$ .  $x$  is stable and therefore

$$|\#\pi^{-i}(y) \cap N^{++}(y) - \#\pi^{-i}(y) \cap N^-(y)| \leq d/30.$$

Since  $\alpha$  is a random assignment

$$E[|\#\mathbf{1}_\alpha(\pi^{-i}(y) \cap N^{++}(y)) - \#\mathbf{1}_\alpha(\pi^{-i}(\bar{y}) \cap N^-(y))|] \leq d/60.$$

If however

$$|\#\mathbf{1}_\alpha(\pi^{-i}(y) \cap N^{++}(y)) - \#\mathbf{1}_\alpha(\pi^{-i}(\bar{y}) \cap N^-(y))| > d/30, \quad (5.1)$$

then at least one of the quantities in (5.1) deviated from its expectation by at least  $(d/30 - d/60)/2$ . Since both quantities are binomially distributed with expectation  $O(d)$ , the probability of the latter happening is  $e^{-\Theta(d)}$ , using standard concentration results. Properties (b) and (c) are bounded similarly using tight concentration results. Using the union bound as in the proof of Proposition 5.6 and the linearity of expectation the bound on the expectation follows.  $\blacksquare$

### 5.5.2 Discrepancy Properties

We prove a proposition very similar to Proposition 2.5, however stated in a way that is more suitable to our setting.

**Proposition 5.9.** *Let  $c > 1$  be an arbitrary constant. Let  $F \in \mathcal{P}_{n,p}$  be as in the setting of Theorem 5.1. Then whp there exists no subset of variables  $U$ , s.t.  $\#U \leq e^{-\Theta(d)}n$  and there are at least  $c\#U$  clauses in  $F$  containing two variables from  $U$ .*

**Proof.** (Outline) For a fixed set  $U$  of variables,  $\#U = k$ , the number of clauses containing two variables from  $U$  is

$$\binom{k}{2}(n-2)2^3 \leq 4k^2n.$$

Each of these clauses is included independently w.p.  $\frac{d}{n^2}$ . Thus, the probability that  $ck$  of them are included is at most

$$\binom{4k^2n}{ck} \left(\frac{d}{n^2}\right)^{ck} \leq \left(\frac{4k^2ne}{ck} \cdot \frac{d}{n^2}\right)^{ck} \leq \left(\frac{12kd}{cn}\right)^{ck}.$$

Using the union bound, the probability there exists a "dense" set  $U$  is at most

$$\sum_{k=2}^{e^{-\Theta(d)}n} \binom{n}{k} \left(\frac{12kd}{cn}\right)^{ck} = O(d^{2c}/n^{2c-2}).$$

The last equality is obtained using standard calculations. ■

### 5.5.3 The Core Variables

The notion of a stable variable is not enough to ensure that the algorithm will set a stable variable according to the planted assignment, as it may happen that a stable variable  $x$  appears in many of its clauses with unstable variables. Thus,  $x$  can be biased in the wrong direction (by wrong we mean disagreeing with the planted assignment). However, if most of the clauses in which  $x$  appears contain only stable variables, then this is already a sufficient condition to ensure that  $x$  will be set correctly by the algorithm. The set  $\mathcal{H}$  captures the notion of such variables. Our notion of core here is quite similar to the previous notion we had (see Definition 2.11), but has more ingredients (in particular, the order in which the messages are evaluated).

This time we chose to define a core via an explicit procedure, rather through an abstract definition. There are several ways to define a set of variables with these desired properties, we present one of them.

Formally,  $\mathcal{H} = \mathcal{H}(F, \varphi, \alpha, \pi)$  is constructed using the following iterative procedure:

*Let  $A_1$  be the set of variables whose support w.r.t.  $\varphi$  is at most  $d/3$ .*

*Let  $A_2$  be the set of non-stable variables w.r.t.  $\pi$ .*

*Let  $A_3$  be the set of stable variables w.r.t.  $\pi$  which are violated by  $\alpha$ .*

1. *Set  $H_0 = V \setminus (A_1 \cup A_2 \cup A_3)$ .*

2. *While there exists a variable  $a_i \in H_i$  which supports less than  $d/4$  clauses in  $F[H_i]$  OR appears in more than  $d/30$  clauses not in  $F[H_i]$  define  $H_{i+1} = H_i \setminus \{a_i\}$ .*

3. *Let  $a_m$  be the last variable removed in step 2. Define  $\mathcal{H} = H_{m+1}$ .*

**Proposition 5.10.** *If both  $\alpha$  and  $\pi$  are chosen uniformly at random then whp  $\#\mathcal{H} \geq (1 - e^{-\Theta(d)})n$ .*

**Proof.** Let  $\bar{\mathcal{H}} = V \setminus \mathcal{H}$ . Set  $\delta = e^{-\Theta(d)}$ . Partition the variables in  $\bar{\mathcal{H}}$  into variables that belong to  $A_1 \cup A_2 \cup A_3$ , and variables that were removed in the iterative step,  $\bar{H}^{it} = H_0 \setminus \mathcal{H}$ . If

$\#\bar{\mathcal{H}} \geq \delta n$ , then at least one of  $A_1 \cup A_2 \cup A_3$ ,  $\bar{H}^{it}$  has cardinality at least  $\delta n/2$ . Consequently,

$$Pr[\#\bar{\mathcal{H}} \geq \delta n] \leq \underbrace{Pr[\#A_1 \cup A_2 \cup A_3 \geq \delta n/2]}_{(a)} + \underbrace{Pr[\#\bar{H}^{it} \geq \delta n/2 \mid \#A_1 \cup A_2 \cup A_3 \leq \delta n/2]}_{(b)}.$$

Propositions 5.4, 5.6, and 5.8 and Azuma's inequality for example are used to bound (a). To bound (b), observe that every variable that is removed in iteration  $i$  of the iterative step (step 2), supports at least  $(d/3 - d/4) = d/12$  clauses in which at least another variable belongs to  $\{a_1, a_2, \dots, a_{i-1}\} \cup A_1 \cup A_2 \cup A_3$ , or appears in  $d/30$  clauses each containing at least one of the latter variables. Consider iteration  $\delta n/2$ . Assuming  $\#A_1 \cup A_2 \cup A_3 \leq \delta n/2$ , by the end of this iteration there exists a set containing at most  $\delta n$  variables, and there are at least  $d/30 \cdot \delta n/2 \cdot 1/3$  clauses containing at least two variables from it (we divide by 3 as every clause might have been counted 3 times). Plugging  $c = d/180$  in Proposition 5.9, (b) is bounded. ■

### 5.5.4 The Factor Graph of the Non-Core Variables

Proposition 5.10 implies that for  $p = c \log n/n^2$ ,  $c$  a sufficiently large constant, *whp*  $\mathcal{H}$  contains already all variables. Therefore the following propositions are relevant for the setting of Theorem 5.1 (namely,  $p = O(1/n^2)$ ).

**Proposition 5.11.** *Whp every connected component in the factor graph induced by the non-core variables contains  $O(\log n)$  variables.*

This proposition is basically Proposition 2.15, though the notion of core is a bit different (we do not repeat it's proof). Proposition 5.11 will not suffice to prove Theorem 5.1, and we need a further characterization of the non-core factor graph, which is not present in any of the aforementioned works.

**Proposition 5.12.** *Whp every connected component in the factor graph induced by the non-core variables contains at most one cycle.*

**Proposition 5.13.** *The probability of a cycle of length at least  $k$  in the factor graph induced by the non-core variables is at most  $e^{-\Theta(dk)}$ .*

**Corollary 5.14.** *Let  $f = f(n)$  be an arbitrary growing function of  $n$  (namely,  $f(n) \rightarrow \infty$  as  $n \rightarrow \infty$ ). Then whp there is no cycle of length  $f(n)$  in the non-core factor graph*

Since Propositions 5.11-5.13 are all proven using similar arguments, we chose to prove Proposition 5.12 which is not present in any of [6, 28]. The proof is given in Section 5.7. We also show in that section how to change the proof of Proposition 5.12 to fit Proposition 5.13.

## 5.6 Proof of Theorem 5.1 and Proposition 5.2

We start by giving an outline of the proof of Theorem 5.1, while Proposition 5.2 is derived as an easy corollary of that proof. We assume that the formula  $F$  and the execution of WP are *typical* in the sense that Propositions 5.10, 5.11, and 5.12 hold. First we prove that after one iteration WP sets the core variables  $\mathcal{H}$  correctly ( $B_i$  agrees with  $\varphi$  in sign) and this assignment

does not change in later iterations. The proof of this property is rather straightforward from the definition of a core. Therefore from iteration 2 onwards WP is basically running on  $F$  in which variables belonging to  $\mathcal{H}$  are substituted with their planted assignment. This subformula is satisfiable and its factor graph contains small (logarithmic size) connected components, each containing at most one cycle. Consider a connected component composed of a cycle and trees “hanging” on the cycle. Proving convergence on the trees is done using a standard inductive argument. The more involved part is proving convergence on the cycle, and this involves a careful case analysis and arguments similar in flavor to the analysis of a random-walk on a line.

The set  $V_A$  of Theorem 5.1 is composed of all variables from  $\mathcal{H}$  and those variables from the non-core factor graph that get assigned. The set  $V_U$  is composed of the UNASSIGNED variables from non-core factor graph. We now proceed with the formal proof.

### 5.6.1 Analysis of WP on the core factor graph

We start by proving that the messages concerning the factor graph induced by the core-variables converge to the correct value, and remain the same until the end of the execution.

We say that a message  $C \rightarrow x$ ,  $C = (\ell_x \vee \ell_y \vee \ell_z)$ , is *correct* if its value is the same as it is when  $y \rightarrow C$  and  $z \rightarrow C$  are 1 (that is agree in sign with their planted assignment). In other words,  $C \rightarrow x$  is 1 iff  $C = (x \vee \bar{y} \vee \bar{z})$  ( $x$  supports  $C$ ).

**Proposition 5.15.** *If  $x_i \in \mathcal{H}$  and all messages  $C \rightarrow x_i$ ,  $C \in F[\mathcal{H}]$  are correct at the beginning of an iteration (line 3 in the WP algorithm), then this invariant is kept by the end of that iteration.*

**Proof.** By contradiction, let  $C_0 \rightarrow x$  be the first wrongly evaluated message in the iteration. W.l.o.g. assume  $C_0 = (\ell_x \vee \ell_y \vee \ell_z)$ . Then at least one of  $y, z$  sent a wrong message to  $C_0$ .

$$y \rightarrow C_0 = \sum_{C \in N^+(y), C \neq C_0} C \rightarrow y - \sum_{C' \in N^-(y), C' \neq C_0} C' \rightarrow y.$$

Every message  $C'' \rightarrow y$ ,  $C'' \in F[\mathcal{H}] \cap \{N^{++}(y) \cup N^-(y)\}$  is 0 (since it was correct at the beginning of the iteration and that didn't change until evaluating  $C_0 \rightarrow x$ ). On the other hand,  $y \in \mathcal{H}$  and therefore it supports at least  $d/4$  clauses in  $F[\mathcal{H}]$ . Thus at least  $(d/4 - 1)$  messages in the left hand sum are '1' (we subtract 1 as  $y$  might support  $C_0$ ).  $y$  appears in at most  $d/30$  clauses with non-core variables (all of which may contribute a wrong '1' message to the right hand sum). All in all,  $y \rightarrow C_0 \geq (d/4 - d/30 - 1) > d/5$ , which is correct (recall, we assume  $\varphi = \mathbf{1}^n$ ). The same applies for  $z$ , contradicting our assumption. ■

**Proposition 5.16.** *If  $x_i \in \mathcal{H}$  and all messages  $C \rightarrow x_i$ ,  $C \in F[\mathcal{H}]$  are correct by the end of a WP iteration, then  $B_i$  agrees in sign with  $\varphi(x_i)$  by the end of that iteration.*

Proposition 5.16 follows immediately from the definition of  $\mathcal{H}$  and the message  $B_i$ . It suffices to show then that after the first iteration all messages  $C \rightarrow x_i$ ,  $C \in F[\mathcal{H}]$  are correct.

**Proposition 5.17.** *If  $F$  is a typical instance in the setting of Theorem 5.1, then after one iteration of WP( $F$ ), for every variable  $x_i \in \mathcal{H}$ , every message  $C \rightarrow x_i$ ,  $C \in F[\mathcal{H}]$  is correct.*

**Proof.** The proof is by induction on the order of the execution in the first iteration. Consider the first message  $C \rightarrow x$ ,  $C = (\ell_x \vee \ell_y \vee \ell_z)$ ,  $C \in F[\mathcal{H}]$ , to be evaluated in the first iteration. Now consider the message  $y \rightarrow C$  at the time  $C \rightarrow x$  is evaluated. All messages  $C' \rightarrow y$ ,  $C' \in F[\mathcal{H}]$  have their initial random value (as  $C \rightarrow x$  is the first core message to be evaluated). Furthermore,  $y \in \mathcal{H}$ , and therefore there are at most  $d/30$  messages of the form  $C'' \rightarrow y$ ,  $C'' \notin F[\mathcal{H}]$ .  $x \in \mathcal{H}$  hence it is stable w.r.t.  $\pi$  and not violated by the initial clause-variable random messages. Therefore

$$y \rightarrow C \geq \underbrace{d/7}_{\text{property (c) in definition. 5.7}} - \underbrace{d/30}_{\text{property (b) in definition. 5.7}} - \underbrace{d/30}_{\text{non-core messages}} > d/14.$$

The same applies to  $z$ , to show that  $C \rightarrow x$  is correct. Now consider a message  $C \rightarrow x$  at position  $i$ , and assume all core messages up to this point were evaluated correctly. Observe that every core message  $C' \rightarrow y$  that was evaluated already, if  $C' \in \{N^{++}(y) \cup N^-(y)\} \cap F[\mathcal{H}]$  then its value is '0' by the induction hypothesis. Since  $x$  is not violated by  $\alpha$ , property (b) in definition 5.7 ensures that to begin with  $|\#1_\alpha(N^{++}(y)) - \#1_\alpha(N^-(y))| \leq d/30$ .  $y \in \mathcal{H}$ , therefore it appears in at most  $d/30$  non-core messages, all of which could have been already wrongly evaluated, changing the above difference by additional  $d/30$ . As for the core messages of  $y$  which were already evaluated, since they were evaluated correctly, property (a) in definition 5.7 ensures that the above difference changes by at most additional  $d/30$ . All in all, by the time we evaluate  $C \rightarrow x$ ,

$$\sum_{C' \in N^{++}(y), C' \neq C} C' \rightarrow y - \sum_{C'' \in N^-(y), C'' \neq C} C'' \rightarrow y \geq -3 \cdot d/30.$$

As for messages that  $y$  supports, property (c) in definition 5.7 ensures that their contribution is at least  $d/7$  to begin with. Every core message in  $N^s(y)$  that was evaluated turned to '1', every non-core message was already counted in the above difference. Therefore  $y \rightarrow C \geq d/7 - 3 \cdot d/30 > d/25$ . The same applies to  $z$  showing that  $C \rightarrow x$  is correct. ■

To prove Proposition 5.2, observe that when  $p = c \log n/n^2$ , with  $c$  a sufficiently large constant, Proposition 5.10 implies  $\mathcal{H} = V$ . Combing this with Proposition 5.17, Proposition 5.2 readily follows.

### 5.6.2 Analysis of WP on the *non-core* factor graph

It now remains to analyze the behavior of WP on the non-core factor graph, given that the messages involving the core factor graph have converged correctly. A key observation is that once the messages in the factor graph induced by the core variables converged, we can think of WP as if running on the formula resulting from replacing every core variable with its planted assignment and simplifying (which may result in a 1-2-3CNF).

**Proposition 5.18.** *Let  $WP_1(t)$  denote the  $t$ 'th iteration of WP after it has converged on the core. Let  $WP_2(t)$  denote the  $t$ 'th iteration of WP on the formula obtained by replacing each core variable with its planted assignment and simplifying. Then for all variables  $x$  and clauses  $C$  not in the core, the messages  $x \rightarrow C$  and  $C \rightarrow x$  are identical in both algorithms.*

**Proof.** First note that all messages  $x \rightarrow C$ ,  $x \in \mathcal{H}$ , do not change (sign) from the second iteration onwards (by the analysis in the proof of Proposition 5.17). Furthermore, if  $\ell_x$  satisfies  $C$  in  $\varphi$ , then  $x \rightarrow C$  is positive (if  $x$  is a true literal in  $C$ , or negative otherwise), and therefore all messages  $C \rightarrow y$ ,  $y \neq x$  are constantly 0. Namely, they don't effect any calculation, and this is as if we replaced  $\ell_x$  with TRUE, and in the simplification process  $C$  disappeared. If  $\ell_x$  is false in  $C$  under  $\varphi$ , then  $x \rightarrow C$  is constantly negative (if  $\ell_x = x$ , or constantly positive if  $\ell_x = \bar{x}$ ), and this is exactly like having  $\ell_x$  removed from  $C$  (which is the result of the simplification process). ■

Note that to prove the convergence of the algorithm we need also to prove that messages of the sort  $C \rightarrow x$  where  $C$  is not in the core and  $x$  is in the core converge. However, if we prove that all messages in the factor graph induced by the non-core variables converge, then this (with the fact that the core factor graph messages converge) immediately implies the convergence of messages of this type. Therefore, our *goal reduces to proving convergence of WP on the factor graph induced by  $F|_\psi$ , where  $\psi$  assigns the core variables their planted assignment, and the rest are UNASSIGNED.*

We define *weakly correct* as follows. A message  $C \rightarrow x$  is said to be *weakly correct* if  $C \rightarrow x = 1$  then  $x$  is positive in  $C$ .

Consider a connected component in the non-core factor graph consisting of a cycle with trees hanging from it. Our analysis proceeds in three steps:

- We first prove that clause-variable and variable-clause messages of the form  $\alpha \rightarrow \beta$  where  $\alpha \rightarrow \beta$  lead from the trees to the cycle, converge weakly correctly w.r.t. the planted assignment. In the case that the component has no cycles, this concludes the proof.
- Then, using a refined case analysis, we show that the messages along the cycle also converge *whp*, this time not necessarily to the planted assignment, but to some satisfying assignment which agrees with the already converged messages.
- Finally, we conclude by showing that messages from the cycles to the trees converge which implies that all messages in the connected component converge correctly according to some satisfying assignment.

Consider the factor graph  $F$  induced by the simplified formula. A *cycle* in  $F$  is a collection  $x_1, C_2, x_3, C_4, \dots, x_r = x_1$  where  $x_i$  and  $x_{i+2}$  belong to  $C_{i+1}$  for all  $i$  (in our description we consider only odd values of  $i$ ) and  $x_i \neq x_{i+2}$ ,  $C_{i+1} \neq C_{i+3}$  for all  $i$ . A factor graph  $F$  is a *tree* if it contains no cycles. It is *unicyclic* if it contains exactly one cycle. Let  $x \rightarrow C$  be a directed edge of  $F$ . We say that  $x \rightarrow C$  *belongs* to the cycle, if both  $x$  and  $C$  belong to the cycle. For an edge  $x \rightarrow C$  that does not belong to the cycle, we say that  $x \rightarrow C$  *is directed towards* the cycle if  $x$  doesn't belong to the cycle and there exists a directed path from  $C$  to some element of the cycle. We say that the edge  $x \rightarrow C$  *is directed away* from the cycle if  $C$  doesn't belong to the cycle and there exists a directed path originating from the cycle and ending at  $x$ . Similarly we define what it means for an edges  $C \rightarrow x$  to belong to the cycle, to be directed towards the cycle and to be directed away from the cycle.

**Proposition 5.19.** *Let  $F$  be a unicyclic factor graph. Then every directed edge of the form  $x \rightarrow C$  or  $C \rightarrow x$  either belongs to the cycle, is directed towards it or directed away from it.*

**Proof.** Recall that the factor graph is an undirected graph, and the direction is associated with the messages. Take an edge  $x \rightarrow C$  (similarly for  $C \rightarrow x$ ), if it lies on the cycle, then we are done. Otherwise, since the factor graph is connected, consider the path in the tree leading from some element of the cycle to  $C$ . This path is either contained in the path to  $x$  or contains it (otherwise there is another cycle). In the first case  $x \rightarrow C$  is directed towards the cycle, and in the latter  $x \rightarrow C$  is directed away from the cycle. ■

### Convergence of messages directed towards the cycle

**Lemma 5.20.** *Let  $C \rightarrow x$  be an edge in the non-core factor graph belonging to a connected component of size  $s$ . Suppose that the factor graph  $F$  is either a tree or  $F$  is unicyclic and  $C \rightarrow x$  is directed towards the cycle then the message  $C \rightarrow x$  converges after at most  $O(s)$  iterations. Furthermore, if  $C \rightarrow x = 1$  then  $x$  appears positively in  $C$ .*

**Remark 5.21.** Lemma 5.20 immediately implies that all messages  $y \rightarrow C'$  pointing towards the cycle also converge correctly as  $y \rightarrow C'$  depends on messages  $C \rightarrow y$  whose direction is towards the cycle (and they converged by the Lemma).

We also immediately obtain that

**Corollary 5.22.** *Let  $x$  be a variable on the cycle, and  $C \rightarrow x$  an edge directed towards the cycle then the message along the edge  $C \rightarrow x$  converges after at most  $O(s)$  iterations and if after convergence  $C \rightarrow x = 1$  then  $x$  appears positively in  $x$ .*

**Proof.** (Lemma 5.20) We consider the case  $C = (\ell_x \vee \ell_y)$  – the case  $C = (\ell_x \vee \ell_y \vee \ell_z)$  where all three literals belong to non-core variables is proved similarly. Using the terminology of [14], for an edge  $(C, x)$  in the factor graph, we define  $\text{level}(C, x)$  to be  $\infty$  if  $(C, x)$  lies on a cycle, or  $i$  if  $i$  is the maximal length of a path between  $C$  and a leaf in the factor graph from which the edge  $(C, x)$  is removed. The lemma is now proved using induction on  $i$ . Namely, after the  $i$ th iteration, all messages  $C \rightarrow x$  associated with an edge  $(C, x)$  at level  $i$  converge, and if  $C \rightarrow x = 1$  then  $x$  appears positively in  $C$ .

The base case is an edge  $(C, x)$  at level 0. If  $\text{level}(C, x) = 0$  then  $C$  is a unit clause containing only the variable  $x$ . By the definition of the messages, in this case  $C \rightarrow x = 1$  and indeed it must be the case that  $x$  is positive in  $C$  (as the other two variables evaluate to FALSE under the planted). Now consider an edge  $(C, x)$  at level  $i$ , and consider iteration  $i$ . Since  $i > 0$ , it must be that there is another non-core variable  $y$  in  $C$  (or two more variables  $y, z$ ). Consider an edge  $(C_i, y)$ ,  $y \in C_i$  (if no such  $C_i$  exists that we are done as  $C \rightarrow x$  will be constantly 0 in this case).  $y$  cannot be on the cycle (otherwise we found another cycle going through  $x$ ), and therefore by definition the edge  $C_i \rightarrow y$  points in the direction of the cycle. Furthermore,  $\text{level}(C_i, y)$  is strictly smaller than  $i$  since every path from  $C$  to a leaf (when deleting the edge  $(C, x)$ ) passes through some edge  $(C_i, y)$ . By the induction hypothesis, all messages  $C_i \rightarrow y$  already converged, and therefore also  $y \rightarrow C$  and in turn  $C \rightarrow x$ . It is only

left to take care of the case  $C \rightarrow x = 1$ . In this case, there must be a clause  $C_i$  s.t.  $C_i \rightarrow y = 1$  and  $y$  appears positively in  $C_i$  (by the induction hypothesis). If  $C \rightarrow x = 1$  it must be that  $y$  appears negatively in  $C$  and therefore  $x$  must appear positively (otherwise  $C$  is not satisfied by the planted assignment). ■

**Remark 5.23.** The same proof (Lemma 5.20) gives that if the connected component is a tree then all messages  $C \rightarrow x$  converge correctly after  $O(s)$  iterations. In fact a more general folklore fact can be proved using this technique: WP converges on a tree induced by a satisfiable formula, and there exists a satisfying assignment  $\psi$  such that every  $B_i$  is either 0 or agrees with  $\psi$ .

### Convergence of messages on the cycle and the adjacent trees

In the rest of the section we will denote the cycle by  $x_1, C_2, x_3, C_4 \dots x_{2r-1}, C_{2r}, x_1$  where by this we mean that  $x_i$  appears in the clauses before/after it and that  $C_i$  contains the two variables before/after it. We consider two different types of cycles.

- *Biased* cycles: cycles that have at least one warning message  $C \rightarrow x_i = 1$  coming into the cycle, where  $C \rightarrow x_i$  directs into the cycle and the value of  $C \rightarrow x_i$  is the value after the edge has converged.
- *Free* cycles: cycles that do not have such messages coming in, or all messages coming in are 0 messages.

### Convergence of WP when the cycle is biased:

First observe that we may assume w.l.o.g. that every clause on the cycle contains exactly two non-core variables. This is because of the following simple observation (similar to Lemma 5.18). Consider an edge going into the cycle,  $z \rightarrow C$ , and w.l.o.g. assume that  $z$  appears positively in  $C$ . After all the edges going into the cycle have converged, if  $z \rightarrow C \geq 0$  it follows that  $C \rightarrow x = 0$  for cycle edges  $(C, x)$ , and thus execution on the cycle is the same as if  $C$  was removed from the formula, only now we are left with a tree, for which convergence to a correct assignment is guaranteed (Remark 5.23). If  $z \rightarrow C < 0$ , then the execution is exactly as if  $z$  was removed from  $C$  (and  $C$  is in 2-CNF form).

**Proposition 5.24.** *Let  $\mathcal{C}$  be a connected component of the factor graph of size  $s$  containing one cycle s.t. there exists an edge directed into the cycle  $C \rightarrow x_i$  where  $x_i$  belongs to the cycle and such that the message converges to  $C \rightarrow x_i = 1$ . Then WP converges on  $\mathcal{C}$  after at most  $O(s)$  rounds. Moreover for the fixed point, if the message  $C' \rightarrow x = 1$  then  $x$  appears positively in  $C'$ .*

**Proof.** Lemma 5.20 guarantees that after  $O(s)$  iterations all messages directed towards the cycle have converged correctly. We now consider the execution of WP from the time when this has happened. Note that a cycle of the type  $C_j \rightarrow x_{j+1}$  depends only on cycle messages of the type  $C_{j'} \rightarrow x_{j'+1}, x_{j'+1} \rightarrow C_{j'+2}$  and on messages coming into the cycle. In other words during the execution of WP the value of all messages  $C_{j'} \rightarrow x_{j'-1}, x_{j'-1} \rightarrow C_{j'-2}$  do not effect the value of the message  $C_j \rightarrow x_{j+1}$ . Recall that we are in the case where there exists a message  $C \rightarrow x_i = 1$  going into the cycle (after the convergence of these messages). Note that  $x_i$  must appear positively in  $C$ . We consider the following cases:

- There exists a variable  $x_j$  that appears positively in both  $C_{j-1}$  and  $C_{j+1}$  (the case  $j = i$  is allowed here). We note that in this case the message  $x_j \rightarrow C_{j+1}$  must take the value either 0 or 1 which implies that the message  $C_{j+1} \rightarrow x_{j+2}$  converges to the value 0. This in turn implies that the value of all messages  $x_r \rightarrow C_{r+1}$  and  $C_{r+1} \rightarrow x_{r+2}$  for  $r \neq j$  will remain the same if the clause  $C_{j+1}$  is removed from the formula. However, this case reduces to the case of tree formula that was previously analyzed.
- $x_i$  appears negatively in  $C_{i+1}$  and positively in  $C_{i-1}$ . We note that in this case the message  $x_i \rightarrow C_{i+1}$  always take the value 1 which implies that the message  $C_{i+1} \rightarrow x_{i+2}$  always take the value 1. Thus in this case we may remove the clause  $C_{i+1}$  from the formula and replace it by the unit clause  $\ell_y$  where  $C_{i+1} = \ell_y \vee \bar{x}_i$ . Again, this reduces to the case of a tree formula.
- The remaining case is the case where  $x_i$  appears negatively in both  $C_{i-1}$  and  $C_{i+1}$  and there is no  $j$  such that  $x_j$  appears positively in both  $C_{j-1}$  and  $C_{j+1}$ . We claim that this leads to contradiction. Note that by the lemma above there exists a satisfying assignment where  $x_i = 1$ . Write  $C_{i+1} = \bar{x}_i \vee \ell_{i+2}$ . Then for the truth assignment we must have  $\ell_{i+2} = 1$ , similarly  $\ell_{i+4} = 1$  etc. until we reach  $\bar{x}_i = 1$  - a contradiction.

To summarize, the biased cycle case reduces to a tree, and thus after an additional  $O(s)$  iterations it converges in a way that does not disagree with the planted assignment. Lastly, repeating the argument of Lemma 5.20 once again, one proves that after additional  $O(s)$  iterations, all messages concerning non-cycle edges that point away from the cycle converge weakly correctly, proving Proposition 5.24. ■

**Corollary 5.25.** *Assume the conditions of Proposition 5.24. Then for every variable  $x_i$  in  $\mathcal{C}$  it holds that once the messages have converged,  $B_i$  does not disagree with  $\varphi$ .*

The corollary follows immediately by the last statement in Proposition 5.24.

### Convergence of WP when the cycle is free:

**Claim 5.26.** *Let  $\mathcal{C}$  be a connected component of the factor graph of size  $s$  containing one cycle of size  $r$  s.t. the fixed point contains no messages  $C \rightarrow x = 1$  going into the cycle (the cycle is free). Then whp WP converges after at most  $O(r^2 \cdot \log n + s)$  rounds. Moreover for the fixed point, if we simplify the formula which induces  $\mathcal{C}$  according to the resulting  $B_i$ 's, then the resulting subformula is satisfiable.*

**Remark 5.27.** Observe that the free case is the only one where convergence according to the planted assignment is not guaranteed. Furthermore, the free cycle case is the one that may not converge “quickly” (or not at all), though this is extremely unlikely. The proof of Proposition 5.26 is the only place in the analysis where we use the fact that in line 3.a of WP we use fresh randomness in every iteration.

We now prove Proposition 5.26. We divide the analysis into two cases. The easy one is the case in which the cycle contains a pure variable w.r.t the cycle (though this variable may not be pure w.r.t to the entire formula).

**Claim 5.28.** *If the cycle contains a variable  $x_i$  appearing in the same polarity in both  $C_{i+1}, C_{i-1}$ , then the messages  $C \rightarrow x$  along cycle edges converge. Moreover for the fixed point, if  $C \rightarrow x = 1$  then  $x$  satisfies  $C$  according to  $\varphi$ .*

The proof is very similar to the first case in the proof of Proposition 5.24. We omit the details.

We now move to the harder case, in which the cycle contains no pure variables (which is the case responsible for Remark 5.27).

**Proposition 5.29.** *Consider a free cycle of size  $r$  with no pure literal, and one of the two directed cycles of messages. Then the messages along the cycle converge whp to either all 0 or all 1 in  $O(r^2 \log n)$  rounds.*

Convergence *whp* in polynomial time suffices due to Corollary 5.14 (which asserts that *whp* every cycle is of length at most  $\log^\varepsilon n$  for every  $\varepsilon > 0$ ). The proof of Proposition 5.29 is given in the end of this section. We proceed by analyzing WP assuming that Proposition 5.29 holds, which is the case *whp*.

**Proposition 5.30.** *Suppose that the cycle messages have converged (in the setting of Proposition 5.29), then the formula resulting from substituting every  $x_i$  with the value assigned to it by  $B_i$  (according to the fixed point of WP), and simplifying, is satisfiable.*

**Proof.** Let  $F$  be the subformula that induces the connected component  $\mathcal{C}$ , and decompose it according to the trees that hang on the cycle's variables and the trees that hang on the cycle's clauses. Observe that the formulas that induce these trees are variable and clause disjoint (since there is only one cycle in the  $\mathcal{C}$ ).

Consider a tree hanging on a cycle variable  $x_i$ . Let  $C$  be some non-cycle clause that contains  $x_i$ , and  $F_C$  the subformula that induces the tree rooted at  $C$ . Observe that once the cycle has converged, then the message  $x_i \rightarrow C$  does not change anymore. If  $x_i \rightarrow C$  agrees with  $\varphi$  then Remark 5.23 guarantees correct convergence on satisfiable trees. Now consider the case where  $x_i \rightarrow C$  disagrees with  $\varphi$ . Recall that we assume  $\varphi(x_i) = TRUE$ , and therefore  $x \rightarrow C$  is negative in the fixed point. If  $x_i$  appears negatively in  $C$  then  $C \rightarrow y = 0$  for every  $y \in C$  (since  $x_i$  signals  $C$  that it satisfies it), and therefore  $C$  doesn't effect any calculation from this point onwards, and the correct convergence of  $F_C$  is again guaranteed by Remark 5.23 on the convergence for satisfiable trees. The more intricate case is if  $C$  contains  $x_i$  positively. Since we are in the free case, it must hold that  $C \rightarrow x = 0$ . Therefore using Proposition 5.31 below one obtains that  $F_C$  is satisfiable (regardless of the assignment of  $x_i$ ), and WP will converge as required (again Remark 5.23).

Now consider a tree hanging on a cycle clause. Namely,  $C_{i+1} = (x_i \vee x_{i+2} \vee y)$ , where  $x_i, x_{i+2}$  are cycle variables, and  $(C_{i+1}, y)$  is a tree edge. If one of the cycle orientations converged to 0, then  $C_{i+1} \rightarrow y$  converges to 0, and then Remark 5.23 guarantees correct convergence. The same applies to the case where both orientations converge to 1 and  $y$ 's polarity in  $C_{i+1}$  agrees with  $\varphi$ . In this case  $C_{i+1} \rightarrow y$  also converges to 1 since no variable is pure on the cycle, and both  $x_i, x_{i+2}$  signal  $C_{i+1}$  that they will not satisfy it.

The delicate case remains when  $C_{i+1} \rightarrow y$  converges to 1 but  $y$ 's polarity in  $C_{i+1}$  disagrees with  $\varphi$ , that is,  $y$  is negative in  $C_{i+1}$ . The key observation is that the message  $y \rightarrow C_{i+1}$  (which

is directed towards the cycle) must have converged to a positive value (otherwise,  $C_{i+1} \rightarrow x_i$  and  $C_{i+1} \rightarrow x_{i+2}$  would have converged to 0). This means that there is already at least one message  $\hat{C}_1 \rightarrow y$  which converges to 1 in which  $y$  appears positively (and is directed towards the cycle). We decompose the tree hanging on  $y$  to a spine, and trees hanging on this spine. The spine is a path of clauses that caused the warning  $\hat{C}_1 \rightarrow y$  to be issued, and is defined inductively with the following being the base case:  $\hat{C}_1$  must be of the form  $\hat{C}_1 = (\bar{w} \vee \bar{k} \vee y)$  (by the convergence according to the planted in the direction of the cycle, and the fact that  $\hat{C}_1$  issues a warning to  $y$ ); therefore there exist  $\hat{C}_2 = (w \vee \dots)$ ,  $\hat{C}_3 = (k \vee \dots)$  that signal a warning to  $w$  and  $k$  respectively in the direction of the cycle.  $\hat{C}_2, \hat{C}_3$  are the second level of the spine, and so it unfolds.

First we claim that every variable that belongs to a spinal clause sends a non-negative message to the trees that hang on it, this is proven via induction on the distance of the variable from  $y$ ; the base case is distance 0, which is  $y$  itself. The messages that we need to verify are of the form  $y \rightarrow C'$ ,  $C' \neq \hat{C}_1$ , which are pointing away from the cycle (all message pointing in the direction of the cycle have converged correctly, Remark 5.23). In this case  $y \rightarrow C' \geq 0$  ( $y$ 's message agrees with the planted); this is because the wrong message  $C_{i+1} \rightarrow y$  is evened by the correct warning  $\hat{C}_1 \rightarrow y$ , and  $y \rightarrow C'$  depends only on one message which is directed away from the cycle – otherwise there is a second cycle. The induction step follows very similarly. This fact grants correct convergence of the variables in the trees hanging from spinal variables.

As for variables that belong to spinal clauses, for example  $y, w, k$  in the above example, there is always at most one message  $\hat{C}_1 \rightarrow w$  in the direction away from the cycle (otherwise there is more than one cycle); this message may be wrong. Using induction one can show that there is always at least one correct warning  $\hat{C}_2 \rightarrow w$  (in the direction of the cycle), therefore  $B_w \geq 0$ .

Regarding the cycle clauses, the key observation is that setting the cycle variables according to one arbitrary orientation (say, set  $x_i$  to satisfy  $C_{i+1}$ ) satisfies the cycle and doesn't conflict with any satisfying assignment of the hanging trees: if the tree hangs on a variable  $x_i$ , then since the cycle is free, the tree is satisfiable regardless of the assignment of  $x_i$  (Proposition 5.31). In the case that the tree hangs on a cycle-clause  $C$ , then the cycle variables and the tree variables are disjoint, and  $C$  is satisfied already by a cycle-variable regardless of the assignment of the tree-variables. If in the fixed point one cycle orientation is 0 and one orientation is 1, then the  $B_i$  messages of the cycle variables implement exactly this policy. If both cycle orientations converged to 1 or to 0, then the corresponding  $B_i$  messages of all cycle variables are UNASSIGNED (since the cycle is free), but then the same policy can be used to satisfy the clauses of the cycle in a manner consistent with the rest of the formula. ■

**Claim 5.31.** *Let  $(C, x)$  be a tree edge, let  $T_C$  be the subtree rooted at  $C$  and  $F_C$  the formula that induces  $T_C$  (without the variable  $x$  in  $C$ ).  $C \rightarrow x = 0$  in the fixed point if and only if  $F_C$  is satisfiable.*

**Proof.** The structure of the proof is similar to that of Lemma 5.20. Recall that for an edge  $(C, x)$  in the factor graph we define  $\text{level}(C, x)$  to be  $\infty$  if  $(C, x)$  lies on a cycle, or  $t$  if  $t$  is the maximal length of a path between  $C$  and a leaf in the factor graph from which the edge  $(C, x)$  is removed. The lemma is now proved using induction on  $t$ .

The base case is an edge  $(C, x)$  at level 0. If  $\text{level}(C, x) = 0$  then  $C$  is a unit clause containing only the variable  $x$ . Indeed  $C \rightarrow x = 1$  by definition and deleting  $x$  from  $C$  results in an unsatisfiable  $F_C$ .

Now consider an edge  $(C, x)$  at level  $t > 0$ . Observe that every edge  $(C, y)$  in  $T_C$  has level value which is strictly smaller than  $t$  – since every path from  $C$  to a leaf (when deleting the edge  $(C, x)$ ) passes through some edge  $(C_j, y)$ . Assume  $C = (x \vee \ell_y \vee \ell_z)$  (maybe only  $\ell_y$ ). First we prove that if  $F_C$  is not satisfiable then it must be that  $C \rightarrow x = 1$ . If  $F_C$  is not satisfiable then it must be that  $\ell_y = \bar{y}$  and  $\ell_z = \bar{z}$  (otherwise, if one of them is positive then  $\varphi$  satisfies  $F_C$ ). Further, observe that there must exist at least one clause  $C_j$  containing  $y$  positively, and at least one  $C_k$  containing  $z$  positively s.t.  $F_{C_i}$  and  $F_{C_j}$  are unsatisfiable. Otherwise, we can define  $\varphi'$  to be  $\varphi$  except that  $y$  or  $z$  are assigned FALSE (depending which of  $C_i$  or  $C_j$  doesn't exist). It is easy to see that  $\varphi'$  satisfies  $F_C$ , contradicting our assumption. By the induction hypothesis  $C_i \rightarrow y = 1$  and  $C_j \rightarrow z = 1$ . This in turn implies that  $C \rightarrow x_i = 1$  (since  $C_i$  and  $C_j$  contain  $y$  and  $z$  respectively in an opposite polarity to  $C$  and there cannot be any message  $C_k \rightarrow y = 1$  where  $y$  appears negatively in  $C_k$  since  $F_{C_k}$  is satisfiable in this case). Now assume that  $C \rightarrow x = 1$ . The same arguments imply that  $C$  must be of the form  $C = (x \vee \bar{y} \vee \bar{z})$  and there exists  $C_i, C_j$  as above. By the induction hypothesis, if one is to satisfy  $F_C$  it must be that  $y = z = TRUE$  (this is the only way to satisfy  $F_{C_i}$  and  $F_{C_j}$  when inserting  $y$  back to  $C_i$  and  $z$  to  $C_j$ ), but then  $C$  is not satisfied. ■

In Theorem 5.1 the unassigned variables are required to induce a “simple” formula, which is satisfiable in linear time. Observe that the factor graph induced by the UNASSIGNED variables consists of connected components whose structure is a cycle with trees hanging on it, or just a tree. A formula whose factor graph is a tree can be satisfied in linear time by starting with the leaves (which are determined uniquely in case that the leaf is a clause – namely, a unit clause, or if the leaf is a variable then it appears only in one clause, and can be immediately assigned) and proceeding recursively. Regarding the cycle, consider an arbitrary variable  $x$  on the cycle. By assigning  $x$  and simplifying accordingly, we remain with a tree. Since there are only two ways to assign  $x$ , the whole procedure is linear in the size of the connected component. This completes the proof of Theorem 5.1.

### 5.6.3 Proof of Proposition 5.29

Since the cycle has no pure literal it must be of the following form:  $C_1 = (\ell_{x_1} \vee \bar{\ell}_{x_2}), C_2 = (\ell_{x_2} \vee \bar{\ell}_{x_3}), \dots, C_L = (\ell_{x_L} \vee \bar{\ell}_{x_1})$ .

Consider one of the directed cycles, say:  $x_1 \rightarrow C_1 \rightarrow x_2 \rightarrow \dots$  and note that when the message  $x_i \rightarrow C_i$  is updated it obtains the current value of  $C_{i-1} \rightarrow x_i$  and when the message  $C_i \rightarrow x_{i+1}$  is updated, it obtains the current value of  $x_i \rightarrow C_i$ .

It thus suffices to show that the process above converges to all 0 or all 1 in time polynomial in the cycle length. This we prove in the lemma below.

**Claim 5.32.** *Consider the following process on  $\{0, 1\}^{[L]}$ . Given the state of the process  $\Gamma^i$  at round  $i$ , the state  $\Gamma^{i+1}$  at round  $i + 1$  is defined by choosing a permutation  $\sigma \in S_k$  uniformly at random. Then let  $\Delta_0 = \Gamma^i$  and for  $1 \leq j \leq L$ , let  $\Delta_j$  be obtained from  $\Delta_{j-1}$  by setting*

$\Delta_j(\sigma(j-1)) = \Delta_{j-1}((\sigma(j-1)+1) \bmod k)$  and  $\Delta_j(i) = \Delta_{j-1}(i)$  for all  $i \neq \sigma(j-1)$ . Finally, let  $\Gamma^{i+1} = \Delta_L$ .

Let  $T$  be the stopping time where the process hits the state all 0 or all 1. Then

$$\Pr[T \geq 4aL^2] \leq L2^{-a}. \quad (5.2)$$

for all  $a \geq 1$  integer.

The proof of Proposition 5.32 is based on the fact that the process defined in this lemma is a martingale. This is established in the following two lemmas.

**Lemma 5.33.** *Consider the following variant  $\tilde{\Gamma}^i$  of the process  $\Gamma^i$  defined in Proposition 5.32. In the variant, different intervals of 0/1 are assigned different colors and the copying procedure is as above. Fix one color and let  $X_i$  denote the number of elements of the cycle of that color in  $\tilde{\Gamma}^i$ . Then  $X_i$  is a martingale with respect to the filtration defined by  $\tilde{\Gamma}^i$ .*

**Proof.** From the definition of the copying process it is easy to see that

$$E[X_{i+1} | \tilde{\Gamma}^i, \tilde{\Gamma}^{i-1}, \dots] = E[X_{i+1} | X_i].$$

We will show below that  $E[X_{i+1} | X_i] = X_i$  and thus that  $X_i$  is a martingale with respect to the filtration  $\tilde{\Gamma}^i$ .

Assume that  $X_i = k$ . Then w.l.o.g. we may assume that the configuration  $\tilde{\Gamma}^i$  consists of an interval of 1's of length  $k$  and an interval of 0's of length  $L - k$ . We calculate separately the expected shifts in the locations of left end-points of the 0 and 1 interval respectively. We denote the two shift random variables by  $L_0$  and  $L_1$ . Clearly  $L_0 = I_{0,1} + I_{0,2} + \dots + I_{0,k-1}$  where  $I_{0,j}$  is the indicator of the event that 0 left-end point shifted by at least  $j$  and similarly for  $L_1$ . Note that

$$E[I_{0,j}] = \frac{1}{j!} - \frac{1}{(L-k+j)!}$$

and that

$$E[I_{1,j}] = \frac{1}{j!} - \frac{1}{(k+j)!}.$$

The last equation follows from the fact that in order for the 1 interval to extend by at least  $j$ , the  $j$  copying has to take place in the correct order and it is forbidden that they all took place in the right order and the interval has become a 0 interval. The previous equation is derived similarly. Thus

$$E[(X_{i+1} - X_i) | X_i] = E[L_1] - E[L_0] = \sum_{j=1}^{L-k} \left( \frac{1}{j!} - \frac{1}{(k+j)!} \right) - \sum_{j=1}^k \left( \frac{1}{j!} - \frac{1}{(L-k+j)!} \right) = 0$$

This concludes the proof that  $X_i$  is a martingale. The proof follows.  $\blacksquare$

The proof of Proposition 5.32 follows by a union bound from the following lemma where the union is taken over all intervals.

**Lemma 5.34.** *Consider the process  $\tilde{\Gamma}^i$  defined in Lemma 5.33. Fix one interval and let  $X_i$  denote its length. Let  $T$  be the stopping time where  $X_i$  equals either 0 or  $L$ . Then*

$$P[T \geq 4aL^2] \leq 2^{-a}.$$

**Proof.** In order to bound the hitting probability of 0 and  $L$ , we need some bounds on the variance of the martingale differences. In particular, we claim that unless  $k = 0$  or  $k = L$  it holds that

$$E[(X_{t+1} - X_t)^2 | X_t = k] \geq 1/2.$$

If  $k = 1$  or  $k = L - 1$  this follows since with probability at least  $1/2$  the end of the interval will be hit. Otherwise, it is easy to see that the probability that  $X_{t+1} - X_t$  is at least 1 is at least  $1/4$  and similarly the probability that it is at most  $-1$  is at least  $1/4$ . This can be verified by considering the event that one end-points moves by at least 2 and the other one by at most 1.

Let  $T$  be the stopping time when  $X_T$  hits 0 or  $n$ . Then by a Wald kind of calculation we obtain:

$$\begin{aligned} L^2 &\geq E[(X_T - X_0)^2] = E\left[\left(\sum_{t=1}^{\infty} 1(T \geq t)(X_t - X_{t-1})\right)^2\right] \\ &= E\left[\sum_{t,s=1}^{\infty} (X_t - X_{t-1})(X_s - X_{s-1})1(T \geq \max t, s)\right] \\ &= E\left[\sum_{t=1}^{\infty} (X_t - X_{t-1})^2 1(T \geq t)\right] \geq \frac{1}{2} \sum_{t=1}^{\infty} P[T \geq t] = E[T]/2, \end{aligned}$$

where the first equality in the last line follows from the fact that if  $s < t$  say then:

$$\begin{aligned} &E[(X_t - X_{t-1})(X_s - X_{s-1})1(T \geq \max t, s)] \\ &= E[(X_t - X_{t-1})(X_s - X_{s-1})(1 - 1(T < t))] \\ &= E[E[(X_t - X_{t-1})(X_s - X_{s-1})(1 - 1(T < t)) | X_1, \dots, X_{t-1}]] \\ &= E[(X_s - X_{s-1})(1 - 1(T < t))E[X_t - X_{t-1} | X_1, \dots, X_{t-1}]] = 0. \end{aligned}$$

We thus obtain that  $E[T] \leq 2L^2$ . This implies in turn that  $P[T \geq 4L^2] \leq 1/2$  and that  $P[T \geq 4aL^2] \leq 2^{-a}$  for  $a \geq 1$  since  $X_t$  is a Markov chain. The proposition follows. ■

## 5.7 Proof of Proposition 5.12

In order to prove Proposition 5.12 it suffices to prove that *whp* there are no two cycles with a simple path (maybe of length 0) connecting the two. To this end, we consider all possible constellations of such prohibited subgraphs and prove the proposition using a union bound over all of them.

Every simple  $2k$ -cycle in the factor graph consists of  $k$  variables, w.l.o.g. say  $x_1, \dots, x_k$  (all different), and  $k$  clauses  $C_1, \dots, C_k$ , s.t.  $x_i, x_{i+1} \in C_i$ . The cycle itself consists of  $2k$  edges.

As for paths, we have 3 different types of paths: paths connecting a clause in one cycle with a variable in the other (type 1), paths connecting two clauses (type 2), and paths connecting two variables (type 3). Clause-variable paths are always of odd length, and clause-clause, variable-variable paths are always of even length. A  $k$ -path  $P$  consists of  $k$  edges. If it is a clause-variable path, it consists of  $(k-1)/2$  clauses and the same number of variables. If it is a variable-variable path, it consists of  $k/2 - 1$  variables and  $k/2$  clauses and symmetrically for the clause-clause path (we don't take into account the clauses/variables that participate in the cycle, only the ones belonging exclusively to the path).

Our prohibited graphs consist of two cycles  $C_1, C_2$  and a simple path  $P$  connecting them. We call a graph containing exactly two simple cycles and a simple path connecting them a *bi-cycle*. The path  $P$  can be either one of three types. Similarly to the bi-cycle case, one can have a cycle  $C$  and a cord  $P$  in it. We call such a cycle a *cord-cycle*. For parameters  $i, j, k \in [1, n]$ , and  $t \in \{1, 2, 3\}$ , we denote by  $B_{2i,2j,k,t}$  a bi-cycle consisting of a  $2i$ -cycle connected by a  $k$ -path of type  $t$  to a  $2j$ -cycle. Similarly, we denote by  $B_{2i,k,t}$  a cord-cycle consisting of a  $2i$ -cycle with a  $k$ -path of type  $t$  as a cord.

Our goal is then to prove that *whp* the graph induced by the non-core variables contains no bi-cycles and no cord-cycles.

For a fixed factor graph  $H$  we let  $F_H \subseteq F$  be a fixed minimal set of clauses inducing  $H$ , and  $V(H)$  be the set of variables in  $H$ . In order for a fixed graph  $H$  to belong to the factor graph induced by the non-core variables it must be that there exists some  $F_H$  s.t.  $F_H \subseteq F$  and that  $V(H) \subseteq \bar{\mathcal{H}}$  (put differently,  $V(H) \cap \mathcal{H} = \emptyset$ ).

Let  $B = B_{2i,2j,k,t}$  (or  $B = B_{2i,k,t}$  if  $B$  is a cord-cycle) be a fixed bi-cycle and  $F_B$  a fixed minimal-set of clauses inducing  $B$ . We start by bounding  $\Pr[F_B \subseteq F \text{ and } V(B) \cap \mathcal{H} = \emptyset]$  and then use the union bound over all possible bi-cycles (cord-cycles) and inducing minimal sets of clauses. As the two events –  $\{F_B \subseteq F\}$  and  $\{V(B) \cap \mathcal{H} = \emptyset\}$  – are not independent, the calculations are more involved. Loosely speaking, to circumvent the dependency issue, one needs to defuse the effect that the event  $\{F_B \subseteq F\}$  might have on  $\mathcal{H}$ . To this end we introduce a set  $\mathcal{H}^*$ , defined very similarly to  $\mathcal{H}$  only "cushioned" in some sense to overcome the dependency issues (the "cushioning" depends on  $F_B$ ). This is done using similar techniques to [6, 28].

We start by defining the new set of core variables  $\mathcal{H}^*$  (again w.r.t. an ordering  $\pi$  of the clause-variable messages and an initial values vector  $\alpha$ ). The changes compared to  $\mathcal{H}$  are highlighted in bold.

Propositions 5.6 and 5.8 could be easily adjusted to accommodate the 6-gap in the new definition in  $B_2$  and  $B_3$ . Therefore Proposition 5.10 can be safely restated in the context of  $\mathcal{H}^*$ :

**Proposition 5.35.** *If both  $\alpha$  and  $\pi$  are chosen uniformly at random then whp  $\#\mathcal{H}^* \geq (1 - e^{-\Theta(d)})n$ .*

**Proposition 5.36.** *Let  $b = \#V(B)$ , then the set  $J$  defined above satisfies  $\#J \geq b/4$*

**Proof.** Observe that if  $F_B$  is minimal then  $\#F_B \leq b + 1$ . This is because in every cycle the number of variables equals the number of clauses, and in the worst case, the path contains at most one more clause than the number of variables, and the same goes for the cord-cycle. Now suppose in contradiction that  $\#J < b/4$ , then there are more than  $3b/4$  variables in  $V(B)$ , each

Let  $B_1$  be the set of variables whose support w.r.t.  $\varphi$  is at most  $d/3$ .

Let  $B_2$  be the set of non-stable variables w.r.t.  $\pi$  where we redefine the gap in Definition 5.5 to be  $(d/30-6)$  in (a) and (b).

Let  $B_3$  be the set of stable variables w.r.t.  $\pi$  which are violated by  $\alpha$  where we redefine the gap in Definition 5.7 to be  $(d/30-6)$  in (a) and (b).

Let  $J \subseteq V(B)$  be the set of variables appearing in no more than 6 different clauses in  $F_B$

1. Set  $H'_0 = V \setminus (B_1 \cup B_2 \cup B_3 \cup (V(F_C) \setminus J))$ .
2. While there exists a variable  $a_i \in H'_i$  which supports less than  $d/4$  clauses in  $F[H'_i]$  OR appears in more than  $(d/30-6)$  clauses not in  $F[H'_i]$ , define  $H'_{i+1} = H'_i \setminus \{a_i\}$ .
3. Let  $a_m$  be the last variable removed at step 2. Define  $\mathcal{H}^* = H'_{m+1} = H'_{m+1}$ .

appearing in at least 6 different clauses in  $F_B$ . Thus,  $\#F_B > (6 \cdot 3b/4)/3 = 1.5b \underset{b \geq 3}{>} b + 1$  (we divided by three as every clause might have been counted 3 times), contradicting  $\#F_B \leq b + 1$ .  
 ■

The following proposition "defuses" the dependency between the event that a bi-cycle (cord-cycle) was included in the graph and the fact that it doesn't intersect the core variables. In the following proposition we fix an arbitrary  $\pi$  and  $\alpha$  in the definition of  $\mathcal{H}^*$ , therefore the probability is taken only over the randomness in the choice of  $F$ .

**Proposition 5.37.**  $Pr[F_B \subseteq F \text{ and } V(B) \cap \mathcal{H} = \emptyset] \leq Pr[F_B \subseteq F] \cdot Pr[J \cap \mathcal{H}^* = \emptyset]$

To prove Proposition 5.37 we need the following Lemma.

**Lemma 5.38.** *For every bi-cycle (cord-cycle)  $B$  and every minimal inducing set  $F_B$ ,  $\mathcal{H}^*(F, \varphi, \alpha, \pi) \subseteq \mathcal{H}(F \cup F_B, \varphi, \alpha, \pi)$ .*

This lemma clarifies the motivation for defining  $\mathcal{H}^*$ . It is not necessarily true that  $\mathcal{H}(F) \subseteq \mathcal{H}(F \cup F_B)$ . For example, a variable which appears in  $\mathcal{H}(F)$  could disappear from  $\mathcal{H}(F \cup F_B)$  since the clauses in  $F_B$  make it unstable. Loosely speaking,  $\mathcal{H}^*$  is cushioned enough to prevent such a thing from happening.

**Proof.** (Proposition 5.37)

$$Pr[F_B \subseteq F \text{ and } V(B) \cap \mathcal{H} = \emptyset] \leq Pr[F_B \subseteq F \text{ and } J \cap \mathcal{H} = \emptyset] = Pr[J \cap \mathcal{H} = \emptyset | F_B \subseteq F] Pr[F_B \subseteq F].$$

Therefore, it suffices to prove

$$Pr[J \cap \mathcal{H} = \emptyset | F_B \subseteq F] \leq Pr[J \cap \mathcal{H}^* = \emptyset].$$

$$Pr[J \cap \mathcal{H}^* = \emptyset] = \sum_{F: J \cap \mathcal{H}^*(F) = \emptyset} Pr[F = F] \underset{\text{Lemma 5.38}}{\geq} \sum_{F: J \cap \mathcal{H}(F \cup F_B) = \emptyset} Pr[F = F]$$

Break each set of clauses  $F$  into  $F' = F \setminus F_B$  and  $F'' = F \cap F_B$ , and the latter equals

$$\sum_{F': F' \cap F_B = \emptyset, J \cap \mathcal{H}(F' \cup F_B) = \emptyset} \sum_{F'': F'' \subseteq F_B} Pr[F \setminus F_B = F' \text{ and } F \cap F_B = F'']$$

Since the two sets of clauses,  $F \setminus F_B$ , and  $F \cap F_B$ , are disjoint, and clauses are chosen independently, the last expression equals,

$$\begin{aligned} & \sum_{F': F' \cap F_B = \emptyset, J \cap \mathcal{H}(F' \cup F_B) = \emptyset} \sum_{F'': F'' \subseteq F_B} Pr[F \setminus F_B = F'] Pr[F \cap F_B = F''] = \\ & \sum_{F': F' \cap F_B = \emptyset, J \cap \mathcal{H}(F' \cup F_B) = \emptyset} Pr[F \setminus F_B = F'] \underbrace{\sum_{F'': F'' \subseteq F_B} Pr[F \cap F_B = F'']}_1 = \\ & \sum_{F': F' \cap F_B = \emptyset, J \cap \mathcal{H}(F' \cup F_B) = \emptyset} Pr[F \setminus F_B = F'] \end{aligned}$$

Since  $(F \setminus F_B) \cap F_B = \emptyset$ , and clauses are chosen independently, the event  $\{F_B \subseteq F\}$  is independent of the event  $\{F \setminus F_B = F'\}$ . Therefore, the latter expression can be rewritten as

$$\sum_{F': F' \cap F_B = \emptyset, J \cap \mathcal{H}(F' \cup F_B) = \emptyset} Pr[F \setminus F_B = F' | F_B \subseteq F] = Pr[J \cap \mathcal{H} = \emptyset | F_B \subseteq F].$$

■

**Proof.** (Lemma 5.38) The lemma is proved using induction on  $i$  ( $i$  being the iteration counter in the construction of  $\mathcal{H}$ ). For the base case  $H'_0(F) \subseteq H_0(F \cup F_B)$ , since every variable in  $H'_0(F)$  appears in at most 6 clauses in  $F_B$  it holds that  $A_i(F \cup F_B) \subseteq B_i(F)$ ,  $i = 2, 3$ .  $A_1(F \cup F_B) \subseteq B_1(F)$  holds at any rate as more clauses can only increase the support, and the set  $J$  was not even considered for  $H_0$ . Suppose now that  $H'_i(F) \subseteq H_i(F \cup F_C)$ , and prove the lemma holds for iteration  $i + 1$ . If  $x \in H'_{i+1}(F)$  then  $x$  supports at least  $d/3$  clauses in which all variables are in  $H'_i(F)$ . Since  $H'_i(F) \subseteq H_i(F \cup F_B)$ , then  $x$  supports at least this number of clauses with only variables of  $H_i(F \cup F_C)$ . Also,  $x$  appears in at most  $d/30 - 6$  clauses with some variable outside of  $H'_i(F)$ , again since  $H'_i(F) \subseteq H_i(F \cup F_B)$  and  $F_B$  contains at most 6 clauses containing  $x$ ,  $x$  will appear in no more than  $d/30$  clauses each containing some variable not in  $H_i(F \cup F_B)$ . We conclude then that  $x \in H_i(F \cup F_B)$ . ■

**Corollary 5.39.** *Let  $B = B_{2i,k,t}$  be a cord-cycle, and let  $\lambda = 1 - \#\mathcal{H}^*/n$ , then  $Pr[F_B \subseteq F \text{ and } V(B) \cap \mathcal{H} = \emptyset] \leq p(i, k)$  where:*

1.  $p(i, k) \leq (d/n^2)^{(i+k/2)} \cdot \lambda^{(i+\frac{k}{2}-1)/4}$  if  $B$  consists of a  $2i$ -cycle and a variable-variable  $k$ -path as a cord.
2.  $p(i, k) \leq (d/n^2)^{(i+k/2-1)} \cdot \lambda^{(i+\frac{k}{2})/4}$  if  $B$  consists of  $2i$ -cycle and a clause-clause  $k$ -path as a cord.
3.  $p(i, k) \leq (d/n^2)^{(i+\frac{k-1}{2})} \cdot \lambda^{(i+\frac{k-1}{2})/4}$  if  $B$  consists of  $2i$ -cycle and a variable-clause  $k$ -path as a cord.

**Proof.** In (a), we have  $i + \frac{k}{2} - 1$  variables and  $i + \frac{k}{2}$  clauses. Since the clauses are chosen independently,

$$\Pr[F_B \subseteq F] \leq (d/n^2)^{i+\frac{k}{2}}.$$

To bound the event  $\{J \cap \mathcal{H}^* = \emptyset\}$ , observe that  $F_B$  is fixed in the context of this event, and there is no pre-knowledge whether  $F_B$  is included in  $F$  or not. Therefore,  $J$  can be treated as a fixed set of variables, thus the choice of  $\mathcal{H}^*$  is uniformly distributed over  $J$ . Recalling that  $\#J \geq (i + \frac{k}{2} - 1)/4$ , it follows that

$$\Pr[J \cap \mathcal{H}^* = \emptyset] \leq \frac{\binom{n-\#\mathcal{H}^*}{\#J}}{\binom{n}{\#J}} = \frac{\binom{\lambda n}{(i+\frac{k}{2}-1)/4}}{\binom{n}{(i+\frac{k}{2}-1)/4}} \leq \lambda^{(i+\frac{k}{2}-1)/4}.$$

The last inequality follows from standard bounds on the binomial coefficients. (a) now follows immediately from Proposition 5.37. In the same manner items  $b, c$  are proven (just counting how many variables and clauses  $B$  contains, depending on the type of its path).

**Corollary 5.40.** *Let  $B = B_{2i,k,t}$  be a bi-cycle, and let  $\lambda = 1 - \#\mathcal{H}^*/n$ , then  $\Pr[F_B \subseteq F \text{ and } V(B) \cap \mathcal{H} = \emptyset] \leq p(i, j, k)$  where:*

1.  $p(i, j, k) \leq (d/n^2)^{(i+j+k/2)} \cdot \lambda^{(i+j+\frac{k}{2}-1)/4}$  if  $B$  consists of a  $2i, 2j$ -cycles and a variable-variable  $k$ -path.
2.  $p(i, j, k) \leq (d/n^2)^{(i+j+k/2-1)} \cdot \lambda^{(i+j+\frac{k}{2})/4}$  if  $B$  consists of  $2i, 2j$ -cycles and a clause-clause  $k$ -path.
3.  $p(i, j, k) \leq (d/n^2)^{(i+j+\frac{k-1}{2})} \cdot \lambda^{(i+j+\frac{k-1}{2})/4}$  if  $B$  consists of  $2i, 2j$ -cycles and a variable-clause  $k$ -path.

Corollary 5.40 is proven in a similar way to Corollary 5.39. ■

To complete the proof of Proposition 5.12, we use the union bound over all possible bi/cordecycles. We prove for the bi-cycle case, the proof of the cord-cycle is analogous. First consider the case where  $B$  is a bi-cycles with a variable-variable path (in which case the path must be of even length). Let  $s = s_{i,j,k} = i + j + \frac{k}{2} - 1$  (namely,  $\#V(B) = s$  and  $\#F_B = s + 1$ ). The probability of  $B$  is then at most

$$\begin{aligned} & \sum_{i,j,\frac{k}{2}=1}^n \binom{n}{s_{i,j,k}} \cdot (s_{i,j,k})! \cdot (7n)^{s_{i,j,k}+1} \cdot \left(\frac{d}{n^2}\right)^{s_{i,j,k}+1} \cdot \lambda^{s_{i,j,k}/4} \leq \\ & \sum_{i,j,\frac{k}{2}=1}^n 7d \cdot \left(\frac{7en}{s}\right)^s \cdot s^s \cdot n^{s+1} \cdot \left(\frac{d}{n^2}\right)^{s+1} \cdot \lambda^{s/4} \leq \sum_{i,j,\frac{k}{2}=1}^n (7e \cdot d \cdot \lambda^{1/4})^s \cdot \frac{7d}{n} \leq \sum_{i,j,\frac{k}{2}=1}^n \left(\frac{1}{2}\right)^s \cdot \frac{7d}{n} \leq \\ & \sum_{i+j+\frac{k}{2} \leq 4 \log n} \frac{7d}{n} + \sum_{i+j+\frac{k}{2} \geq 4 \log n} \left(\frac{1}{2}\right)^s \leq (4 \log n)^3 \cdot \frac{7d}{n} + n^3 \cdot \frac{1}{n^4} = o(1). \end{aligned}$$

We now move to the case  $B$  is a bi-cycles with a clause-clause path (in which case the path again must be of even length). Let  $s = s_{i,j,k} = i + j + \frac{k}{2}$  (namely,  $\#V(B) = s$  and  $\#F_B = s - 1$ ).

Observe that in this case the number of classes in  $F_B$  is  $s - 1$ , however only for  $s - 3$  clauses one has the freedom in choosing the third variable (the two clauses which are the endpoints of the path are completely determined once the order of the variables is fixed). The probability is then at most

$$\begin{aligned} & \sum_{i,j,\frac{k}{2}=1}^n \binom{n}{s_{i,j,k}} \cdot (s_{i,j,k})! \cdot (7n)^{s_{i,j,k}-3} \cdot \left(\frac{d}{n^2}\right)^{s_{i,j,k}-1} \cdot \lambda^{s_{i,j,k}/4} \leq \\ & \sum_{i,j,\frac{k}{2}=1}^n \left(\frac{7en}{s}\right)^s \cdot s^s \cdot n^{s-3} \cdot \left(\frac{d}{n^2}\right)^{s-1} \cdot \lambda^{s/4} \leq \sum_{i,j,\frac{k}{2}=1}^n (7e \cdot d \cdot \lambda^{1/4})^s \cdot \frac{1}{n} = o(1) \end{aligned}$$

Lastly, we need to consider the case  $B$  is a bi-cycles with a clause-variable path (in which case the path must be of odd length). Let  $s = s_{i,j,k} = i + j + \frac{k-1}{2}$  (namely,  $s = \#V(B) = \#F_B$ ). Again, one clause in  $F_B$  is completely determined once the order of the variables is fixed. The probability is then at most

$$\begin{aligned} & \sum_{i,j,\frac{k-1}{2}=1}^n \binom{n}{s_{i,j,k}} \cdot (s_{i,j,k})! \cdot (7n)^{s_{i,j,k}-1} \cdot \left(\frac{d}{n^2}\right)^{s_{i,j,k}} \cdot \lambda^{s_{i,j,k}/4} \leq \\ & \sum_{i,j,\frac{k-1}{2}=1}^n \left(\frac{7en}{s}\right)^s \cdot s^s \cdot n^{s-1} \cdot \left(\frac{d}{n^2}\right)^s \cdot \lambda^{s/4} \leq \sum_{i,j,\frac{k-1}{2}=1}^n (7e \cdot d \cdot \lambda^{1/4})^s \cdot \frac{1}{n} = o(1) \end{aligned}$$

To sum up, the probability of a bi-cycle in the graph induced by the non-core variables is  $3 \cdot o(1) = o(1)$ .

### 5.7.1 Outline of Proof of Proposition 5.13

The proof is basically the same as that of Proposition 5.12. One defines the same notion of ‘‘cushioned’’ core  $\mathcal{H}^*$ , and proceeds similarly. We therefore reprove only the last part – the union bound over all possible cycles.

First let us bound the number of cycles of length  $k$ . There are  $\binom{n}{k}$  ways to choose the variables inducing the cycle, and  $k!/2$  ways to order them on the cycle. As for the set of clauses that induces the cycle, once the cycle is fixed, we have at most  $(7n)^k$  ways of choosing the third variable and setting the polarity in every clause. In what follows we let  $\lambda n$  be the number of vertices in the non-core factor graph.

Using the union bound, the probability of a cycle of length at least  $k$  in the non-core factor graph is at most

$$\begin{aligned} \sum_{t=k}^{\lambda n} \underbrace{\binom{n}{t} \cdot t! \cdot (7n)^t}_{\text{choose the cycle}} \cdot \underbrace{\left(\frac{d}{n^2}\right)^t \cdot \lambda^{t/2}}_{\text{cycle included but doesn't intersect } \mathcal{H}^*, \text{Prop. 5.37}} & \leq \sum_{t=k}^{\lambda n} \left(\frac{7en}{t}\right)^t \cdot t^t \cdot n^t \cdot \left(\frac{d}{n^2}\right)^t \cdot \lambda^{t/2} \\ & = \sum_{t=k}^{\lambda n} (7e \cdot d \cdot \sqrt{\lambda})^t \end{aligned}$$

Assuming that Proposition 5.35 holds (which is the case *whp*), then  $\lambda = e^{-\Theta(d)}$  and  $7e \cdot d \cdot \sqrt{e^{-\Theta(d)}} = e^{-\Theta(d)}$  – which is much smaller than 1. In particular, the last summation is simply

the sum of a decreasing geometric series with quotient  $e^{-\Theta(d)}$ , which sums up to at most twice the first item, which is at most  $e^{-\Theta(dk)}$ .

## 5.8 Discussion

Our results show that WP is effective in solving  $\mathcal{P}_{n,p}$ . Though not being the first to give an algorithm for  $\mathcal{P}_{n,p}$ , our results are a first example of rigorously analyzing a message passing algorithm on a natural non-trivial random SAT distribution. We remark that our goal was to analyze WP under its most common definition, resisting attempts to modify the algorithm in ways that would simplify the analysis. One such simplification would result if messages are updated in *parallel* in two phases: clause-variable messages updates and then variable-clause messages updates.

It is interesting to compare the results obtained here to the non-planted case. First note that in the non-planted case, the computational hardness of finding a satisfying assignment is clearly increasing with the formula's density. This should be contrasted with the planted case where some algorithms are proved more efficient as the density increases: for example, here we prove that WP converges after two iteration for dense formulas, while  $\log n$  iterations are needed for sparse formulas (also compare [28] to [10]).

In the non-planted case, for low density formulas (considerably below the satisfiability threshold), some algorithms were rigorously shown to find *whp* a satisfying assignment efficiently [5, 15]. Experimental results predict that as the density of the formula increases, more sophisticated algorithms are needed in order to find a satisfying assignment. At higher densities (closer to the satisfiability threshold), there is a major gap between the experimental performance of the best known algorithms [14] (shown to work for density  $\sim 4.2$ ), and the best rigorously-analyzed algorithm [35] (density 3.52).

One possible explanation for the increasing computational hardness of finding solutions in the non-planted case is based on the geometry of the space of satisfying assignment. It is now established [1, 44] that when  $k \geq 8$  and just below the satisfiability threshold, the space of solutions decomposes into an exponential number of (Hamming-distance) connected clusters such that the distance between each to is linear in the number of variables. Such complex "fractal" geometry of the space of solutions poses a complex algorithmic challenge.

Our results, and similarly [28, 27], show that in the planted case (with density some large constant above the satisfiability threshold), the algorithmic task of finding a satisfying assignment is much easier than in the near-threshold regime, and in particular, the naïve WP algorithm is effective in finding satisfying assignments. This is consistent with the fact that planted formulas in this regime have only one cluster of satisfying assignments (Chapter 2).

We conclude with an open problem. Can our analysis be extended to show that Belief Propagation (BP) finds a satisfying assignment to  $\mathcal{P}_{n,p}$  in the setting of Theorem 5.1? Experimental results predict the answer to be positive. However, our analysis of WP does not extend as is to BP. In WP, all warnings received by a variable (or by a clause) have equal weight, but in BP this need not be the case (there is a probability level associated with each warning). In particular, this may lead to the case that messages received from non-core portions of the formula can

---

effect the core, a possibility that our analysis managed to exclude for the WP algorithm.



# Bibliography

- [1] D. Achlioptas and A. Coja-Oghlan. Algorithmic barriers from phase transitions. *preprint*, 2008.
- [2] D. Achlioptas and C. Moore. Random  $k$ -SAT: Two moments suffice to cross a sharp threshold. *CoRR*, cond-mat/0310227, 2003.
- [3] D. Achlioptas and Y. Peres. The threshold for random  $k$ -SAT is  $2^k \log 2 - O(k)$ . *Journal of the AMS*, 17(4):947–973, 2004.
- [4] D. Achlioptas and F. Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In *Proc. 38th ACM Symp. on Theory of Computing*, pages 130–139, 2006.
- [5] M. Alekhnovich and E. Ben-Sasson. Linear upper bounds for random walk on small density random 3-cnfs. *SIAM J. on Comput.*, 36(5):1248–1263, 2007.
- [6] N. Alon and N. Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM J. on Comput.*, 26(6):1733–1748, 1997.
- [7] N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. *Random Structures and Algorithms*, 13(3-4):457–466, 1998.
- [8] N. Alon and J. Spencer. *The probabilistic method*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, second edition, 2000. With an appendix on the life and work of Paul Erdős.
- [9] J. Ardelius and E. Aurell. Behavior of heuristics on large and hard satisfiability problems. *Phys. Rev.*, E 74, 037702, 2006.
- [10] E. Ben-Sasson, Y. Bilu, and D. Gutfreund. Finding a randomly planted assignment in a random 3CNF. *manuscript*, 2002.
- [11] A. Blum and J. Spencer. Coloring random and semi-random  $k$ -colorable graphs. *J. of Algorithms*, 19(2):204–234, 1995.
- [12] J. Böttcher. Coloring sparse random  $k$ -colorable graphs in polynomial expected time. In *Proc. 30th International Symp. on Mathematical Foundations of Computer Science*, pages 156–167, 2005.

- 
- [13] J. Böttcher and D. Vilenchik. On the tractability of coloring semirandom graphs. doi:10.1016/j.ipl.2008.04.011, 2008.
- [14] A. Braunstein, M. Mezard, and R. Zecchina. Survey propagation: an algorithm for satisfiability. *Random Structures and Algorithms*, 27:201–226, 2005.
- [15] A. Z. Broder, A. M. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-CNF formulas. In *Proc. 4th ACM-SIAM Symp. on Discrete Algorithms*, pages 322–330, 1993.
- [16] H. Chen. An algorithm for sat above the threshold. In *6th International Conference on Theory and Applications of Satisfiability Testing*, pages 14–24, 2003.
- [17] A. Coja-Oghlan. Coloring semirandom graphs optimally. In *Proc. 31st International Colloquium on Automata, Languages, and Programming*, pages 383–395, 2004.
- [18] A. Coja-Oghlan, M. Krivelevich, and D. Vilenchik. Why almost all  $k$ -colorable graphs are easy. In *Proc. 24th Symp. on Theoretical Aspects of Comp. Science*, volume 4393 of *Lecture Notes in Comput. Sci.*, pages 121–132, 2007.
- [19] A. Coja-Oghlan, M. Krivelevich, and D. Vilenchik. Why almost all satisfiable  $k$ -CNF formulas are easy. In *13th conference on Analysis of Algorithms, DMTCS proceedings*, pages 89–102, 2007.
- [20] S. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd ACM Symp. on Theory of Computing*, pages 151–158, 1971.
- [21] P. Erdős, S. Suen, and P. Winkler. On the size of a random maximal graph. *Random Structures and Algorithms*, 6(2-3):309–318, 1995.
- [22] U. Feige. Relations between average case complexity and approximation complexity. In *Proc. 34th ACM Symp. on Theory of Computing*, pages 534–543, 2002.
- [23] U. Feige, A. Flaxman, and D. Vilenchik. On the diameter of the set of satisfying assignments in random satisfiable  $k$ -CNF formulas. *manuscript*, 2008.
- [24] U. Feige and J. Kilian. Heuristics for semirandom graph problems. *J. Comput. and Syst. Sci.*, 63(4):639–671, 2001.
- [25] U. Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures and Algorithms*, 16(2):195–208, 2000.
- [26] U. Feige, E. Mossel, and D. Vilenchik. Complete convergence of message passing algorithms for some satisfiability problems. In *RANDOM*, pages 339–350, 2006.
- [27] U. Feige and D. Vilenchik. A local search algorithm for 3SAT. Technical report, The Weizmann Institute of Science, 2004.
- [28] A. Flaxman. A spectral technique for random satisfiable 3CNF formulas. In *Proc. 14th ACM-SIAM Symp. on Discrete Algorithms*, pages 357–363, 2003.

- 
- [29] E. Friedgut. Sharp thresholds of graph properties, and the  $k$ -sat problem. *J. Amer. Math. Soc.*, 12(4):1017–1054, 1999.
- [30] A. Frieze and C. McDiarmid. Algorithmic theory of random graphs. *Random Structures and Algorithms*, 10(1-2):5–42, 1997.
- [31] R. Gallager. *Low-Density Parity-Check Codes*. SMIT Press, Cambridge, 1963.
- [32] J. Håstad. Some optimal inapproximability results. *J. of the ACM*, 48(4):798–859, 2001.
- [33] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.
- [34] C. Hui and A. Frieze. Coloring bipartite hypergraphs. In *Proceedings of the 5th International Conference on Integer Programming and Combinatorial Optimization*, pages 345–358, 1996.
- [35] A. Kaporis, L. Kirousis, and E. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. In *Proc. 10th Annual European Symposium on Algorithms*, volume 2461 of *Lecture Notes in Comput. Sci.*, pages 574–585. Springer, Berlin, 2002.
- [36] E. Koutsoupias and C. H. Papadimitriou. On the greedy algorithm for satisfiability. *Info. Process. Letters*, 43(1):53–55, 1992.
- [37] M. Krivelevich, B. Sudakov, and D. Vilenchik. On the random satisfiable 3CNF process. *manuscript*, 2007.
- [38] M. Krivelevich and D. Vilenchik. Semirandom models as benchmarks for coloring algorithms. In *Third Workshop on Analytic Algorithmics and Combinatorics*, pages 211–221, 2006.
- [39] M. Krivelevich and D. Vilenchik. Solving random satisfiable 3CNF formulas in expected polynomial time. In *Proc. 17th ACM-SIAM Symp. on Discrete Algorithms*, pages 454–463, 2006.
- [40] L. Levin. Average case complete problems. *SIAM J. on Comput.*, 15(1):285–286, 1986.
- [41] L. Lovász. *Combinatorial problems and exercises*. Elsevier Science Publishers, Amsterdam, second edition, 1993.
- [42] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman. Analysis of low density parity check codes and improved designs using irregular graphs. In *Proc. 30th ACM Symp. on Theory of Computing*, pages 249–258, 1998.
- [43] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman. Efficient erasure correcting codes. *IEEE Trans. on Info. Theory*, 47:569–584, February 2001.
- [44] M. Mezard, T. Mora, and R. Zecchina. Clustering of solutions in the random satisfiability problem. *Physical Review Letters*, 94:197–205, 2005.

- 
- [45] R. Mulet, A. Pagnani, M. Weigt, and R. Zecchina. Coloring random graphs. *Phys. Rev. Lett.*, 89(26):268701, 2002.
- [46] C. Papadimitriou. On selecting a satisfying truth assignment. In *Proc. 32nd IEEE Symp. on Found. of Comp. Science*, 1991.
- [47] A. Parke. Scaling properties of pure random walk on random 3SAT. In *n Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, 2002.
- [48] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
- [49] T. Richardson, A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity check codes. *IEEE Trans. on Info. Theory*, 47:619–637, February 2001.
- [50] S. Seitz, M. Alava, and P. Orponen. Focused local search for random 3-satisfiability. *Journal of Statistical Mechanics*, P06006:1–27, 2005.
- [51] B. Selman, H. Kautz, and B. Cohen. Local search strategies for satisfiability testing. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 26:521–531, 1996.
- [52] D. Vilenchik. It’s all about the support: a new perspective on the satisfiability problem. *Journal on Satisfiability, Boolean Modeling, and Computation*, 3:125–139, 2007.