RICE UNIVERSITY

Probabilistic Phenomena in Random Combinatorial Problems

by

Demetrios D. Demopoulos

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

MASTER OF SCIENCE

Approved, Thesis Committee:

Moshe Y. Vardi , Chair
Professor of Computer Science
Rice University

Devika Subramanian
Professor of Computer Science
Rice University

Nathaniel Dean
Professor of Mathematical Sciences
Texas Southern University

Houston, Texas

April, 2003

ABSTRACT

Probabilistic Phenomena in Random Combinatorial Problems

by

Demetrios D. Demopoulos

This is an experimental investigation of three combinatorial problems. I examined the average-case complexity of random 3-SAT and of 3-Colorability of random graphs, and the satisfiability of random 1-3-HornSAT. All these problems, not only are interesting for their own sake, but also are of great practical importance since many other problems in computer science, engineering and other fields can be reduced to these. We systematically explored a large part of the problems' space, varying the size and the constrainedness of the instances, as well as the tools we used to solve them. We observed new phase transitions from polynomial to exponential complexity for random 3-SAT. A similar picture emerged for 3-Colorability. These experimental observations are important for understanding the inherent computational complexity of the problems. In the case of random 1-3-HornSAT, our findings suggest that there is a threshold at which the satisfiability changes from 1 to 0.

Acknowledgments

I would like to thank my advisor, Moshe Y. Vardi, for his constant assistance and support, and his wise advising. But even more, I would like to thank him for offering me a great teacher-student relationship. The lessons I learned from him about professionalism, collaboration, and ethics are at least as valuable as the training I got as a researcher while working on my thesis.

I also want to thank the other two members of my committee for their support; Devika Subramanian for offering me her constant encouragement and all the resources that she holds in her brains and Nate Dean for always being more than happy to discuss with me about research and throw at me questions that reminded me that there are always many ways to look at a subject.

I have really enjoyed working with Alfonso San Miguel Aguirre, Cristian Coarfa, and Gerrick Green, towards the completion of parts of the research presented in this thesis.

I feel the need to say a big thank you to all the staff of the Computer Science Department at Rice for doing their best for all of us to have a smooth, productive day everyday at school; especially to Iva Jean Jorgensen and Darnell Price, for being there for me all these years as mentors and friends.

It was a great experience to be in the same research group with Armando Tac-

chella, Aniello Murano, and Guoqiang Pan. Discussing with them about research related issues and having their feedback was very helpful for me; discussing with them about all other issues of life was lots of fun.

Finally, I am very grateful to Maria for being there for me from the beginning of this journey, and for being "my Greece" all this time that I have been thousands of miles away from Greece.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

In the last decade phase transitions in randomly generated combinatorial problems have been studied intensively. Although the idea of phase transition in combinatorial problems has been introduced as early as 1960 [37], in recent years it has been a main subject of research in the communities of theoretical computer science, artificial intelligence and statistical physics. This interest was stimulated by the discovery of a fascinating connection between the *density* of combinatorial problems and their computational complexity, see [16, 68].

Combinatorial phase transitions are also known as *threshold phenomena*. Phase transitions have been observed both on the probability that an instance of a problem has a solution and on the computational cost of solving an instance. In few cases these phase transitions have been also proved[17, 31, 43]. Families of problems that has been the focus of this research are propositional satisfiability, the colorability of graphs, and constraint satisfaction. For the purposes of this thesis we are concerned about problems of the first two families. More specifically, we will present results concerning the following three problems: random 3-SAT, 3-Colorability of random graphs, and a variation of HornSAT called 1-3-HornSAT.

In what follows in this chapter, we will define each of these problems, we will state

the questions we are trying to answer, and we will shortly discuss our experiments and our findings. In Chapter 2, we will analytically present our research on the average-case complexity of random 3-SAT. In Chapter 3 we present a similar analysis on the complexity of the 3-Colorability of random graphs. In the case of 1-3-HornSAT, discussed in Chapter 4, we are interested in the probability that a random instance is satisfiable. Finally, in Chapter 5, we draw our conclusions and present some future work as an extension to this thesis.

1.1 Definitions

Let us review some definitions* related to combinatorial phase transitions. Let X be a finite set and $|X| = n$. Let A be a random subset of X constructed by a random procedure according to the probability space $\Omega(n, m) \stackrel{D}{=} (2^X, 2^{2^X}, \Pr)$, where Pr is defined as :

$$\Pr_{\Omega(n,m)}(A) = \begin{cases} 1/\binom{n}{m^*} & \text{if } |A| = m^* \\ 0 & \text{otherwise} \end{cases},$$

where m can be an integer and

*The definitions found in this paper as well as more definitions and results can be found in [28]

$$m^* = \begin{cases} 0 & \text{if } m < 0 \\ m & \text{if } 0 \leq m \leq n \\ n & \text{if } m > n \end{cases}$$

The random procedure consists of selecting m^* elements of X without replacement. An property Q of X is a subset of 2^X. Q is increasing if $A \in Q$ and $A \subseteq B \subseteq X$ implies $B \in Q$. Q is non-trivial if $\emptyset \notin Q$ and $X \in Q$. A property Q consists of a sequence of sets $\{X_n : n \geq 1\}$ such that $|X_n| < |X_{n+1}|$ and a family $\{Q_n : n \geq 1\}$ where each Q_n is a property of X_n. Q is increasing (non-trivial) if Q_n is increasing (resp. non-trivial) for every $n \geq 1$.

Let Q be an increasing non-trivial property and $\theta : N \to R^+$ be a strictly positive function. We say that θ is a threshold for Q if for every $f : N \to N$:

1. If $\lim_{n\to\infty} f(n)/\theta(n) = 0$ then $\lim_{n\to\infty} \Pr_{\Omega(n,f(n))}(Q_n) = 0$

2. If $\lim_{n\to\infty} f(n)/\theta(n) = \infty$ then $\lim_{n\to\infty} \Pr_{\Omega(n,f(n))}(Q_n) = 1$

θ is a sharp threshold Q if for every $f : N \to N^+$:

1. If $\sup_{n\to\infty} f(n)/\theta(n) < 1$ then $\lim_{n\to\infty} \Pr_{\Omega(n,f(n))}(Q_n) = 0$

2. If $\inf_{n\to\infty} f(n)/\theta(n) > 1$ then $\lim_{n\to\infty} \Pr_{\Omega(n,f(n))}(Q_n) = 1$

We will say that a problem exhibits a phase transition if there is a sharp threshold.

1.2 Random 3-SAT

A problem that has received a lot of attention is the *3-satisfiability problem* (3-SAT), a paradigmatic combinatorial problem that is important also for its own sake. An instance of 3-SAT consists of a conjunction of clauses, each one a disjunction of three literals. The goal is to find a truth assignment that satisfies all clauses. The *density* of a 3-SAT instance is the ratio of the number of clauses to the number of Boolean variables. We call the number of variables the *order* of the instance. Clearly, a low density suggests that the instance is under-constrained, and therefore is likely to be satisfiable, while a high density suggests that the instance is over-constrained and is unlikely to be satisfiable. Experimental research [23, 68] has shown that for ratio below (roughly) 4.26, the probability of satisfiability of a random 3-SAT instance goes to 1 as the order increases, while for ratio above 4.26 the probability goes to 0. At 4.26, the probability of satisfiability is near 0.5. This satisfiability threshold density has been called the *crossover point*. Theoretically establishing the density at the crossover point is difficult, and is the subject of continuing research, cf. [38, 33, 1].

The experiments in [23, 68], which applied algorithms based on the so-called *Davis-Longemann-Loveland method* (abbr., DLL method) (a depth-first search with unit propagation [30]), also show that the density of a 3-SAT instance is intimately related to its computational complexity. Intuitively, under-constrained instances are easy to solve, as a satisfying assignment can be found fast, and over-constrained instances are

also easy to solve, as all branches of the search terminate quickly. Indeed, the data displayed in [23, 68] demonstrate a peak in running time essentially at the crossover point. Using finite-size scaling techniques, [56] demonstrated a *phase transition* at the crossover point, viz., a marked qualitative change in the structural properties of the problem. This pattern of computational behavior with a peak at the crossover point is called the *easy-hard-easy* pattern in [67].

This picture, however, is quite simplistic for various reasons. First, the terms "easy" and "hard" do not carry any rigorous meaning. The computational complexity of a problem is typically measured by its *scalability*, that is, its growth as a function of the input size. Thus, one studies computational complexity on an infinite collection of instances. The easy-hard-easy pattern, however, is observed when the order is fixed while the density varies, but once the order is fixed, there are only finitely many possible instances. For that reason, theoretical analyses of the random 3-SAT problem focus on collections of fixed-density instances, rather than on collections of fixed-order instances, cf. [4]. Second, in the context of a concrete application, e.g., bounded model checking [6], planning [53], or scheduling [24], it is typically the order that tends to grow while the density stays fixed, for example, as we search for longer and longer counterexamples in bounded model checking. Thus, experimental results that vary density while fixing the order tell us little about the computational complexity of 3-SAT in such settings. Finally, it is not clear where the boundaries between the

so-called "easy", "hard", and "easy" regions are. A widely held belief [68, 67] is that random 3-SAT problems are "hard" only for densities very close to the crossover point. Much work has therefore focused on explaining the "jump" in computational complexity around the crossover point using finite-size scaling [67] and backbones [65]. This alleged "jump", however, has not been documented experimentally. In fact, there is almost no experimental work that studies how the running time of a SAT solver varies as a function of the order for fixed-density instances (a few results of this nature, though not a systematic study, are reported in [22, 23, 67]). Further, the experiments reported in [68, 23] are based solely on DLL algorithms. While these are indeed the most popular algorithms for the satisfiability problem, one cannot jump to conclusions about the inherent and practical complexity of random 3-SAT based solely on experiments using these algorithms. We may observe different phenomena by experimenting with SAT solvers that embody different algorithms.

The goal of our experimental algorithmic research reported here is to determine how the average-case complexity of random 3-SAT, understood as a function of the order for fixed density instances, depends on the density. Is there a phase transition in which the complexity shifts from polynomial to exponential? Is such a transition dependent or independent of the solver?

To explore these questions, we set out to obtain a good coverage of an initial quadrangle of the two-dimensional $d \times n$ quadrant, where d is the density and

n is the order. We explored the range $0 \leq d \leq 15$. We attempted to maximize the order of the sampled instances, given our resource constraints. We used three different SAT solvers, embodying different underlying algorithms. GRASP (`vinci.inesc.pt/~jpms/grasp/`) is based on the DLL method, but it augments the search with a conflict-analysis procedure that enables it to backtrack non-chronologically and record the causes of conflict. Experimental results [61] show that GRASP is very efficient for a large number of realistic SAT instances, and it has proven to be a very effective SAT solver in the context of automated hardware design [66]. The CPLEX MIP Solver is a commercial optimizer for linear-programming problems with integer variables (`www.cplex.com`). It employs a branch-and-bound technique using linear-programming relaxations that can be complemented with the dynamic generation of cutting planes. While branch-and-bound is related to depth-first-search, the cutting-planes technique is more powerful than resolution [48]. CUDD (`bessie.colorado.edu/~fabio/CUDD`) implements functions to manipulate Reduced Ordered Binary Decision Diagrams (ROBDDs), which provide an efficient representation for Boolean functions [13]. Unlike GRASP and CPLEX, CUDD does not search for a single satisfying truth assignment. Rather, it constructs a compact symbolic representation of the set of satisfying truth assignments and then checks whether this set is nonempty. Uribe and Stickel [76] compared ROBDDs with the DLL method for SAT solving, concluding that the methods are incomparable, and that ROBDDs

dominate the DLL method on many examples. Recent work by Groote and Zantema proved the incomparability of ROBDDs and resolution [45]. ROBDDs have proven in the 1990s to be very effective in the context of hardware verification [14, 50].

Our aim was not to directly compare the performance of the different solvers in order to see which one has the "best" performance, but rather to understand their behavior in the $d \times n$ quadrant in order to make qualitative observations on how the complexity of random 3-SAT is viewed from different algorithmic perspectives. It is important to note that the algorithms used in GRASP, CPLEX, and CUDD do not explicitly refer to the density of the input instances. Thus, a qualitative change in the behavior of the algorithm, as a result of changing the density, indicates a genuine structural change in the SAT instances from the perspective of the algorithm.

In analyzing our experimental results we focus on measuring the *median* running time as a function of the order for a set of instances of fixed density.* This gives us a measure of the running time of the algorithm for that density. Our findings show that for GRASP and CPLEX the easy-hard-easy pattern is better described as an *easy-hard-less-hard* pattern, where, as is the standard usage in computational complexity theory, "easy" means *polynomial time* and "hard" means *exponential time*.

*It is easy to see that 3-SAT is NP-complete for instances of each fixed density, as the generic reduction of NP to 3-SAT [40] produces instances of fixed density and each density can be obtained by adding linearly many redundant variables or redundant clauses. Thus, we'd expect the worst-cases running time to be exponential for all densities, and, consequently, to find hard instances in the "easy" region, cf. [41]. The issue of median vs. mean running time is discussed later.

When we start with low-density instances and then increase the density, we go from a region of polynomial running time, to a region of exponential running time, where the exponent first increases and then decreases as a function of the density. Thus, we observe at least *two* phase transitions as the density is increased: a transition at around density 3.8 from polynomial to exponential running time and a transition at around density 4.26 (the crossover point) from an increasing exponent to a decreasing exponent. The region between 3.8 and 4.26 is also characterized by the prevalence of very hard instances, the so called "heavy-tail phenomenon", cf. [41, 47, 62, 67]. Our results indicate one or more phase transitions in this region, where the ratio of the mean to median running time peaks. For CPLEX we also observe another phase transition at around density 1.7 from linear running time to quadratic running time. Note that we are using the term "phase transition" in a somewhat liberal sense. The phase transitions that we observed involve various measures: a change in the degree of polynomial running time, a change from polynomial to exponential running time, and a change in the direction of the heavy-tail phenomenon. All these suggest to us marked qualitative changes in structural properties.

A very different picture emerges for CUDD. Here the algorithm is exponential (in both time and space) for densities between 0.5 and 15. There is, however, no peak around the crossover point and no heavy-tail phenomenon was observed. We observed, however, a peak in the size of the final BDDs constructed by the algorithm

at around density 2, indicating a phase transition at around this density. At a very low density (0.1) we did observe polynomial (cubic) behavior, which suggests that another phase transition is "lurking" between densities 0.1 and 0.5.

There are two conclusions that can be drawn from our experiments. First, the "phase transition" in average case computational complexity of random 3-SAT should not be identified with the "phase transition" in satisfiability. The sharp shift of average case complexity from polynomial to exponential occurs well before the crossover point. This implies that explanations for shifts in computational complexity cannot center around phenomena observed at the crossover point [67, 65] Second, unlike earlier predictions (cf. [58, 21]), phase transitions in average-case complexity (unlike the one for satisfiability) are not solver-independent. This implies that any theory attempting to explain the sharp shift in computational complexity must take the characteristics of the solver into account, as in [2].

1.3 3-Colorability of random graphs

A constraint satisfaction problem that has been the center of number of studies* is that of *graph coloring*. An instance of a k-coloring problem consists of a graph, that is a set of vertices and a set of edges, and a set of colors of size k. The goal is to find a coloring of the vertices using the k colors, such that each vertex has exactly one color and there is no pair of adjacent vertices (i.e. vertices connected with an edge)

*For an extensive list of publications on graph coloring see [25]

that has the same color. In this study, we will focus on the problem of 3-colorability (i.e. there are three different colors available) of random graphs.

We call *connectivity* the average degree of a graph, i.e. the average number of edges incident to a vertex of the graph. The number of vertices of the graph is called the order of the graph. As the density of the formula in the 3-SAT, the connectivity of the graph in 3-colorability is a parameter that relates to the probability that the problem has a solution. Graphs of low connectivity, i.e. sparse graphs, are most likely colorable, while graphs of high connectivity are dense and unlikely to be colorable. Experimental results [47, 16] have shown that for connectivity below 4.6 (roughly) the probability that a random graph is 3-colorable goes to 1 as the order increases, while for connectivity above 4.6 the probability goes to 0. The colorability threshold is believed to be around connectivity 4.6*, where the probability that a random graph is 3-colorable is roughly 0.5.

In the previous section, we describe our research on 3-SAT, where we observe a polynomial to exponential complexity phase transition located to the left of the satisfiability threshold. We are interested to see if this phenomenon can also be observed in other combinatorial problems, such as the 3-colorability.

Recall that in 3-SAT research has shown that the density of a formula is closely related to the cost of finding a satisfying assignment. This is also the case for 3-

*The earlier experiments in [16] are on reduced graphs and locate the threshold slightly above 5. Later experiments [47] on random graphs show that the threshold is around 4.6

colorability. The results in [16, 47] show that, when applying complete depth-first backtracking algorithms based on the Brélaz heuristic [11, 75] to random graphs, sparse (low-connectivity) graphs are relatively easy to color, and dense (high-connectivity) graphs are also relatively easy to prove non-colorable. The experiments demonstrate that the computational cost peaks around the threshold (connectivity 4.6).

Hogg and Williams in [47] also observe a second phase transition to the left of the colorability threshold. It is at a region, where although the majority of the instances are relatively easy, there are a few instances that are extremely hard to solve. According to the authors this region corresponds to a transition from polynomial to exponential scaling of the average computational cost. A model that relates the average cost to the number of partial solutions of different sizes [78] is used to approximate the location of this transition. By specializing their model to the constraints of the colorability problem, the authors estimate that the transition is happening at around connectivity 2.2 (while the original model gives an estimation of 2.9).

Culberson and Gent in [26] suggest that the double phase transition conjectured in [47] occurs only in graphs of small order. Their analysis is based on the notion of a "frozen development" (that is an idea analogous to the backbone for SAT). A pair of vertices in a colorability instance is called *frozen* if the two vertices have always the same color in every valid coloring of the graph. The authors study the development of frozen pairs on random graphs, which is shown to be happening very close to

the colorability threshold. When frozen pairs are present it is likely that a search algorithm can make an early mistake by setting two nodes of a frozen pair to different colors and then start thrashing. Since the frozen pairs seem to appear only when we are very close to the threshold, the authors suggest that the "double phase transition" can only be seen in small graphs, and that as the order of the instances increases it converges to a single phase transition.

Motivated by the seemingly opposite results in [47] and [26] and having already studied a similar problem for 3-SAT we set out to investigate how the average-case complexity of the 3-colorability of random graphs depends on the connectivity. Our goal is determine how does the average-case complexity of 3-colorability changes with the order, for a fixed connectivity, if there is a phase transition from polynomial to exponential complexity, and where is such a transition located.

To answer these questions we systematically explored the two-dimensional $\gamma \times n$ quadrant, where γ is the connectivity and n is the order of a random graph. We covered the range $1 \leq \gamma \leq 20$, and we tried to maximize the order of the instances, given the resources we had available. To solve the problems we used Culberson's SMALLK program (`http://www.cs.ualberta.ca/~joe/Coloring/Colorsrc/smallk.tar.gz`). SMALLK is a backtrack based program for coloring graphs. The underlying algorithm has two steps; firstly, a recursive backtrack search reduces the graph by deleting vertices, edges and available colors while each of this reductions is recorded in a

stack. Then, the information in the stack is used to reconstruct and color the graph. SMALLK has been shown [27] to be perform very well when used to solve graphs with small chromatic number, like in the case of 3-colorability. Note, that this solver is also used in [26].

Our goal here is to make qualitative observations about the computational cost of the 3-colorability of random graphs. When analyzing our experimental data from the $\gamma \times n$ quadrant, we focus on the median* running time. Our experimental findings show that the median running time of SMALLK scales polynomially with the order for connectivity up to 4.2, while it scales exponentially for connectivity 4.4 and above. We observe a phase transition at around connectivity 4.3, where the median running time shifts from being polynomial in the order to exponential. This transition is also followed by a heavy-tail phenomenon, similar to the one we observed for random 3-SAT. Our findings agree with the double phase transition conjectured in [47], and refute the suggestion made in [26] that the previously conjectured double phase transition is only an effect of small graph order. The observed phase transition might be solver-dependent, but it is not an artifact of the solver since the algorithm used is oblivious to the connectivity, or any other structural property, of the instance. These findings also suggest that when we look at a polynomial to exponential complexity phase transition, there is a robust behavior among problems such as 3-SAT

*In [47] the authors conjecture a second phase transition from polynomial to exponential average cost.

and 3-Colorability (and possibly more combinatorial problems).

1.4 Random 1-3-HornSAT

A problem similar to random SAT is that of the satisfiability of random Horn formulae. An instance of the random Horn-SAT in conjunctive normal form (CNF) is a conjunction of Horn clauses; each Horn clause is a disjunction of literals* of which at most one can be positive.

Although random Horn-SAT is a problem very close to random 3-SAT, it is a tractable problem unlike 3-SAT. The complexity of the Horn-SAT is linear in the size of the formula [32]. The linear complexity of Horn-SAT allow us to study experimentally the satisfiability of the problem for much bigger input sizes than those used in our studies on 3-SAT and 3-Colorability and also in similar research [47, 23, 68, 19].

An additional motivation for studying random Horn-SAT comes from the fact that Horn formulae are connected to several other areas of Computer Science and Mathematics [60]. Horn formulae are connected to automata theory. The transition relation, the starting state, and the set of final states of an automaton can be described using Horn clauses. For example, if we consider a binary-tree automaton, then Horn clauses of length three can be used to describe its transition relation while Horn clauses of length one can describe the starting state and the set of the final states of the automaton. In the case of a word automaton, Horn clauses of length two can be

*A positive literal is a variable; a negative literal is a negated variable.

used to describe its transition relation, while clauses of length one can describe the starting state and the final states. Then, the question about the emptiness of the language of the automaton can be translated to a question about the satisfiability of the formula. There is also a correspondence between Horn formulae and hypergraphs that we use to show how results on random hypergraphs relate to our research on random Horn formulae.

The probability of satisfiability of random Horn formulae generated according to a variable-clause-length model has been studied by Istrate in [49]. In this work it is shown that according to this model random Horn formulae have a coarse satisfiability threshold, i.e. the problem does not have a phase transition. The variable-clause-length distribution model used by Istrate is better suited if we study Horn formulae in connection to knowledge-based systems [60].

Motivated by the connection between the emptiness of automata language and the Horn satisfiability, we studied the satisfiability of two types of random Horn formulae in conjunctive normal form (CNF) that are generated according to a variation of the fixed-clause-length distribution model. That is, formulae that consist of clauses of length one and three only, and formulae that consist of clauses of length one and two only. We call these problems 1-3-HornSAT and 1-2-HornSAT respectively. We are looking to identify regions in the problems' space where instances are almost surely satisfiable or almost surely unsatisfiable. We are also interested in finding if

the problems exhibit a phase transition, i.e. a sharp threshold.

Notice that the random 1-2-HornSAT problem is related to the random 1-3-HornSAT problem in the same way that random 2-SAT is related to random 3-SAT. That is, as some algorithm searches for a satisfying truth assignment for a random 1-3-Horn formula by assigning truth values to the variables, a random 1-2-Horn formula is created as a subformula of the original formula. This is a result of 3-clauses being simplified to 2-clauses. The relation between random 2-SAT and random 3-SAT is exploited by Achlioptas in [1] to improve on the lower bound for the threshold of random 3-SAT. In this work, Achlioptas uses differential equations to analyze the execution of a broad family of SAT algorithms. The analysis is based on a Markov chain used to trace the number of 2-clauses and 3-clauses as the algorithm executes. Essential role for the success of the analysis plays the already proven sharp satisfiability threshold for random 2-SAT [17, 31, 43]. In our case, there is no such threshold known for the random 1-2-HornSAT that we could possibly use. Not only that, but as we show later in this paper, random 1-2-HornSAT lacks a phase transition. Because of that, we believe that an analysis of the random 1-3-HornSAT based on the differential equations method presented by Achlioptas is not possible.

The 1-2-HornSAT problem can be analyzed with the help of random digraphs [9]. We will show how results on random digraph connectivity, presented by Carp in [52], can be used to model the satisfiability of random 1-2-Horn formulae. These results

can be used to show that there is no phase transition for the 1-2-HornSAT and are matched by our experimental data.

Our experimental investigation on 1-3-HornSAT shows that there are regions where a random 1-3-Horn formula is almost surely satisfiable and regions where is almost surely unsatisfiable. Analysis of the satisfiability percentiles' window and finite-size scaling [71], suggest that there is a "sharp threshold line" between these two regions. As 1-2-HornSAT can be analyzed using random digraphs, 1-3-HornSAT can be analyzed using random hypergraphs. We show that some recent results on random hypergraphs [29] fit well our experimental data. Unlike the data analysis, the hypergraph-based model suggests that the transition from the satisfiable to unsatisfiable regions is a steep function rather than a step function. It is therefore, not clear if the problem exhibits a phase transition, even though we were able to get experimental data for instances of large order.

Our work here also relates to this presented by Kolaitis and Raffill in [57]. There, the authors carried out a search for a phase transition in another NP-complete problem, that of AC-matching. The similarity between their work and ours is that the experimental data provide evidence that both problems have a slowly emerging phase transition. The difference is that in our case, because of the linear complexity of Horn satisfiability, we are able to test instances of Horn satisfiability of much bigger size, than the instances of AC-matching in [57] or actually most of the NP-complete

problems like 3-SAT, 3-colorability etc.

Chapter 2
Random 3-SAT

2.1 Related Work

The fact that the "easy-hard-easy" pattern is quite simplistic is known, though rather under-emphasized, cf. [69, 2]. For example, in the high-density region (above density 5.2), an exponential lower bound on the length of resolution proofs is proved in [18]. This entails an exponential lower bound on the running time of DLL algorithms, implying that the high-density region can, at best, be described as "less hard". (Note that this lower bound does not apply to algorithms that are based on cutting planes or ROBDDs [21, 48, 45].) It is also known that the probability crossover is not the only phase transition involving random 3-SAT and that phase transitions can be solver dependent. In [72], the authors demonstrated experimentally a change from exponentially fast to power law relaxation at around density 3. In [12, 63], the authors proved linear median running time of the pure-literal algorithm at the low-density region (below 1.63) and showed a phase transition at 1.63 for this algorithm. In [39], the authors proved a linear median running time for the GUC heuristic for low-density instances and showed a phase transition near density 3 for this algorithm.

These latter results indicate that the low-density region is indeed in some sense "easy", but they do not establish that complete SAT solvers have polynomial median running time in this region. The analytical results of Franco and his collaborators suggest that

in this region we might expect a polynomial median running time for certain heuristic algorithms, cf. [15], but they do not prove it definitively. In [22], the authors reported linear median running time of Tableau, their SAT solver, for densities 1, 2, and 3, and an exponential median running time for densities 4.26 and 10. In [67], the authors reported linear median running time of their DLL SAT solver for density 3, and an exponential median running time for density 4.26. Neither of these papers, however, systematically explores the dependence on the density of the running time as a function of the order.

The performance of integer-programming algorithms on random SAT instances is studied in [48], but the author did not systematically study how the running time depends on the density of the instances. Similarly, in [10, 44] the authors studied the behavior of ROBDDs on random SAT instances, but did not study how this behavior varies as a function of the density.

While we focus in this paper on the study of collections of fixed-density instances, it would also be interesting to study the behavior of SAT solvers on instances where the order and the density vary simultaneously; for example, the density may increase together with the order. For DLL solvers, the results in [4] show that unless the density increases linearly with the order we should still expect to see exponential running time. Indeed, if one considers a logarithmic increase of the density as a function of the order, then our data (e.g., using Figure 2.1) shows that the median

running time for GRASP is still exponential.

While the finite-size scaling studies in [42, 67] do aim to explore how both the density and the order affect the running time of SAT solvers, they do not reveal the same detailed picture that emerges from our experiments on the density-order quadrant. First, the finite-size scaling studies for SAT are limited to DLL solvers. Second, finite-size scaling studies show a very good fit only around the crossover point, but the fit gets worse as the scale value gets further from it. This makes it very difficult to draw conclusions on the dependence on the order for fixed-density instance in regions with density far below the crossover point. For example, it is not at all clear how one can obtain linear-time behavior at density 3 reported in [67] from their normalized and rescaled results. Finally, the fact that the running time of DLL is exponential in the high-density region and polynomial in the low-density region makes it rather unlikely that the scale factor observed in [67] applies anywhere but very near the crossover point. We elaborate on this point below.

Beame at al. [4] showed that in the high-density region, a certain variant of DLL terminates with high probability within time $2^{an/d+blogn}$, where a and b are some positive constants, n is the order and d is the density of the instance. This matches the exponential lower bound of [18]. Thus, the *normalized* running time (i.e., the ratio of the running time to the running time at the crossover point) is $2^{an(1/d-1/t)}$, where t is the crossover density. Thus, in the high-density region the finite-size scale

factor is $n(1/d - 1/t)$. More generally, let the running time of a SAT solver in the high-density region be $2^{an^b/d^c}$, where a, b, c, and d are positive constants. Then the scale factor will be $n^b(1/b^c - 1/t^c)$. Note that in the high-density region, order and the density have opposing effects. Increasing the order increases the running time, but increasing density decreases the running time. The scale factor of $n^\alpha(d/t - 1)$ proposed in [67] does not reflect this opposition, as it increases with both n and d, which explains why the study in [67] shows a very good fit only around the crossover point.

On the other hand, our experiments, as well as other experiments mentioned above, provide evidence to the fact that in the low-density region the running time of DLL solvers is polynomial, i.e., $f(d)n^e$, where f is some function and e is a positive constant. Thus, the normalized running time is $f(d)n^e/2^{an^b/t^c}$. Therefore, the scale factor $n^\alpha(d/t-1)$ of [67] does not appear to be a reasonable scale factor in this region. For the low-density region, it makes more sense to normalize the running time with respect to some other threshold s in the low-density region. The normalized running time is then $f(d)/f(s)$, which implies that d is the appropriate scale factor in this region. The bottom line regarding finite-size scaling is that running times in the low- and high-density regions are very different functions of density and order. Expecting the same scale factor to work in both regions is unrealistic.

As noted above, since the sharp shift of average-case running time from polynomial

to exponential is solver dependent and occurs well before the crossover point for all the solvers we tested, explanations for this shift cannot be solver independent and cannot center around phenomena observed at the crossover point [67, 65]. In an interesting recent development, Achlioptas, Beame, and Molloy [2] showed that for mixtures of 2-clauses with density $1-\epsilon$ and 3-clauses with density 2.28, which are unsatisfiable with high probability, DLL solvers take an exponential time to refute. If $(1-\epsilon, 2.28)$ 2-clause/3-clause mixtures occur during the solution of a satisfiable 3-SAT instance, then a DLL solver will take time exponential in the order of the instance to solve it. Achlioptas et al. used this to show that a certain DLL solver behaves exponentially at density 3.81. This could also provide an explanation for our observed exponential behavior of GRASP (which is a modified DLL solver) in the low-density region. Cocco and Monasson [20] recently analyzed the computational complexity of random 3-SAT using the $2+p$ SAT problem, in which a clause in chosen to be 3-clause with probability p and a 2-clause with probability $1-p$. They identified a region to the left of the crossover point, where all instances are almost surely satisfiable, and the complexity of a DLL-based algorithm is exponential. As in [2], it is an appropriate mixture of 2-clauses and 3-clauses, that forces the algorithm to build an exponentially large refutation subtree before finding a solution. Cocco and Monasson show that, depending on the starting density of the 3-SAT instance, a heuristic will or will not avoid building this exponentially large subtree. For low enough density the hard

subtree can be avoided with high probability, but at some point it cannot be avoided and we see a transition from polynomial running time to exponential running time. For the GUC heuristic they show that the transition occurs around density 3 (recall that it is known that GUC is linear below 3.003 [15, 39]). All these agree with our observations that the polynomial to exponential behavior occurs in the satisfiable region and is solver-dependent.

2.2 Experimental Setup

Our experimental setup is identical to that of [23, 68]. We generate dn clauses, each by picking three distinct variables (out of n) at random and choosing their polarity uniformly. For each studied point in the $d \times n$ quadrant we generate at least 100 random instances and apply our solvers. Our experiments were run on Sun Ultra 1 machines. As in [68], we chose to focus on median running time rather than mean running time. The difficulty of completing the runs on very hard instances makes it less practical to measure the mean. Furthermore, the median and the mean are typically quite close to each other, except for the regions that display heavy-tail phenomena, where the median and the mean diverge dramatically [67]. It would be interesting to analyze our data at percentiles other than the 50th percentile (the median) (cf. [67]), though a meaningful analysis for high percentiles would require many more sample points than we have in our experiments.

For the statistical analysis and plotting of data, we used MATLAB, which is

an integrated technical computing environment that combines numeric computation, advanced graphics and visualization, and a high-level programming language. The MATLAB (`www.mathworks.com`) functions we used for statistical analysis were:

- *polyfit*, for computing the best linear, quadratic, or cubic fit to the data (or the logarithm of the data) using polynomial regression, and
- *corrcoef*, for computing r^2, the square of correlation (r^2 is the fraction of the variance of one variable that is explained by regression on the other variable) [54].

For each of our data sets we tried to fit:

- a linear curve on the logarithm of both coordinates (notice that this corresponds to a fit of the form $y = ax^c$ to the data)
- a polynomial curve of the form $y = ax^{c'} + b$, where c' is an integer close to c' of the previous fit, and
- a linear curve on the logarithm of the y-coordinate, while keeping the x-coordinate as is (this corresponds to a fit of the form $y = ae^{cx}$ to the data)

For each fit we computed the r^2 as a measurement of the quality of the fit. Unless stated otherwise, for the results reported in this paper, r^2 exceeded 0.98. This establishes high confidence in the validity of the fit of the curve to the data points.

2.3 Random 3-SAT and GRASP

GRASP [61] is a SAT solver that augments the basic backtracking search with a conflict-analysis procedure. In order to cut down on the search space, a dynamic-learning mechanism based on diagnosing the causes of the conflicts is used. By analyzing conflicts and discovering their causes, GRASP can backtrack non-chronologically to earlier levels in the search tree, potentially pruning large portions of the search space. Moreover, by recording the causes of conflicts, GRASP can avoid running into similar conflicts later during the search.

The experiments described in this section were run on a Sun Ultra 1 with a 167MHz UltraSPARC processor and 128MB RAM. Some changes were made to the default GRASP configuration; we increased the maximum number of backtracks allowed to 1,000,000 and the maximum number of conflicts allowed to 2,000,000. CPU time limit was set to 10,800 seconds. These changes were necessary in order to limit the portion of SAT instances on which GRASP aborted. This artificially lowers our measurements of mean running time, but does not affect our measurements of median running time.

The goal of the experiments was to evaluate GRASP's performance on an initial quadrangle of the $d \times n$ quadrant. We explored densities from 0.9 to 15. The order of the instances explored depends on the density:

- Density 0.9: 2000 variables (25 variables per step)

- Densities 1, 2, 3, 3.4, and 3.5: 1000 variables (10 variables per step)
- Density 3.6: 800 variables (10 variables per step)
- Density 3.7: 480 (10 variables per step)
- Density 3.8: 450 variables (10 variables per step)
- Density: 4.26: 170 variables (10 variables per step)
- Density 5: 210 variables (10 variables per step)
- Densities 4, 6-15: 250 variables (10 variables per step)

In Figure 2.1 the median running time is shown on a logarithmic (base 2) scale. (For densities 4.26 and 5 we extrapolated the data up to 250 variables).

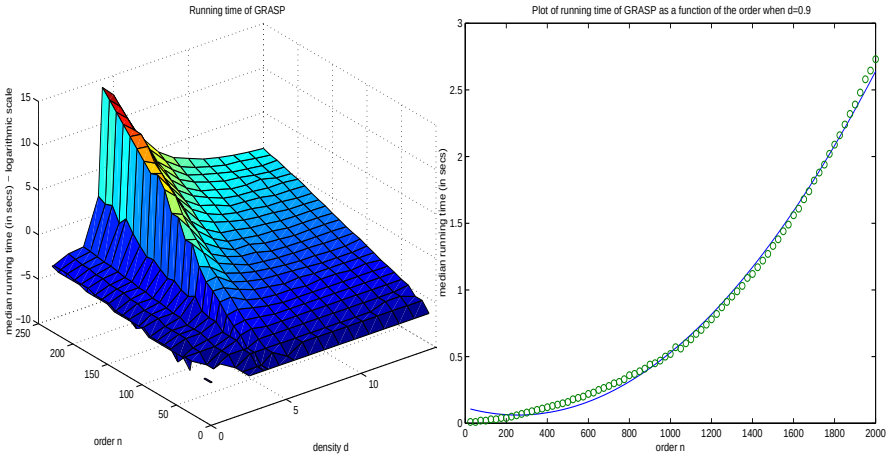

Figure 2.1 GRASP – (left) 3-D Plot of median running time, and (right) median running time for density 0.9 as a function of the order of the instances. A quadratic function fits these points better (with an $r^2 > 0.98$) than an exponential function.

We analyzed the median running time as a function of the order for fixed density instances. For low densities (at or below 3.5), our data indicate a quadratic running time. See Figures 2.1 and 2.2, where we plot the median running time as a function of the order for instances of density 0.9 and 3.5, respectively. The quadratic behavior of GRASP at low densities should be contrasted with the linear running time at low densities that was reported in [22, 67]. It seems that GRASP's conflict-analysis component has a quadratic overhead.

At densities 3.8 and above, the median running time is exponential in the order, i.e., it behaves as $2^{\alpha n}$, where the exponent α depends on d (see discussion below). See Figure 2.2 where we plot the median running time as a function of the order for instances of density 3.8*. Thus, a phase transition seems to occur between densities

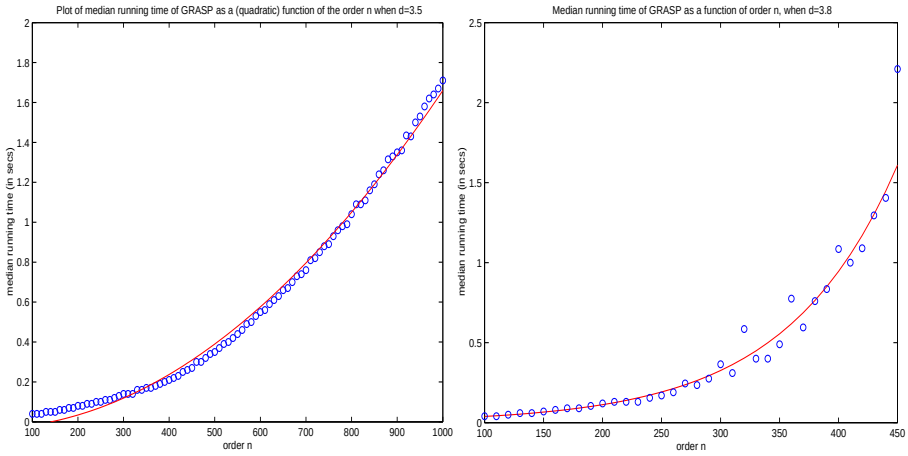

Figure 2.2 GRASP – median running time for density 3.5 (left) and density 3.8 (right) as a function of the order of the instances. At density 3.5, the best fit curve is quadratic in the order, while at 3.8, the best fit curve is exponential in the order.

*The r^2 for this plot is 0.95, while the r^2 for all of the polynomial fits that we tried was ≤ 0.9. That gives us confidence that the running time is exponential in the order.

3.5 and 3.8, where the median running time shifts from polynomial to exponential. As the density is increased beyond 3.8, the exponent α also increases. It peaks at around density 4.26, after which it declines with increased density. Thus, we observe two phase transitions. The second one, in which the exponent reaches its peak, essentially coincides with the crossover point, at which the probability of satisfiability is 0.5. This is the phase transition that was reported at [68] and then studied extensively. This transition, however, is preceded by another one, in some sense a more significant one, near density 3.8, where we observe a qualitative shift in the behavior of GRASP. A transition from polynomial to exponential behavior in graph coloring was conjectured in [47] and counter-conjectured in [27]. Such a transition in random 3-SAT near the crossover point is claimed in [22]; this claim, however, was removed in a later paper [23]. We believe that we are the first to demonstrate such a transition in random 3-SAT, and to show that it occurs significantly below the crossover point. The more recent works of Achlioptas et al. [2] and Cocco and Monasson [20] provide us with an intuition (based on the 2+p-SAT analysis) why such a transition happens to the left of the crossover point for GRASP (see discussion in Section 2.1). Comparing with the results in [20], it seems that GRASP is more successful than GUC in avoiding hard subtrees, pushing the transition from polynomial to exponential to a higher density.

The phase transition near 3.8 is accompanied by a "heavy-tail phenomenon", which is a prevalence of *outliers*, i.e., instances on which the actual running time is

at least an order of magnitude (10) larger than the median running time, as well as a divergence of the mean and the median. See Figure 2.3, where we plot the mean to median ratio and the proportion of outliers as a function of the density. The plots

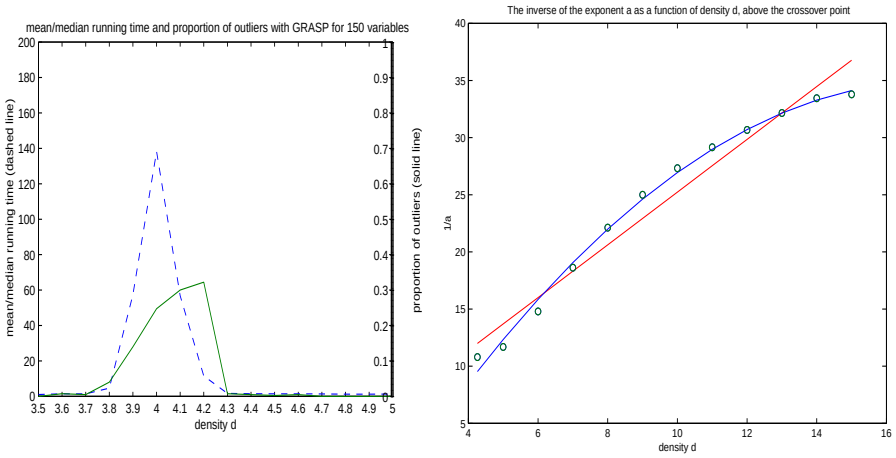

Figure 2.3 GRASP – (left) Ratio of mean to median running time and the proportion of outliers, and (right) the exponent $1/\alpha$ of median running time as a function of density.

show a drastic change in the region between density 3.7 and density 4.3. Both plots show a quick rise and decline. The mean to median ratio peaks at around density 4.0 and the proportion of outliers peaks at around density 4.2. For densities between 3.7 and 4.0 we found it quite difficult to analyze the median running time as a polynomial (of low degree) or exponential function of the order (note the lower r^2 reported above for density 3.8).

There are several ways to analyze inequality, asymmetry and outliers among data. One way to measure inequality within a sample set is to use the mean log deviation GE(0) [74], which has been used extensively to measure inequality in the context of

economic studies of populations and income distributions. It is defined by $\mathrm{GE}(0) = \frac{1}{n}\Sigma_{i=1}^{n}\log(\frac{\bar{y}}{y_i})$, where n is the number of individuals in the sample, y_i is the income of individual i and $\bar{y}$ the mean income. See Figure 2.4 where we plot the mean log deviation for the running time of GRASP against the mean over median runtime ratio, in the density region of 3.5 to 5. Mean log deviation shows a similar behavior with the mean over median ratio, and also with the number of outliers (see Figure 2.3). It peaks at density 4 and it indicates a dramatic increase of inequality to the left of the crossover point. Also, kurtosis and skewness [54] can be used to show whether the data are peaked or flat relative to a normal distribution and whether data are symmetrical or skewed to the right or left. Calculations of kurtosis and skewness on the running time data of GRASP show them both to rise quickly and then quickly drop again; they both peak at around density 3.6. These indicates, as the rest of the statistics reported in this paper, that a significant number of outliers appears between densities 3.5 and 4.

Our data suggest that as the density increases from 3.7 to 4.3, random 3-SAT formulas go through a series of changes and perhaps more than one phase transition. The heavy-tail phenomenon for random 3-SAT deserves further study (with many more samples per point in the $d \times n$ quadrant) to confirm our findings. In particular, the divergence of the two peaks in Figure 2.3 needs to be reconfirmed or refuted.

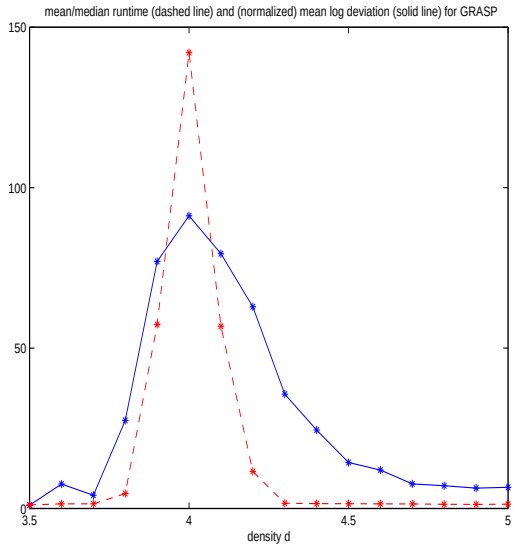

Figure 2.4 Mean log deviation vs. mean over median ratio for running time of GRASP.

As noted above, beyond density 4.26 the exponent α declines. A theoretical analysis suggests that for DLL solvers α may decline inversely linearly, i.e., as $\frac{c}{d}$, for some constant c, see [4]. Our data, however, suggest a slower decline, even though one may expect GRASP to be faster than DLL solvers. See Figure 2.3, where we plot $\frac{1}{\alpha}$ as a function of d. Thus, GRASP is not as efficient in the high-density region as it could be. (We should caution, however, that we only have 11 data points, and these data points themselves have been obtained by fitting a linear curve to the logarithm of the median running time. Thus, the finding of a slower decline should be viewed as quite preliminary.)

2.4 Random 3-SAT and CPLEX

The CPLEX MIP Solver is a commercial linear-programming solver for integer variables. It employs a branch-and-bound technique starting from a linear-programming relaxation of the given integer-programming problem. This may be complemented with the dynamic generation of cutting planes [7, 8].

The experiments described in this section were run on a Sun Ultra 1 with a 167MHz UltraSPARC processor and 64 MB RAM. SAT problems were encoded as 0-1 integer-programming problems. Values true and false are represented as 1 and 0. For a clause to be true the sum of the representations of the literals has to be greater or equal to 1. For example, the clause $\neg x_1 \vee x_2 \vee \neg x_3$ is represented by the inequality $(1 - x_1) + x_2 + (1 - x_3) \geq 1$.

We used CPLEX to solve problems for densities from 0.9 to 15. The order of the instances was chosen according to the density:

- Densities 0.9, 1.5, 1.6, 1.7 and 1.8: 2000 variables (25 variables per step)
- Density 1: 10000 variables (50 variables per step)
- Density 2: 1800 variables (25 variables per step)
- Density 2.5: 460 variables (10 variables per step)
- Density 3: 250 variables (10 variables per step)

- Density 3.5: 150 variables (10 variables per step)
- Densities 4, 4.26, and 5-15: 120 variables (10 variables per step)

In Figure 2.5, the median running time is shown on a logarithmic (base 2) scale. Note that the peak at the crossover point is much less pronounced than the one in Figure 2.1.

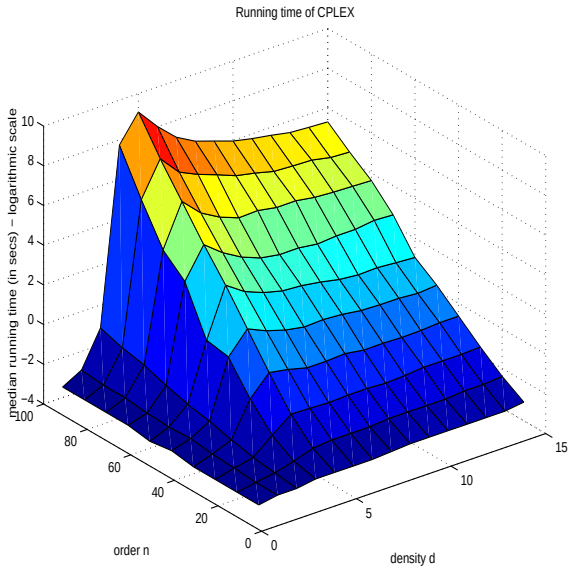

Figure 2.5 CPLEX – 3-D Plot of median running time

The median running time was analyzed as a function of the order for fixed density-instances. For low densities (below 1.7) our data indicate a linear running time. See Figure 2.6 for median running times for instances of density 1 with up to 10000 variables. For density 2 the median running time is quadratic, while for density 2.5 the running time is cubic. See Figure 2.6 for median running time for instances of density 2, where for order above 400 the behavior is quadratic. See Figure 2.7 (left)

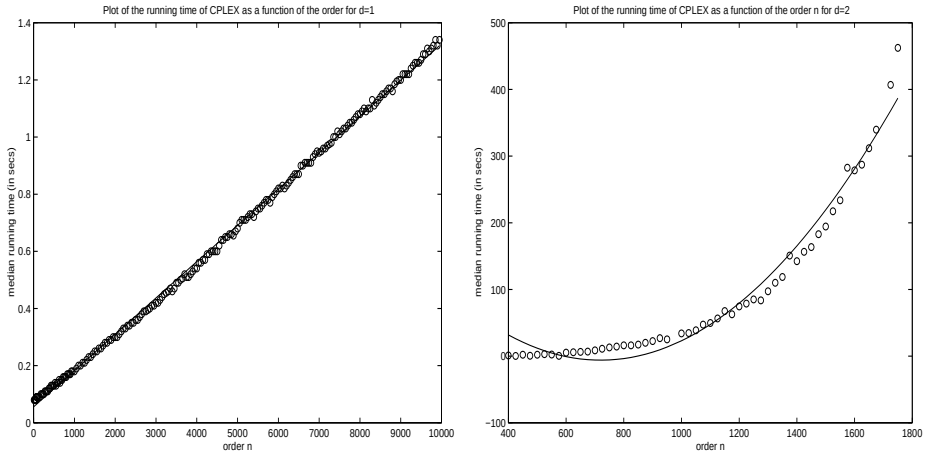

Figure 2.6 CPLEX – median running time for density 1 (left) and density 2 (right) as a function of the order of the instances.

for median running time for instances of density 2.5, where the behavior is cubic. Thus we seem to have two phase transitions, corresponding to a shift from a linear

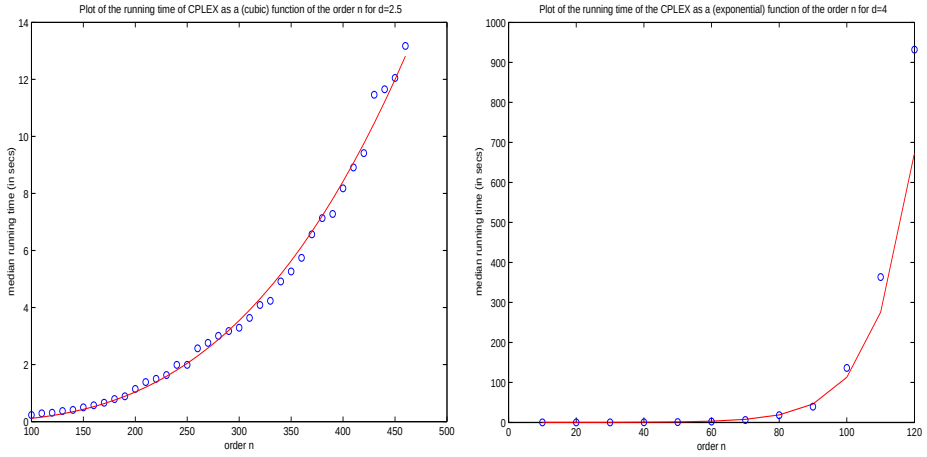

Figure 2.7 CPLEX – median running time for density 2.5 (left) and density 4 (right) as a cubic and exponential respectively function of the order of the instances.

to quadratic behavior between densities 1 and 2 and a subsequent shift to a cubic behavior between densities 2 and 2.5. The first shift may coincide with the phase transition proved in [12, 63] around density 1.63, as described in Section 2.1.

At densities 4.0 and above, see Figure 2.7 (right), the median running time is exponential in the order, i.e., it behaves as $2^{\alpha n}$, where the exponent α depends on d. As with GRASP, a phase transition seems to occur between densities 2.5 and 4.0. It corresponds to the shift from polynomial to exponential behavior. Again, we believe that as with GRASP, this shift is related to the 2+p-SAT results in [2, 20]. Note that the polynomial to exponential running time shift for CPLEX is happening in the same region (near density 3.0) that the shift for GUC is happening. While CPLEX is a branch-and-bound technique (that can be related to DLL-like heuristics), it also uses cutting-planes technique. Unfortunately, as CPLEX is a commercial tool, we have little access to its underlying algorithms and heuristics, which makes it difficulty to offer a precise analysis of its behavior.

As with GRASP, the polynomial-to-exponential transition is accompanied by heavy-tail phenomena. See Figure 2.8, where we plot the mean to median ratio and the proportion of outliers as a function of the density. Note that the heavy-tail phenomenon for CPLEX is not as marked as with GRASP; both peak mean-to-median ratio and peak proportion of outliers are lower for CPLEX than for GRASP. Note also that the peak for CPLEX occurs at lower densities (around 3.6) than for GRASP (around 4.0).

As with GRASP, the exponent α peaks at density 4.26 and then declines. Again, our data show a slower decline than $\frac{c}{d}$, as suggested in [4] (though the analysis there

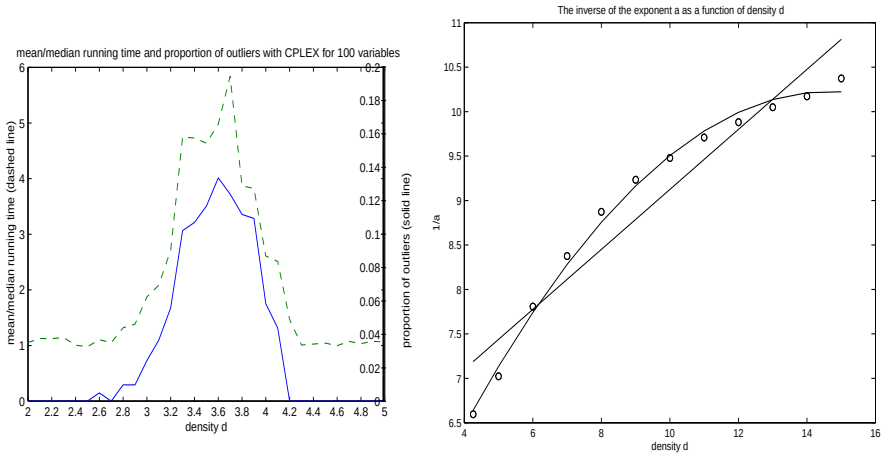

Figure 2.8 CPLEX – (left) Ratio of mean to median running time and proportion of outliers, and (right) the exponent $1/\alpha$ of median running time as a function of density.

is for resolution-based procedures, which are weaker than the cutting-planes method used in CPLEX.) See Figure 2.8, where we plot $\frac{1}{\alpha}$ as a function of d.

2.5 Random 3-SAT and CUDD

CUDD [70] is a package that provides functions for the manipulation of Boolean functions, based on the reduced, ordered, binary decision diagram (ROBDD) representation [13]. A binary decision diagram (BDD) is a rooted directed acyclic graph that has only two terminal nodes labeled 0 and 1. Every non-terminal node is labeled with a Boolean variable and has two outgoing edges labeled 0 and 1. An ordered binary decision diagram (OBDD) is a BDD with the constraint that the input variables are ordered and every path in the OBDD visits the variables in ascending order. An ROBDD is an OBDD where every node represents a distinct logic function.

Unlike GRASP and CPLEX, CUDD does not search for a satisfying truth as-

signment. Rather, it constructs a compact symbolic representation of the set of *all* satisfying truth assignments. Then, the resulting ROBDD is compared against the predefined constant 0 in order to find if an instance is (un)satisfiable. It is important to note that very large sets of truth assignments can have very compact ROBDD representation [13], which explains the effectiveness of ROBDDs in hardware verification [14, 50]. As we see later, CUDD performs well in the very-low-density region, where the set of satisfying truth assignment is very large.

The experiments described in this section were run on a Sun Ultra 1 with a 167MHZ UltraSPARC processor and 64MB RAM. The CUDD package has been used through the GLU C–interface [73], a set of low-level utilities to access BDD packages. It is well known that the size of the ROBDD for a given function depends on the variable order chosen for that function. We have used automatic dynamic reordering during the tests with the default method for automatic reordering of CUDD.

As in the preceding two sections, the goal of the experiments was to evaluate CUDD's performance on an initial quadrangle of the $d \times n$ quadrant. We explored densities 0.1, 0.5, and 1 to 15. The order of the instances explored depends on the density:

- Density 0.1: 1480 variables (10 variables per step)
- Density 0.5: 136 variables (2 variables per step)

- Density 0.9 and 1: 68 variables (2 variables per step)
- Densities 1.5, 2-4, 4.26, 5-15: 46 variables (2 variables per step)

In Figure 2.9 the median running time is shown on a logarithmic (base 2) scale. Note the absence of a peak (contrast with Figures 2.1 and 2.5).

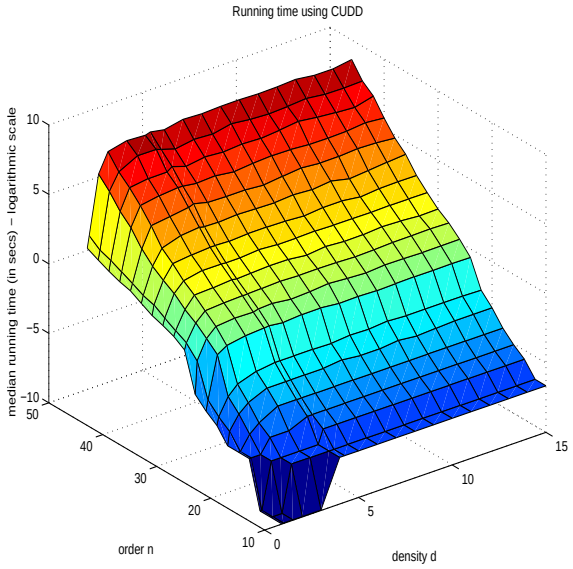

Figure 2.9 CUDD – 3-D Plot of median running time

We analyzed the median running time as a function of the order for fixed-density instances. At densities 0.5 and above, the median running time is exponential in the order, i.e., it behaves as $2^{\alpha n}$. See Figure 2.10 (right) for median running time for instances of density 1, where the behavior is exponential. At density 2 and above the exponent α is independent of the density. In particular, there seems to be nothing special about the crossover point at density 4.26. The explanation for this behavior is that the running time of ROBDD-based algorithms is determined mostly by the

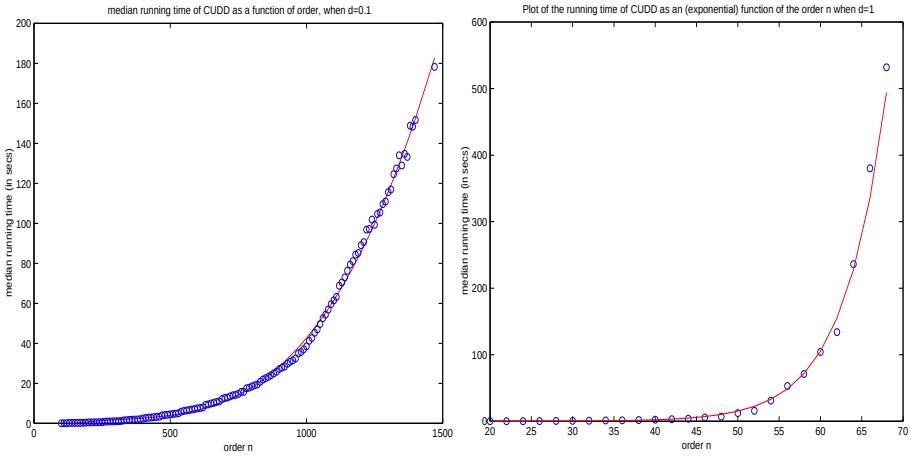

Figure 2.10 CUDD – median running time for density 0.1 (left) and for density 1 (right) as a cubic and an exponential respectively function of the order of the instances.

size of the manipulated ROBDDs. Our algorithm involves dn product operations between a possibly large ROBDD (representing all truth assignments of the clauses processed so far) and a small ROBDD (representing seven truth assignments of the currently processed clause). Thus, the running time of our algorithm is determined by the largest intermediate ROBDD constructed. As is shown in Figure 2.11, the peak in ROBDD size is attained after processing about $2n$ clauses, which explains the flattening of the running-time plot at density 2, and suggests that a phase transition in terms of ROBDD size occurs at about this density.

As ROBDDs are symmetrical with respect to the set they represent and its complement, both very small sets and very large sets can be represented by small ROBDDs [13]. This suggests that we may see polynomial behavior for very low density instances, which have a large number of satisfying truth assignments. To check this conjecture we measured the median running time of CUDD for instances of density

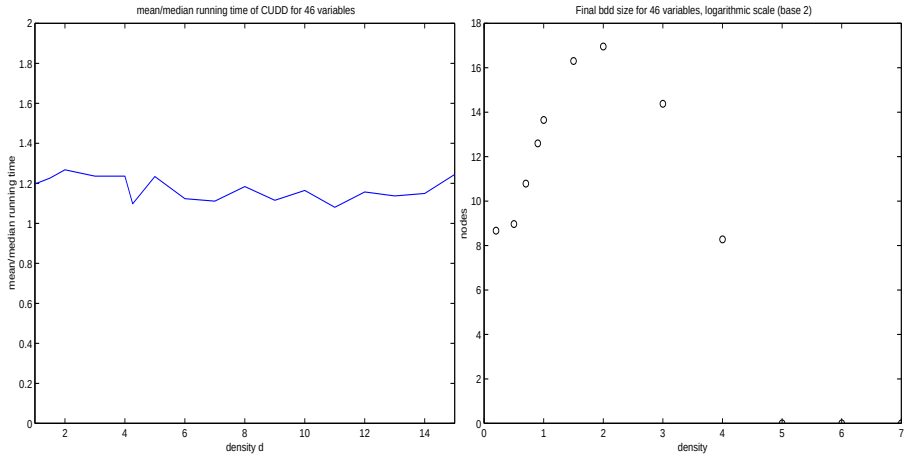

Figure 2.11 CUDD – (left) Ratio of mean to median running time and (right) median ROBDD size as a function of density

0.1. Our results indicate a cubic-time behavior, see Figure 2.10 This suggests the existence of another phase transition between densities 0.1 and 0.5. This result should be contrasted with that of [69], in which the running time for explicitly enumerating all solutions of random constraint-satisfaction instances increases as the density decreases.

Unlike with GRASP and CUDD, we did not observe a heavy-tail phenomenon with CUDD: there are no outliers and the mean to median ratio is independent of the density (see Figure 2.11).

Chapter 3
3-Colorability of random graphs

3.1 Experimental Setup

In order to study the average-case complexity of the 3-colorability of random graphs, we used the $G(n, m)$ random model. For each instance of order n, we select uniformly at random and with replacement m edges (out of all $\frac{n(n-1)}{2}$ possible edges). This model is common when analyzing phase transitions [47, 26, 35]. For each point we study in the $\gamma \times n$ quadrant, we generate and solve 200 random instances according to the $G(n, m)$ model. We use SMALLK to solve those instances. Our experiments were run on Sun Ultra 1 machines.

For the statistical analysis and plotting of the data, we used MATLAB. As with the 3-SAT experiments, we tried to fit a linear curve on the log-log data, a linear curve on the semi-log, and a polynomial curve on the plain data (see section 2.2 for more details). The MATLAB functions that were most useful for our purposes are polyfit (for computing the best polynomial fit) and corrcoef (to estimate the correlation between the actual data and the fit). For each fit, we compute the r^2 and we report it here. In most cases the r^2 is at least 0.98; a sign that the we get a good fit.

3.2 3-Colorability of random graphs: experimental results

For the purposes of our study, we explored connectivities from 1 to 20. The maximum order of the instances explored varies with the connectivity. In details:

- Densities 1-4: 580 vertices
- Densities 4.1, 4.2, 4.3, and 4.4: 1000 vertices
- Density 4.5: 580 vertices
- Density 4.6: 520 vertices
- Density 4.7: 480 vertices
- Densities 5-20: 580 vertices

The increment on the order of the instances was 20 vertices per step.

In Figure 3.1 (left) the percentage of the colorable graphs across the $\gamma \times n$ quadrant is shown. Below connectivity 4 a random instance is almost surely colorable, while above connectivity 5 an instance is almost surely non-colorable. When the connectivity is between 4 and 5, there is a steep transition on the colorability probability*. Although these observations are old news, we believe that this is the first time that the colorability probability (and the average-case complexity of 3-colorability that we will

*The actual 3-colorability threshold for the G(n, m) random graph model is believed to be at connectivity 4.6. In the plot in Figure 3.1 only the data for densities $1, 2, 3, 4, 4.5, 5, \cdots, 20$ are presented.

discuss shortly) has been reported while systematically varying both the connectivity and the order of the random instances.

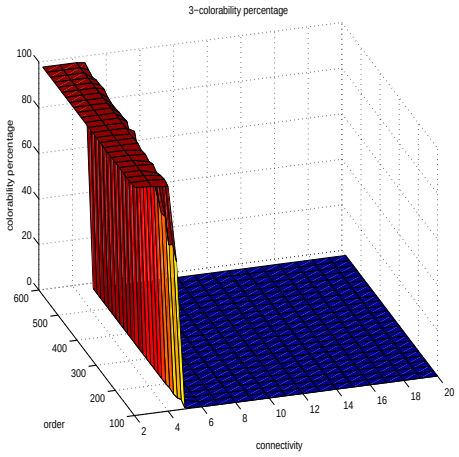

Figure 3.1 3-D plot of the percentage of the colorable instances across the $\gamma \times n$ quadrant.

The median and mean running time of SMALLK on a logarithmic (base 10) scale is shown in Figure 3.2.

We analyzed the median running time of SMALLK as a function of the order for instances of fixed density. Below connectivity 4, we observe a polynomial (quadratic) median running time. We also know that above the colorability threshold, the complexity of the problem is exponential in the order. We then focus on the connectivity range of $[4, 4.7]$. Our data indicate that the running time is quadratic up to connectivity 4.2, while from connectivity 4.4 and above the running time is exponential. See Figure 3.3 where we plot the median running time of SMALLK for connectivities 4.2

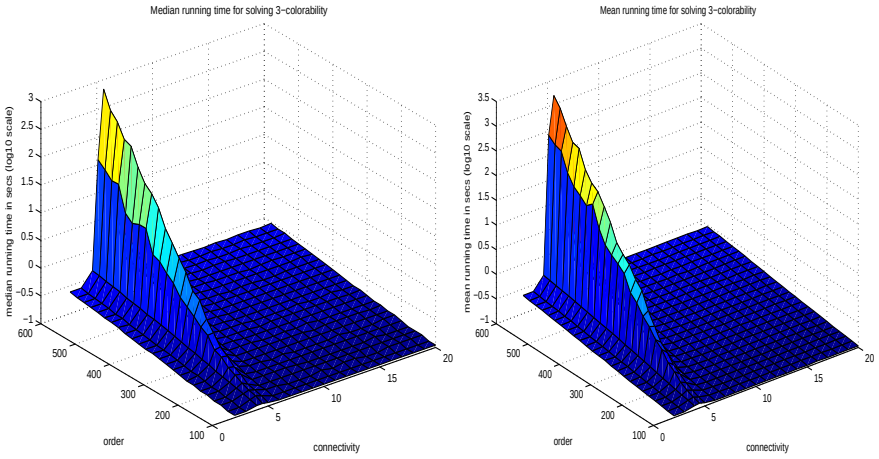

Figure 3.2 3-D plot of the median (left), and mean (right) running time of SMALLK.

and 4.4*. So, we observe a transition from polynomial to exponential median running time of SMALLK happening around connectivity 4.3. This is the phase transition that was conjectured in [47] and counter-conjectured in [26].

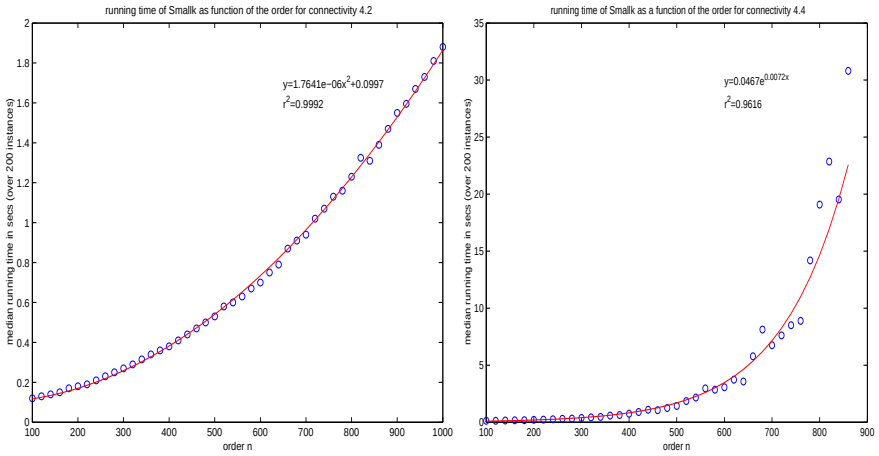

Figure 3.3 Median running time of SMALLK for connectivity 4.2 (left) and 4.4 (right). At connectivity 4.2 the best fit curve is quadratic in the order, while at connectivity 4.4 the best fit curve is exponential in the order.

*The r^2 for this fit is 0.96, while the r^2 for all the polynomials we tried was ≤ 0.84. That gives us confidence that the running time for the particular connectivity is exponential in the order.

In Chapter 2 we show a similar polynomial to exponential phase transition for the 3-SAT average-case complexity. There we also show that, such a phase transition is accompanied by a "heavy-tail phenomenon". This is also the case for the phase transition observed for 3-colorability. See Table 3.1 where we present the mean over median ratio, and the percentage of outliers (recall that our definition of an outlier is an instance for which the running time is at least an order of magnitude larger than the median) for connectivity $4.1 \leq \gamma \leq 4.7$. Like in 3-SAT, both the ratio of mean over median running time, and the number of outliers, start increasing around the point where the phase transition is happening (in our case around connectivity 4.3), they peak just below the threshold, and quickly decrease after it. The region of the heavy-tail phenomenon for the 3-colorability is narrower than the one for 3-SAT (see Figures 2.3 and 2.8).

connectivity γ	mean/median	% outliers
4.1	1.005	0
4.2	1.051	0
4.3	1.612	2
4.4	3.281	5
4.5	6.401	15.5
4.6	2.918	6.5
4.7	1.990	1.5

Table 3.1 Heavy-tail phenomenon at the phase transition.

Chapter 4
Random 1-3-HornSAT

4.1 Preliminaries

Our main motivation for studying the satisfiability of Horn formulae is that, unlike 3-SAT, this problem is tractable. Therefore we will have data for instances of much larger order to help us answer questions similar to those previously asked about 3-SAT.

Apart from that, it is of interest to us that Horn formulae can be used to describe finite automata. A finite automaton A is a 5-tuple $A = (S, \Sigma, \delta, s, F)$, where S is a finite set of states, Σ is an alphabet, s is a starting state, $F \subseteq S$ is the set of final (accepting) states and δ is a transition relation.

In a word automaton, δ is a function from $S \times \Sigma$ to 2^S. In a binary-tree automaton δ is a function from $S \times \Sigma$ to $2^{S \times S}$. A run of an automaton on a word $a = a_1 a_2 \cdots a_n$ is a sequence of states $s_0 s_1 \cdots s_n$ such that $s_0 = s$ and $(s_{i-1}, a_i, s_i) \in \delta$. A run is succesful if $s_n \in F$; in this case we say that A accepts the word a. A run of an automaton on a binary tree t labeled with letters from Σ, is a binary tree r labeled with states from S such that $\text{root}(r) = s$ and for a node i of t, $(r(i), t(i), r(\text{left-child-of-}i), r(\text{right-child-of-}i)) \in \delta$. A run is succesfull if for all leaves l of r, $r(l) \in F$; in this case we say that A accepts the tree t. The language $L(A)$ of a word (resp. tree) automaton A, is the set of all words a (resp. trees t) for which there

is a succesful run of A on a (resp. t). An important question on automata theory that also is of great practical importance in the field of formal verification [77] is, given an automaton A is $L(A)$ non-empty ? We can show how the problem of non-emptiness of automata language translates to Horn satisfiability.

Consider first a word automaton $A = (S, \Sigma, \delta, s_0, F)$. Construct a Horn formula ϕ_A over the set S of variables as follows:

- create a clause $(\bar{s_0})$
- for each $s_i \in F$ create a clause (s_i)
- for each element (s_i, a, s_j) of δ create a clause $(\bar{s_j}, s_i)$,

where $(s_i, \cdots, s_k)$ represents the clause $s_i \vee \cdots \vee s_k$ and $\bar{s_j}$ is the negation of s_j.

Theorem 1 *Let A be a word automaton and ϕ_A the Horn formula constructed as described above. Then $L(A)$ is non-empty if and only if ϕ_A is unsatisfiable.*

Proof. ($\Rightarrow$) Assume that $L(A)$ is non-empty, i.e. there is a path $\pi = s_{i_0} s_{i_1} \cdots s_{i_m}$ in A such that $s_{i_0} = s_0$ and $s_{i_m} = s_k$ where s_k is a final state. Since s_k is a final state (s_k) is a clause in ϕ_A. Also $(\bar{s_k}, s_{i_{m-1}})$ is a clause in ϕ_A. For ϕ_A to be satisfiable s_k should be true and consequently, $s_{i_{m-1}}$ must be true. By induction on the length of the path π we can show that for ϕ_A to be satisfiable s_0 must be true, which is a contradiction.

($\Leftarrow$) Assume that ϕ_A is unsatisfiable. Because of the way we constructed ϕ_A, the only way for this to happen is if s_0 is required to take the value true (and thus create a contradiction with the clause $(\bar{s_0})$). If ϕ_A is unsatisfiable, then it has a positive-unit resolution refutation [46], i.e. a proof by contradiction where in each step one of the resolvents must be a positive literal. Let (s_i) be the first positive literal resolvent in the proof. By construction, s_i is a final state of A. We can construct a path in A from s_0 to s_i, using the resolution refutation of ϕ_A. Therefore, $L(A)$ is non-empty. $\Box$

Similarly to the word automata case, we can show how to construct a Horn formula from a binary tree automaton. Let $A = (S, \Sigma, \delta, s_0, F)$ be a binary tree automaton. Then we can construct a Horn formula ϕ_A using the construction above with the only difference that since δ in this case is a function from $S \times \{\alpha\}$ to $S \times S$, for each element (s_i, α, s_j, s_k) of δ, we create a clause $(\bar{s_j}, \bar{s_k}, s_i)$. It is not difficult to see that also in this case we have,

Theorem 2 *Let A be a binary tree automaton and ϕ_A the Horn formula constructed as described above. Then $L(A)$ is non-empty if and only if ϕ_A is unsatisfiable.*

Motivated by the connection between tree automata and Horn formulas described in Theorem 2 we studied the satisfiability of two types of random Horn formulae. More precisely:

Let $H^{1,2}_{n,d_1,d_2}$ denote a random formula in CNF over a set of variables $X = \{x_1, \cdots, x_n\}$ that contains:

- a single negative literal chosen uniformly among the n possible negative literals
- $d_1 n$ positive literals that are chosen uniformly, independently and without replacement among all $n-1$ possible positive literals (the negation of the single negative literal already chosen is not allowed)
- $d_2 n$ clauses of length two that contain one positive and one negative literal chosen uniformly, independently and without replacement among all $n(n-1)$ possible clauses of that type.

We call the number of variables n the *order* of the instance. Let also $H^{1,3}_{n,d_1,d_3}$ denote a random formula in CNF over the set of variables $X = \{x_1, \cdots, x_n\}$ that contains:

- a single negative literal chosen uniformly among the n possible negative literals
- $d_1 n$ positive literals that are chosen uniformly, independently and without replacement among all $n-1$ possible positive literals (the negation of the single negative literal already chosen is not allowed)

The sampling spaces $H^{1,3}$ and $H^{1,2}$ are slightly different; we sample with replacement in the first, and without replacement in the second. We explain here why there is this difference. Assume that we sample dn clauses out of N uniformly at random with replacement. Let us consider the (asymptotic) expected number of distinct clauses we get. Each one of the N clauses will be chosen with probability $1-(1-\frac{1}{N})^{dn}$. The expected number of distinct chosen clauses is $N(1-(1-\frac{1}{N})^{dn})$. Notice that $N(1-(1-\frac{1}{N})^{dn}) \approx N(1-\exp\frac{-dn}{N}) = N(1-(1-\frac{dn}{N}+O((\frac{dn}{N})^2))) = dn - O(\frac{(dn)^2}{N})$. In the case of a random $H^{1,3}_{n,d_1,d_3}$ formula $N = \frac{n(n-1)(n-2)}{2}$ and clearly the expected number of distinct clauses we sample is asymptotically equivalent to dn; thus we sample with replacement for practical reasons. In the case of a random $H^{1,2}_{n,d_1,d_2}$ formula we sample without replacement to ensure that we do not have many repetitions among the chosen clauses.

- $d_3 n$ clauses of length three that contain one positive and two negative literals chosen uniformly, independently and with replacement*among all $\frac{n(n-1)(n-2)}{2}$ possible clauses of that type.

4.2 On the 1-2-HornSAT

In this section we present our results on the probability of satsifiability of random 1-2-Horn formulae. We first present an experimental investigation of the satisfiability on the $d_1 \times d_2$ quadrant. We then discuss the relation between random 1-2-Horn formulae and random digraphs and show that our data agree with analytical results on graph reachability presented in [52].

We studied the probability of satisfiability of $H^{1,2}_{n,d_1,d_2}$ random formulae in the $d_1 \times d_2$ quadrant. We generated and solved 1200 random instances of order 20000 per data point. See Figure 4.1 where we plot the average probability of satisfiability against the two input parameters d_1 and d_2 (left) and the corresponding contour plot (right).

The satisfiability plot shown in Figure 4.1 indicates that the problem does not have a phase transition. This can also been observed if we fix the value of one of the input parameters. See Figure 4.2, where we show the satisfiability plot for random 1-2-HornSAT for various order values ranging from 500 to 32000, and for fixed $d_1 = 0.1$. We now explain why random 1-2-HornSAT does not have a phase transition, based on known results on random digraphs.

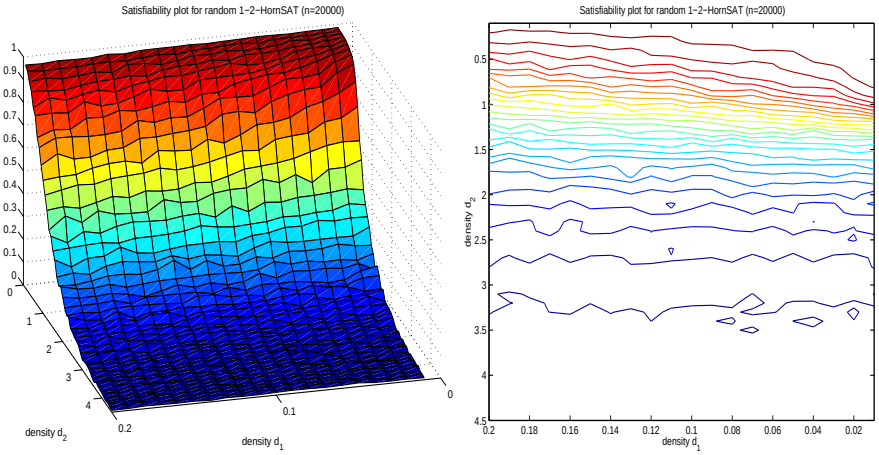

Figure 4.1 Average satisfiability plot of a random 1-2-Horn formula of order=20000 (left) and the corresponding contour plot (right).

There are two most frequently used models of random digraphs. The first one, $G(n, m)$ consists of all digraphs on n vertices having m edges; all digraphs have equal probability. The second model, $G(n, p(\text{edge}) = p)$ with $0 < p < 1$, consists of all digraphs on n vertices in which the edges are chosen independently with probability p. It is known that in most investigations the two models are interchangeable, provided certain conditions are met. In what follows, we will take advantage of this equivalence in order to show how our experimental results relate to analytical results on random digraphs [52].

We will first show that there is a relation between the satisfiability of a random $H^{1,2}_{n,d_1,d_2}$ formula and the vertex reachability of a random digraph $G(n, d_2 n)$. Let $\phi \in H^{1,2}_{n,d_1,d_2}$, $(\bar{x_0})$ be the unique single negative literal in ϕ, and F be the set of all variables that appear as single positive literals in ϕ. Obviously $|F| = d_1 n$. Construct a graph G_ϕ such that for every variable x_i in ϕ there is a corresponding node v_i in

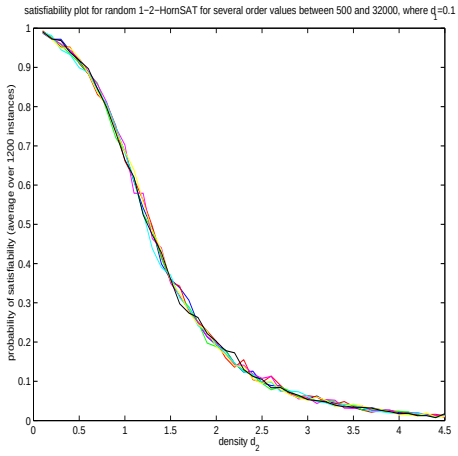

Figure 4.2 Satisfiability plot of random 1-2-Horn formulae when $d_1 = 0.1$

G_ϕ and for each clause $(\bar{x}_i, x_j)$ of ϕ there is a directed edge in G_ϕ from v_i to v_j. G_ϕ is a random digraph from the $G(n, d_2 n)$ model.

It is not difficult to see that ϕ is unsatisfiable if and only if the node v_0 in G_ϕ is reachable from a node v_i such that $x_i \in F$. In other words, the probability of unsatisfiability of a random $H^{1,2}_{n,d_1,d_2}$ formula ϕ, is equal to the probability that a vertex of the random digraph $G(n, d_2 n)$ is reachable from a set* of vertices of size $d_1 n$.

As mentioned above the $G(n, m)$ and $G(n, p((edge) = p))$ models can be used interchangeably, when $m \approx \binom{n}{2} p$ [9]. Therefore, the relation we established between the satisfiability of a random $H^{1,2}_{n,d_1,d_2}$ formula ϕ and the vertex reachability of a random digraph $G(n, d_2 n)$, holds also between ϕ and a random digraph $G(n, p = \frac{d_2}{n})$.

*A vertex is reachable from a set of vertices if it is reachable by at least one of the vertices of the set.

The vertex reachability of random digraphs generated according to the model $G(N,p)$ has been studied and analyzed by Karp in [52]. We use his results to study the satisfiability of random $H^{1,2}_{n,d_1,d_2}$ formulae. Karp showed that as n tends to infinity, when $np < 1-h$, where h is a fixed small positive constant, the expected size of a connected component of the graph is bounded above by a constant $C(h)$. When $np > 1+h$, as n tends to infinity, the set of vertices reachable from one vertex is either "small" (expected size bounded above by $C(h)$) or "large" (size close to Θn, where Θ is the unique root of the equation $1-x-e^{-(1+h)x}=0$ in $[0,1]$). Moreover, a giant strongly component emerges of size approximately $\Theta^2 n$.

Let us now consider the two cases; $d_2 = 1-h$ and $d_2 = 1+h$, where h is a psoitive number. Remember that in our case $p = \frac{d_2}{n}$. In the analysis below we use the notation w.h.p. (with high probability) as shorthand for "with probability tending to 1 at the limit".

In the case where $d_2 = 1-h$, that is $np < 1-h$, the size of the set $X(v_i)$ of vertices reachable by a vertex v_i is w.h.p. less than or equal to $3\ln n h^{-2}$, and the expected size of this set is bounded above by a constant related to h. Thus we get that the probability that v_0 is reachable by v_i w.h.p. lies in the interval $[0, \frac{3\ln n}{n(1-d_2)^2}]$, and its expected value is bounded above by a constant. The expected probability that v_0 is reachable by a set of $d_1 n$ vertices should increase with the d_1. See the plots in Figures 4.1 and 4.2, where it shows that the probability of satisfiability of ϕ (which

is 1 minus the probability that v_0 is reachable by a set of $d_1 n$ vertices in G_ϕ), while $d_2 < 1$, is decreasing as we increase d_2 and/or d_1.

When $d_2 = 1 + h$, that is $np > 1 + h$, we know that the set $X(v_i)$ of vertices reachable by a vertex v_i is w.h.p. either in the interval $[0, \frac{3 \ln n}{(1-d_2)^2}]$, or around Θn. We also know that the probability that $X(v_i)$ is "small" tends to $1 - \Theta$. Therefore, w.h.p. at least one of the $d_1 n$ vertices will have a "large" reachable set. That is, the probability that v_0 is reachable by a set of $d_1 n$ vertices is bounded below from Θ. Notice that Θ increases with d_2. Again, see the plots in Figures 4.1 and 4.2 where we can see that the probability of satisfiability of ϕ when $d_2 > 1$ is decreasing as d_2 increases. So the experimental observations are in agreement with the expectations based on the digraph reachability analysis.

Going back to digraphs' reachability, Karp's results show that for each vertex the set of its reachable vertices is very small up to the point where $np = 1$. We can observe the same behaviour in 1-2-HornSAT if we change our distribution model by setting $d_1 = c/n$ for some constant c. By doing that, we are adjusting our model to fit the reachability analysis done by Karp that is based on a single starting vertex in the digraph. The result of this modification is that d_1 is no longer a factor on the probability of satisfiability of ϕ, that is solely now depends on d_2. See Figure 4.3, where we show the satisfiability plot in that case, and contrast with the picture that emerges when d_1 is a constant (shown in Figure 4.2). While before the satisfiability

probability was steadily decreasing as we increased d_2, now the satisfiability probability is practically 1, until d_2 gets a value bigger than one. In both cases, however, the reachability analysis and the experimental data show that the satisfiability of random 1-2-Horn formulae is a problem that lacks a phase transition.

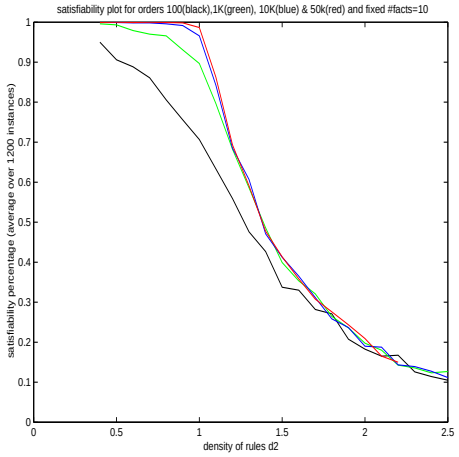

Figure 4.3 Satisfiability plot of random 1-2-Horn formulae when $d_1 = 10/n$ for orders 100(black), 1000 (green), 10000(blue), and 50000(red).

4.3 On the 1-3-HornSAT

In this section we present our results on the probability of satisfiability of random 1-3-Horn formulae. We first present a thorough experimental investigation of the satisfiability on the $d_1 \times d_3$ quadrant. We then show that analytic results on vertex identifiability in random hypergraphs [29] fit well our results on the satisfiablity of random 1-3-Horn formulae.

We studied the probability of satisfiability of $H^{1,3}_{n,d_1,d_3}$ random formulae in the

$d_1 \times d_3$ quadrant. We generated and solved 3600 random instances of order 20000 per data point. See Figure 4.4 where we plot the average probability of satisfiability against the two input parameters d_1 and d_3 (left) and the corresponding contour* plot (right).

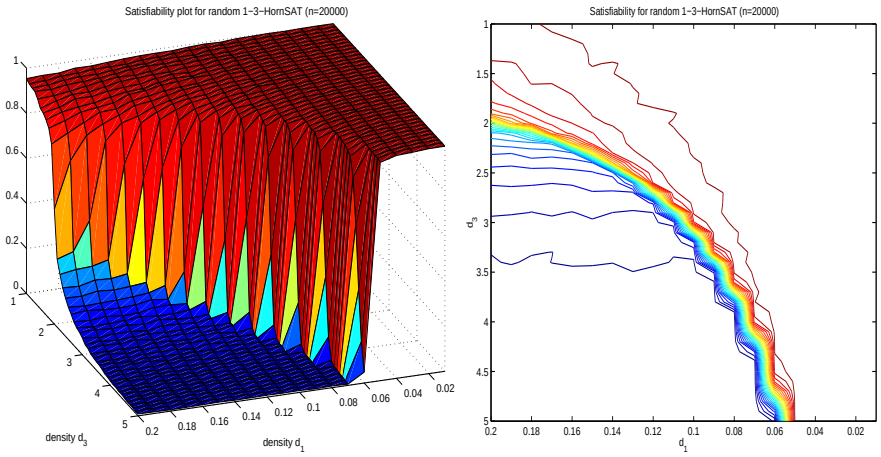

Figure 4.4 Average satisfiability plot of a random 1-3-Horn formula of order=20000 (left) and the corresponding contour plot (right).

From our experiments we see that there is a region where the formula is under-constrained (small values of d_1 and d_3) and the probability of satisfiability is almost 1. As the values of the two input parameters increase, there is a rapid change in the satisfiability terrain, what we call the *waterfall*. As the values of d_1 and d_3 cross some boundaries (the projection of the waterfall shown in the contour plots) the probability of satisfiability becomes almost 0. In other words, we observe a transition similar to these observed in other combinatorial problems like 3-SAT, 3-coloring etc.

*In this plot there are 25 lines that separate consecutive percentages intervals, i.e. $[0\% - 4\%), [4\% - 8\%), \cdots, [96\% - 100\%]$.

There is a significant difference though, between these previously studied transitions and the one we observe in 1-3-HornSAT. In cases like 3-SAT or 3-colorability there are two input parameters describing a random instance; the order and the constrainedness (also called density in 3-SAT, and connectivity in 3-colorability) of the instance. The constrainedness is defined as the ratio of clauses for 3-SAT (or edges for 3-colorability) over variables (resp. vertices). In random 1-3-HornSAT, there are three parameters: the order of the instance and the two densities, namely d_1 and d_3. By taking a cut along the three dimensional surface shown in Figure 4.4 (left), we can study the problem as if it had only two input parameters.

We took two straight line cuts of the surface. For the first cut, we fixed d_1 to be 0.1, we let d_3 take values in the range $[1, 5.5]$ with step 0.1, and we chose order values 500, 1000, 2500, 5000, 10000, 20000 and 40000. See Figure 4.5(left), where we plot the probability of satisfiability along this cut. This plot reveals a quick change on the probability of satisfiability as the input parameter d_3 passes through a critical value (around 3). One technique that has been used to support experimental evidence of a phase transition is finite-size scaling. It is a technique coming from statistical mechanics that has been used in studying the phase transitions of several NP-complete problems, as k-SAT and AC-matching [56, 57]. This technique uses data from finite size instances to extrapolate to infinite size instances. The transformation is based on a rescaling according to a power law of the form $d' = \frac{d-d_c}{d_c} n^r$, where d is

the density, d' is the rescaled parameter, d_c is the critical value, n is the order of the instance and r is a scaling exponent. As a result, a function $f(d, n)$ is transformed to a function $f(d')$. We applied finite-size scaling to our data to observe the sharpness of the transition. We followed the procedure presented by Kolaitis et al. in [57]. Our analysis yields the following finite-size scaling transformation:

$$d' = \frac{d_3 - 3.0385}{3.0385} n^{0.4859}$$

We then superimposed the curves shown in Figure 4.5(left) rescaled according to this transformation. The result is shown in Figure ref01facts(right). The fit appears to be very good around zero, where curves collapse to a single universal curve, but as we move away from it is getting weaker. In the plot, the universal curve seems to be monotonic with limits $lim_{d'->-\infty} f(d') = 1$ and $lim_{d'->\infty} f(d') = 0$. This evidence suggests that there is a phase transition near $d_3 = 3$ for $d_1 = 0.1$.

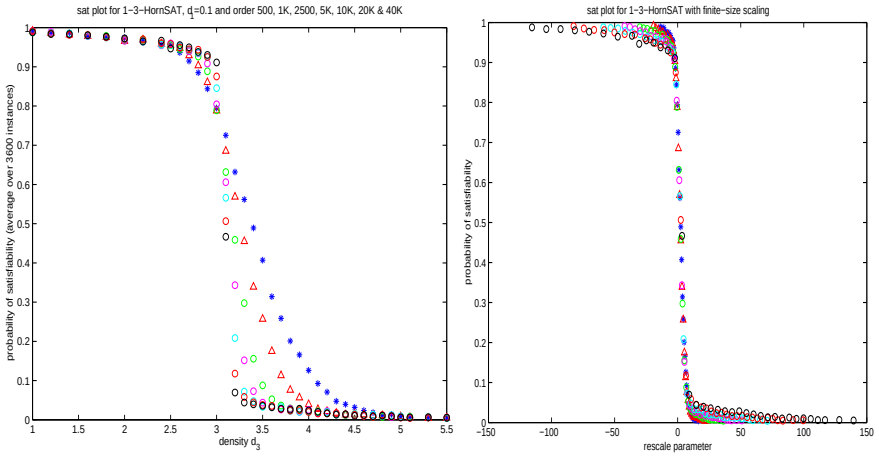

Figure 4.5 Average satisfiability plot of a random 1-3-Horn formula along the $d_1 = 0.1$ cut (left) and the satisfiability plot with rescaled parameter using finite-size scaling (right).

We repeated the same experiment and analysis with the second cut, a straight line cut along the diagonal of the $d_1 \times d_3$ quadrant. In this case our formal parameter is an integer i. An instance with input paremeter value i, corresponds to an instance with densities $d_1 = \frac{i}{200}$ and $d_3 = \frac{i}{10} + 1$. In this case, by making the two input parameters d_1 and d_3 dependent, we effectively reduce the input parameters of the problem from three, (d_1, d_3, n), to two, (i, n). We let i take values in the range $[1, 40]$ with step 1, and we chose order values 500, 1000, 2500, 5000, 10000, 20000 and 40000. See Figure 4.6(left) where we plot the probability of satisfiability along this cut. This plot, as the one for the previous cut, reveals a quick change on the probability of satisfiability as the input parameter i passes through a critical value (around 19). We again used finite-size scaling on these data, looking for further support of a phase transition. For this cut, the analysis yields the following transformation:

$$i' = \frac{i - 19.1901}{19.1901} n^{0.2889}$$

See Figure 4.6(right) where we superimpose the curves shown in the same figure (left) using the above transformation. As with the previous cut, the fit seems quite good, especially around zero, and the universal curve seems to have limits 1 and 0 in the infinities.

In our search for more evidence of a phase transition, we performed the following experiment for the cut used to produce the data in Figure 4.5 ($d_1 = 0.1$). For several values of order between 500 and 200000 and for densitiy d_3 taking values in the range

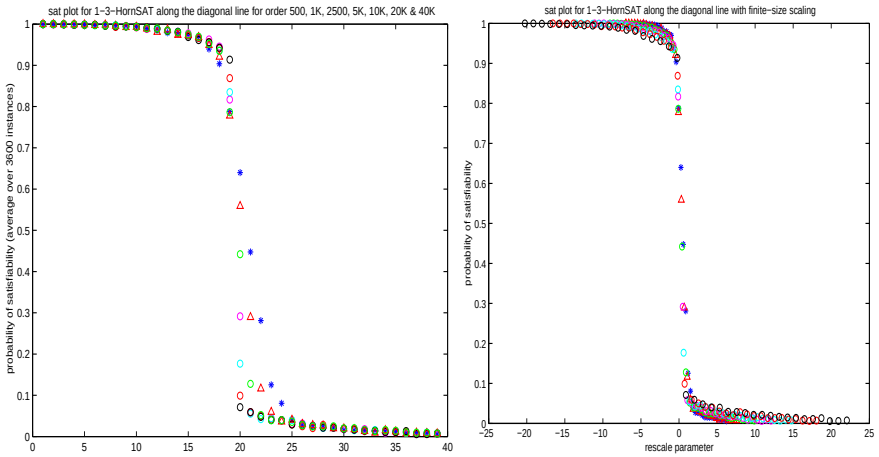

Figure 4.6 Average satisfiability plot of a random 1-3-Horn formula along the diagonal cut (left) and the satisfiability plot with rescaled parameter using finite-size scaling (right).

$[2.7, 3.8]$ with step 0.02, we generated and solved 1200 instances. We recorded for each different order value the values of density d_3 for which the average probability of satisfiability was 0.1, 0.2, 0.8 and 0.9 respectively*. The idea behind this experiment is that if the problem has a sharp threshold, i.e. a phase transition, then as the order of the instances increases the window between 10th and 90th probability percentiles, as well that between 20th and 80th probability percentiles should shrink and at the limit become zero. In Figure 4.7 we plot these windows. Indeed, they get smaller as the order increases.

Although Figure 4.7 shows that these windows indeed shrink as the order increases, it is not clear at all if at the limit they would go to zero. A further curve fitting analysis is more revealing. See Figure 4.8 where we plot the size of the 10%-90% probability

*We actually did linear regression on the two closest points to compute the density for each satisfiability percentage

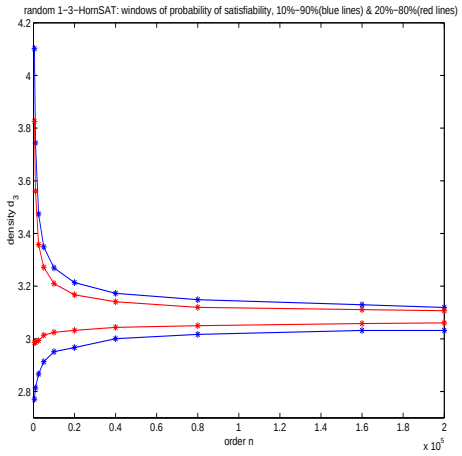

Figure 4.7 Windows of probability of satisfiability of random 1-3-Horn formulae along the $d_1 = 0.1$ cut

of satisfiability window (left) and the 20%-80% probability of

satisfiability window (right) as a function of the order . Using MATLAB to do curve fitting on our data, we find that both windows decrease almost as fast as $\frac{1}{\sqrt{n}}$. The correlation coefficient r^2 is almost 0.999, which gives a high confidence for the validity of the fit. This analysis, suggests that indeed the two windows should be zero at the limit. That is an evidence that supports the existence of a phase transition for 1-3-HornSAT.

Similar analysis has been done before for the k-SAT. The *width* of the satisfiability phase transition, which is the amount by which the number of clauses of a random instance needs to be increased so that the probability of satisfiability drops from $1-\epsilon$ to ϵ, is thought to grow as $\Theta(n^{1-\frac{1}{\nu}})$. Notice that the window that we estimate is equal to the normalized width (divided by the order). The exponent ν for $2 \leq k \leq 6$ is

estimated in [55, 56, 64, 65]. It was also conjectured that as k gets large, ν tends to 1. Recently, Wilson in [79] proved that for all $k \geq 3$, $\nu \geq 2$, therefore the transition width is at least $\Theta(n^{\frac{1}{2}})$. Our experiments suggest that the window of the satisfiability transition for 1-3-HornSAT shrinks as fast as $n^{-\frac{1}{2}}$, thus the transition width grows as $n^{\frac{1}{2}}$. We believe that the analysis in [79] can be applicable in the case of the 1-3-HornSAT, and can complement our experimental findings.

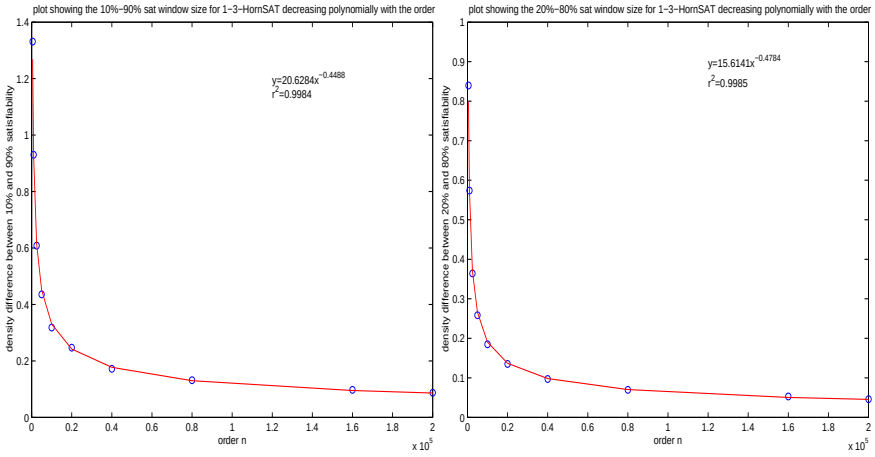

Figure 4.8 Plot of the 10%-90% probability of satisfiability window as a function of the order n (left) and of the 20%-80% probability of satisfiability window (right)

In the rest of this section we will discuss the connection between random Horn formulae and random hypergraphs. We will show how recent results on random hypergraphs, provide a good fit for our experimental data on random 1-3-HornSAT presented so far. On the other hand, these results suggest that the transition is steep, but not a step function.

There is a one to one correspondence between random Horn formulae and random

directed hypergraphs. Let ϕ be a $H^{1,3}_{n,d_1,d_3}$ random formula. We can represent ϕ with the following hypergraph G_ϕ*:

- represent each variable x_i in ϕ with a node v_i in G_ϕ
- represent each unit clause $\{x_k\}$ as a hyperedge in G_ϕ over v_k *
- represent each clause $\{x_j, \bar{x_k}, \bar{x_l}\}$ as a directed hyperedge in G_ϕ over the set $\{v_j, v_k, v_l\}$

In some recent development, Darling and Norris [29] proved some results on the vertex identifiability in random undirected hypergraphs. A vertex v of a hypergraph is *identifiable in one step* if there is a hyperedge over v. A vertex v is *identifiable in* n steps if there is a hyperedge over a set S, such that $v \in S$ and all other elements of S are identifiable in less than n steps. Finally, a vertex v is *identifiable* if it is identifiable in n steps for some positive n.

We now establish the equivalence between the satisfiability of ϕ and the *identifiability* of vertex v_k of G_ϕ, where $c = \{\overline{x}_k\}$ is the unique single negative literal clause of ϕ. First, we introduce an algorithm for solving Horn satsifiability.

We use a simple algorithm for deciding wether a Horn formula is satisfiable or not, presented by Dowling and Gallier in [32] (see also [5]). This algorithm runs in time $O(n^2)$ where n is the number of variables in the formula. Dowling and Gallier in their

*This representation actually omitts the single negative literal that appears in ϕ.
*Hyperedges over vertices are called *patches* in [29] or *loops* in [34].

work actually describe how to improve this algorithm to run in linear time. For our purposes and for the sake of simplicity we will be using the simple quadratic algorithm.

Algorithm A.

begin

let $\phi = \{c_1, \cdots, c_m\}$

consistent:=**true**; change:=**true**;

set each variable x_i to be **false**;

for each variable x_i such that $\{x_i\}$ is a clause in ϕ

set x_i to **true**

endfor;

while (change **and** consistent) **do**

change:=**false**;

for each clause c_j in ϕ do

if (c_j is of the form $(\bar{x_1}, \cdots, \bar{x_q})$

and all $x_1, \cdots, x_q$ are set to **true**) **then**

consistent:=**false**;

else

if c_j is of the form $\{x_1, \bar{x_2}, \cdots, \bar{x_q}\}$

and all $x_2, \cdots, x_q$ are set to **true**

and x_1 is set to **false**

then set x_1 to **true**; change:=**true**; $\phi := \phi - c_j$

endif

endif

endfor

endwhile

end

If algorithm A terminates with consistent:=**true** then a satisfying truth assignement has been found. Otherwise, the formula ϕ is unsatisfiable.

Given a formula ϕ, its corresponding directed hypergraph G_ϕ, and a variable x_i, we will prove the following relation between the truth value that algorithm A assigns to x_i and the identifiability of vertex v_i of G_ϕ:

Lemma 1 *Algorithm A running on ϕ assigns the value* true *to x_i if and only if the vertex v_i of G_ϕ is identifiable.*

Proof. It is easy to show the equivalence by induction on the number of steps required to identify v_k (equivalently the number of iterations of the while loop of algorithm A needed to set the value of x_k to **true**).

Basic Step: If v_k is identifiable in one step, then $\{x_k\}$ is a clause in ϕ and algorithm A will immediately assign the value **true** to it, and vice versa.

Inductive Hypothesis: A vertex is identifiable in $n-1$ steps if and only if the corresponding variable is set to **true** by algorithm A in no more than $n-1$ iterations of the while loop. *Inductive Step*: A vertex v_j that is identifiable in n steps, corresponds to a variable that appears in a clause of the form $\{v_j, \bar{v_{i_1}}, \cdots, \bar{v_{i_q}}\}$ and since all of $x_{i_1}, \cdots, x_{i_q}$ are already set to true, A will set x_j to **true** in the nth iteration of the while loop. Conversely, if x_j is set to **true** in the nth iteraton of the while loop of algorithm A, then we derive that it appears in a clause of the form $\{x_j, \bar{x_{i_1}}, \cdots, \bar{x_{i_q}}\}$, where all of $x_{i_1}, \cdots, x_{i_q}$ are already set to **true**. But this implies that all $v_{i_1}, \cdots, v_{i_q}$ are identifiable in $n-1$ steps; therefore v_j is identifiable in n steps.

□

As an immediate result of this lemma we get:

Corollary 1 *Let ϕ be a $H^{1,3}_{n,d_1,d_3}$ random formula and $c = \{\bar{x_k}\}$ be the unique single negative literal clause of ϕ. Let G_ϕ be the directed hypergraph corresponding to ϕ. The formula ϕ is satisfiable if and only if the vertex v_k of G_ϕ is not identifiable.*

Darling and Norris in [29] studied the vertex identifiability in random undirected hypergraphs. Although, Horn formulae correspond to directed hypergraphs, we decided to use the results of Darling and Norris in an effort to approximate the satisfiability of Horn formulae. The authors use the notion of a *Poisson random hypergraph*. A Poisson random hypergraph on a set V of n vertices with non-negative parameters $\{\beta_k\}_{k=0}^{\infty}$ is a random hypergraph Λ such that, if $\Lambda(A)$ is hyperedges of Λ over the set

of vertices $A \in V$, then $\{\Lambda(A)\}_{A\in 2^V}$ are independent Poisson random variables with $E\Lambda(A) = n\beta_{|A|}/\binom{n}{|A|}$. Equivalently, the distribution of $\Sigma_{|A|=k}\Lambda(A)$, the total number of $k-$hyperedges, is Poisson*$(n\beta_k)$, and they are distributed uniformly at random among all $\binom{n}{k}$ possible $k-$ sets.

One of the key results they proved is the following:

Theorem 3 *[Darling-Norris] Let $\beta = (\beta_j : j \in N)$ be a sequence of non-negative parameters. Let $\beta(t) = \Sigma_{j\geq 0}\beta_j t^j$ and $\beta'(t)$ the derivative of $\beta(t)$. Let $z^* = inf\{t \in [0,1) : \beta'(t) + \log(1-t) < 0\}$; if the infimum is not well-defined then let $z^* = 1$. Denote by ζ the number of zeros of $\beta'(t) + \log(1-t)$ in $[0, z^*)$.*

Assume that $z^* < 1$ and $\zeta = 0$. For $n \in N$, let V^n be a set of n vertices and let G^n be a Poisson(β) hypergraph on V^n. Then, as $n \to \infty$ the number of identifiable vertices V^{n*} satisfies the following limit w.h.p.: $V^{n*}/n \to z^*$.

If we ignore the direction* of the hyperedges then the random hypergraph G_ϕ representing a $H^{1,3}_{n,d_1,d_3}$ random formula corresponds to a Poisson(β) hypergraph G^n. To see that, notice that the hyperedges in G_ϕ are distributed uniformly at random among all possible 1- and 3-sets of vertices, just like in a Poisson random hypergraph

*The Poisson Distribution is a discrete distribution which takes on the values $X = 0, 1, 2, 3, \cdots$. The distribution is determined by a single parameter λ. The distribution function of the Poisson is $f(x) = \frac{\exp(-\lambda)\lambda^x}{x!}$

*Ignoring the direction of the hyperedges is equivalent to adding to the formula for each clause $(x \vee \bar{y} \vee \bar{z})$ two more clauses: $(\bar{x} \vee y \vee \bar{z})$ and $\bar{x} \vee \bar{y} \vee z$. Therefore we expect that the probability of satisfiability we get from the hypergraph model should be lower than the actual probability as it is measured by our experiments. This is indeed the case as we can see in Figure 4.10.

with only two non-zero parameters, β_1 and β_3. To find the values of these parameters, we set equal the probabilities that a hyperedge exists in the two graphs G_ϕ and G^n: $d_1 = 1 - (1 - \frac{1}{\binom{n}{1}})^{n\beta_1} = 1 - e^{-\beta_1}$, $\frac{d_3 n}{\binom{n}{3}} = 6d_3/(n-1)(n-2) = 1 - (1 - \frac{1}{\binom{n}{3}})^{n\beta_3} = 1 - e^{6\beta_3/(n-1)(n-2)}$. As $n \to \infty$, $\beta_1 = -\log(1 - d_3)$ and $\beta_3 = d_3$.

We use MATLAB (`www.mathworks.com`) to compute the z^* for the hypergraph G^n on the quadrant $d_1 \times d_3$. From Corollary 1, we get that the probability of satisfiability of ϕ is 1 minus the probability that v_k is identifiable in G^n, and that is $1 - z^*$. See Figure 4.9(left) where we plot the probability of satisfiability of ϕ against the input parameters d_1 and d_3. A contour plot of the probability of satisfiability is given in Figure 4.9 (right).

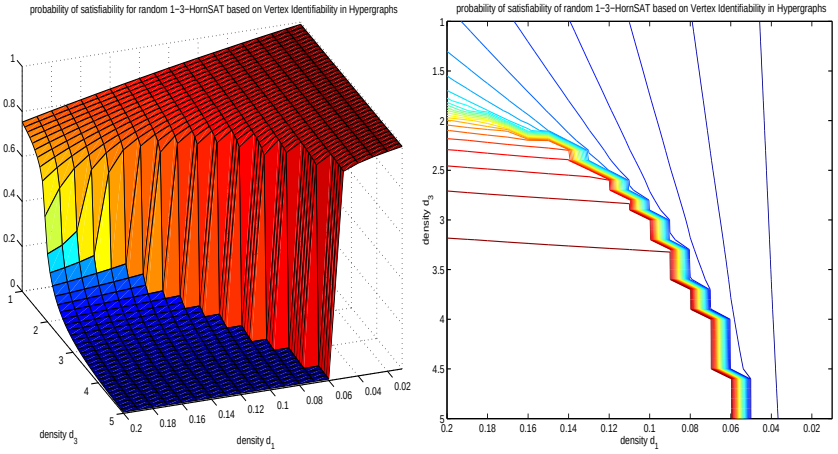

Figure 4.9 Probability of satisfiability plot of a random 1-3-Horn formula according to the vertex-identifiability model(left) and the corresponding contour plot (right).

Comparing the results derived by this model (Figure 4.9) and the results obtained by our experiments (Figure 4.4), we see that the model derived by the hypergraph

analysis provides a very good fit of the experimental data. This is also obvious in Figure 4.10 where we plot the 50% satisfiability line according the model above (the rough curve) and according to our experimental data (smoother curve).

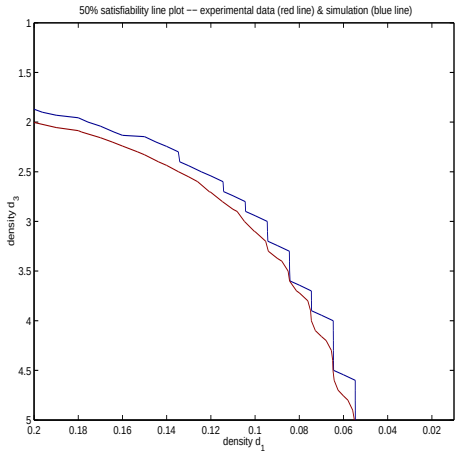

Figure 4.10 50% satisfiability line – According to the model derived through hypergraphs (line with jumps) and according to our experimental data (smoother line).

Finally, we used our model to estimate the probability of satisfiability along the same two cuts that we presented earlier (the $d_1 = 0.1$ and the diagonal cut). See Figure 4.11 for the probability estimation along the two cuts according to the hypergraph-based model, and compare with our experimental fidings shown in Figure 4.5 (left) and Figure 4.6 (left). For both cuts, the estimated probability has a steep drop that is happening at the exact same point that respective drop is observed on the experimental data. In Table 4.1 we give the raw data that correspond to the plots in Figure 4.11. Notice that, despite the very quick transition, the estimated curve is not a step

function, as we would expect by looking our data and the limit curve after the finite-size scaling analysis (Figures 4.5 and 4.6 (right)). Should this be an accurate model for the 1-3-HornSAT, the probability of satisfiability will not be a step function at the limit*.

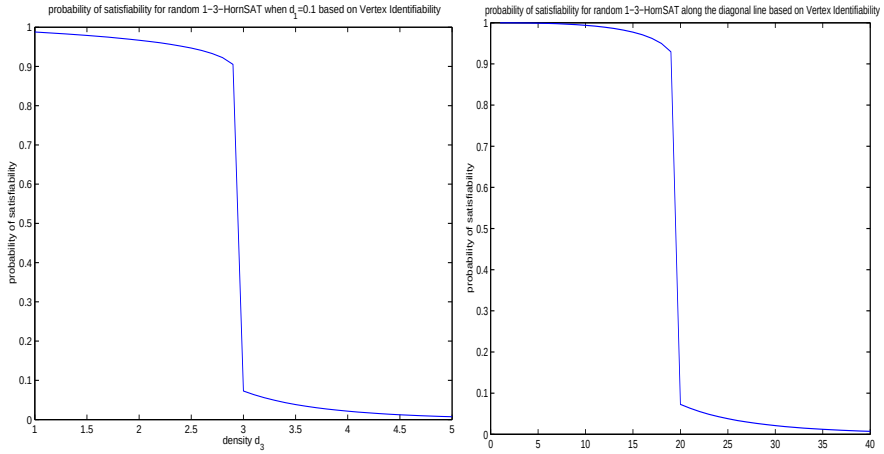

Figure 4.11 Probability of satisfiability plot of a random 1-3-Horn formula according to the vertex-identifiability model, along the $d_1 = 0.1$ cut (left) and the diagonal cut (right).

*This can also be the case for 3-SAT.

$d_1 = 0.1$ cut		diagonal cut	
d_3	prob. of sat.	input parameter i	prob. of sat.
1	0.98775	1	0.99997
1.1	0.98619	2	0.99988
1.2	0.98455	3	0.9997
1.3	0.98282	4	0.99941
1.4	0.98098	5	0.99899
1.5	0.97903	6	0.99841
1.6	0.97694	7	0.99764
1.7	0.9747	8	0.99664
1.8	0.9723	9	0.99537
1.9	0.96969	10	0.99376
2	0.96685	11	0.99175
2.1	0.96372	12	0.98924
2.2	0.96026	13	0.98611
2.3	0.95637	14	0.98217
2.4	0.95194	15	0.97717
2.5	0.94679	16	0.97069
2.6	0.94062	17	0.96202
2.7	0.9329	18	0.94968
2.8	0.92244	19	0.9294
2.9	0.90522	20	0.072832
3	0.072832	21	0.063411
3.1	0.063588	22	0.055476
3.2	0.055745	23	0.048727
3.3	0.049039	24	0.042943
3.4	0.043267	25	0.038
3.5	0.038272	26	0.0335
3.6	0.033928	27	0.029856
3.7	0.030137	28	0.026559
3.8	0.026815	29	0.023665
3.9	0.023896	30	0.021117
4	0.021324	31	0.018868
4.1	0.019052	32	0.016878
4.2	0.017041	33	0.015114
4.3	0.015257	34	0.013547
4.4	0.013672	35	0.012153
4.5	0.012262	36	0.01091
4.6	0.011006	37	0.0098016
4.7	0.0098849	38	0.0088112
4.8	0.0088836	39	0.0079255
4.9	0.0079881	40	0.0071324

Table 4.1 Data for the prob. of satisfiability of random 1-3-Horn formula according to the vertex-identifiability model, along the $d_1 = 0.1$ and the diagonal cut.

Chapter 5
Conclusions

The research work presented in this thesis can be summarized as follows: we studied the 3-SAT problem and we observed a polynomial to exponential complexity phase transition that is located to the left of the satisfiability threshold; we made similar observations for the case of the 3-Colorability problem, showing a robustness of this phase transition phenomenon among different combinatorial problems; finally, in an effort to avoid the limitations imposed to us by the intractability of the previous two problems, we studied the 1-3-HornSAT problem, showing that even for a tractable problem observing and analyzing a phase transition can be really hard. In the following paragraphs, we discuss the results of our work and draw conclusions.

In the case of the random 3-SAT we provide experimental evidence for the following hypotheses. First, the connection between the phase transition in computational complexity and the phase transition in satisfiability is not as tight as has been claimed. It is not the case that the shift from polynomial to exponential complexity occurs at or very close to the crossover point, as has been widely believed [67, 68]. Second, not only does the density at which the shift from polynomial to exponential time complexity vary with the choice of solver, but the very shape of the surface of the median running time (an experimental surrogate for average-time complexity), as a function

of the density d and the order n, changes with the solver. Finally, the density-order quadrant contains several phase transitions; in fact, the region between density 0 and density 4.26 seems to be rife with phase transitions, which are also solver dependent. In essence, each solver provides us with a different tool with which to study the complexity of random 3-SAT. This is analogous to astronomers observing the sky using telescopes that operate at different wave lengths. We thus hope to alleviate the "fixation" with DLL solvers and the crossover point at 4.26.

Our experiments reveal a marked difference between solvers like GRASP and CPLEX, which are search based and display interesting similarities in the shapes of the median running time surface despite their different underlying algorithmic techniques, and ROBDD-based solvers, like CUDD, which are based on compactly representing all satisfying truth assignments. While the interesting region for GRASP and CPLEX is between 2.5 and 4.3, the interesting region for CUDD occurs below density 2. This refutes earlier conjectures (cf. [58]) that the peak in median running time around the crossover point is essentially solver independent. For both GRASP and CPLEX, we observed a new phase transition where the median running time shifts from being polynomial in the order to being exponential in the order. For GRASP the transition is happening at around density 3.8, while for CPLEX the transition is happening earlier, near density 3.0. ¿From the perspective of average-time complexity this is a significant phase transition because it corresponds to a qualitative shift in the

behavior of the solver. We also observed several other phase transitions for CPLEX and for CUDD. This suggests that it would be interesting to explore the behavior of other SAT solvers, such as RELSAT [51] or SATZ [59], on the $d \times n$ quadrant.

With fine grained sampling of the density parameter, and by exploring a greater range in the number of variables, we can start to document for each solver, phase transitions that correspond to significant shifts in the shape of the running time of the solver. These phase transitions are important to our understanding of the computational complexity of random 3-SAT, and can be used as a justification to develop density-based solvers for 3-SAT, i.e., solvers which use information about the density of an instance, to choose the most appropriate algorithmic technique.

While our results are purely empirical, as the lack of success with formally proving a sharp complexity threshold at the crossover point indicates (cf. [38, 33, 1]), providing rigorous proof for our qualitative observations may be a very difficult task, especially for sophisticated solvers like the ones studied in this paper.

We also studied the average-case complexity of the 3-Colorability of random graphs. Our goal was to find how the complexity of the problem scales as a function of the order of the instance, for instances of fixed connectivity, and see if a similar behavior to that of 3-SAT can be observed for 3-Colorability. A polynomial complexity to exponential complexity phase transition was first conjectured in [47] and later counter-conjectured in [26].

Our experimental findings provide evidence that 3-colorability is a problem, like 3-SAT, that has a double phase transition, as conjectured in [47]. Using SMALLK to solve the random instances, we find that the running time shifts from polynomial in the order to exponential in the order before the threshold and around connectivity 4.3. In the region where this transition is happening, we observe a heavy tail phenomenon; a significant number of instances require much more time than the median running time. Also the mean and median running time, that are in almost equal along the connectivity range, in that narrow region differ significantly, with the mean been much larger than the median.

The complexity phase transition and the phenomena that are observed in the same region, could be signaling some development in the structure of the problem, and require further study. A number of colorability solvers are based on the very effective Brélaz heuristic. A different approach to solve colorability is to use evolutionary algorithms. In [36] it is shown that an adaptive evolutionary algorithm for hard instances of large order outperforms a powerful traditional graph coloring technique from Brélaz. The algorithm is a genetic algorithm that uses exclusively mutation and has population size 1. In [3], two different formulations of the colorability problem are introduced; one that is based on the connection between the chromatic number of a graph and its acyclic orientations and another one that aims to develop programs that color all graphs that belong to a same class according to their order and other common

attributes. The heuristics developed in this study are tested on some colorability benchmarks and shown to perform very well. It would be interesting to study how these evolutionary algorithms scale with the order, and how does the average running time surface look like over the $\gamma \times n$ quadrant.

Finally, we set out to investigate the existence of a phase transition on the satisfiability of the random 1-3-HornSAT problem. This is a problem that is similar to 3-SAT, but its polynomial complexity allows us to collect data for much higher order, unlike the other two problems we studied (3-SAT and 3-Colorability) were the exponential complexity is limiting the order for which we can experimentally study these problems, especially around the phase transition.

We first showed, through our experimental findings and an analysis based on known results from digraphs' reachability, that the 1-2-HornSAT is a problem that lacks a phase transition.

On the contrary, our experiments provide evidences that the 1-3-HornSAT has a phase transition. By thoroughly sampling the $d_1 \times d_3$ quadrant, solving a large number of random instances of large order, we document a waterfall-like probability of satisfiability surface. In addition, by taking cuts of this surface, we are able to observe a quick transition from a satisfiable to an unsatisfiable region. When finite-size scaling is applied on these cuts, it suggests that there is a phase transition. Finally, analysis of the transition window provide further evidence for the phase transition.

We then used some recent results on random hypergraphs to generate a model for our experimental data. By comparing the waterfall-like probability surface against the estimated probability according to this model, we see that the hypergraph-based model fits well our experimental data. This suggests that further analysis based on hypergraphs could provide a rigorous analysis of the conjectured phase transition for the 1-3-HornSAT. This would be very significant since there are very few phase transitions that have been analytically proved (like that of 2-SAT [17, 31, 43]). Although this model fits well our experimental data, when calculating the estimated probability along the two cuts, we see that the probability of satisfiability as the order goes to infinity is a very steep function, but not a step function. This last finding, that suggests the opposite than the experimental findings, shows the difficulty of experimentally showing a phase transition, even in tractable problem like 1-3-HornSAT.

References

1. D. Achlioptas. Setting two variables at a time yields a new lower bound for random 3-SAT. In *Proc. 32th ACM Symp. on Theory of Computing*, pages 28–37, 2000.

2. D. Achlioptas, P. Bemame, and M. Molloy. A sharp threshold in proof complexity. In *Proc. 33rd ACM Symp. on Theory of Computing*, pages 337–346, 2001.

3. V. C. Barbosa, C. A. G. Assis, and J. O. do Nascimento. Two novel evolutionary formulations of the graph coloring problem. *Journal of Combinatorial Optimization*, to appear.

4. P. Beame, R. M. Karp, T. Pitassi, and M. E. Saks. On the complexity of unsatisfiability proofs for random k-CNF formulas. In *Proc. 30th ACM Symp. on Theory of Computing*, pages 561–571, 1998.

5. C. Beeri and P. A. Bernstein. Computational problems related to the design of normal form relational schemas. *ACM Trans. on Database Systems*, 4:30–59, 1979.

6. A. Biere, A. Cimatti, E. M. Clarke, M. Fujita, and Y. Zhu. Symbolic model checking using SAT procedures instead of BDDs. In *Proc. 36th Conf. on Design Automation*, pages 317–320, 1999.

7. R. E. Bixby. Implementing the Simplex method: The initial basis. *ORSA J. on Computing*, 4(3):267–284, 1992.

8. R. E. Bixby. Progress in linear programming. *ORSA J. on Computing*, 6(1):15–22, 1994.

9. B. Bollobás. *Random Graphs.* Academic Press, London, 1985.

10. F. Bouquet. *Gestion de la dynamicité et énumération d'implicants premiers: une approche fondée sur les Diagrammes de Décision Binaire.* PhD thesis, Université de Provence, France, 1999.

11. D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.

12. A. Z. Broder, A. M. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-CNF formulas. In *Proc. 4th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 322–330, 1993.

13. R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Trans. on Computers*, 35(8):677–691, 1986.

14. J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170, June 1992.

15. M. Chao and J. V. Franco. Probabilistic analysis of a generalization of the unit-clause literal selection heuristics for the k-satisfiability problem. *Information Sciences*, 51:289–314, 1990.

16. P. Cheeseman, B. Kanefsky, and W. M. Taylor. Where the really hard problems are. In *Proc. 12th Int'l Joint Conf. on Artificial Intelligence (IJCAI '91)*, pages 331–337, 1991.

17. V. Chvátal and B. Reed. Mick gets some (the odds are on his side). In *Proc. 33rd IEEE Symp. on Foundations of Computer Science*, IEEE Comput. Soc. Press, pages 620–627, 1992.

18. V. Chvátal and E. Szemerédi. Many hard examples for resolution. *J. of the ACM*, 35(4):759–768, 1988.

19. C. Coarfa, D.D. Demopoulos, A. San Miguel Aguirre, D. Subramanian, and M.Y. Vardi. Random 3-sat – the plot thickens. In R. Dechter, editor, *Proc. Principles and Practice of Constraint Programming (CP'2000)*, Lecture Notes in Computer Science 1894, pages 143–159, 2000.

20. S. Cocco and R. Monasson. Statistical physics analysis of the computational complexity of solving random satisfiability problems using backtrack algorithms. *European Physical Journal B*, 22:505–531, 2001.

21. S. A. Cook and D. G. Mitchell. Finding hard instances of the satisfiability problem: A survey. In Du, Gu, and Pardalos, editors, *Satisfiability Problem: Theory and Applications*, volume 35 of *Dimacs Series in Discrete Math. and Theoretical Computer Science*. 1997.

22. J. M. Crawford and L. D. Auton. Experimental results on the crossover point in satisfiability problems. *AAAI*, pages 21–27, 1993.

23. J. M. Crawford and L. D. Auton. Experimental results on the crossover point in random 3-SAT. *Artificial Intelligence*, 81(1-2):31–57, 1996.

24. J. M. Crawford and A. B. Baker. Experimental results on the application of satisfiability algorithms to scheduling problems. *AAAI*, 2:1092–1097, 1994.

25. J. Culberson. Graph Coloring Bibliography. `http://www.cs.ualberta.edu/ joe/Coloring/Bibliography/` .

26. J. Culberson and I. P. Gent. Well out of reach: why hard problems are hard. Technical Report APES-13-1999, APES Research Group.

27. J. Cullberson and I. P. Gent. Frozen development in graph coloring. Technical report, Department of Computer Science, University of St. Andrews, 2000. APES Research Report APES-19-20004.

28. V. Dalmau. Thresholds of combinatorial problems. Manuscript.

29. R. W. R. Darling and J. R. Norris. Vertex identifiability in large random hypergraphs, 2001. arXiv:math.PR/0109020.

30. M. Davis, G. Longemann, and D. Loveland. A machine program for theorem proving. *Communications of the ACM*, 5:394–397, 1962.

31. W. Fernandez de la Vega. On random 2-sat, 1992. Manuscript.

32. W. F. Dowling and J. H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *Logic Programming. (USA) ISSN: 0743-1066*, 1(3):267–284, 1984.

33. O. Dubois, Y. Boufkhad, and J. Mandler. Typical random 3-SAT formulae and the satisfiability theshold. In *Proc. 11th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 126–127, 2000.

34. P. Duchet. Hypergraphs. In R. Graham, M. Grötschel, and L. Lovász, editors, *Handbook of Combinatorics*. Elsevier Science, 1995.

35. P. E. Dunne and M. Zito. An improved upper bound on the non-3-colourability threshold. *Information Processing Letters*, 65(1):17–23, 1998.

36. A. E. Eiben, J. K. van der Hauw, and J. I. van Hemert. Graph coloring with adaptive evolutionary algorithms. *Journal of Heuristics*, 4(1):25–46, 1998.

37. P. Erdös and A. Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Science*, 5:17–61, 1960.

38. E. Friedgut. Necessary and sufficient conditions for sharp threshold of graph properties and the k-SAT problem. *J. Amer. Math. Soc.*, 12:1917–1054, 1999.

39. A. Frieze and S. Suen. Analysis of two simple heuristics for random instances of k-SAT. *J. of Algorithms*, 20(2):312–355, 1996.

40. M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979.

41. I. P. Gent and T. Walsh. Easy problems are sometimes hard. *Artificial Intelligence*, 70(1-2):335–345, 1994.

42. I.P. Gent and T. Walsh. The SAT phase transition. In *Proc. European Conf. on Artificial Intelligence*, pages 105–109, 1994.

43. A. Goerdt. A threshold for unsatisfiability. *J. Comput. System Sci.*, 53(3):469–486, 1996.

44. J. F. Groote. Hiding propositional constants in BDDs. *Formal Methods in System Design*, 8:91–96, 1996.

45. J.F. Groote and H. Zantema. Resolution and binary decision diagrams cannot simulate each other polynomially. Technical report, Department of Computer Science, Utrecht University, 2000. Technical Report UU-CS-2000-14.

46. L. Henschen and L. Wos. Unit refutations and Horn sets. *J. of the ACM*, 21(4):590–605, 1974.

47. T. Hogg and C. P. Williams. The hardest constraint problems: A double phase transition. *Artificial Intelligence*, 69(1-2):359–377, 1994.

48. J. N. Hooker. Resolution vs. cutting plane solution of inference problems: Some computational experience. *Operations Research Letters*, 7:1–7, 1988.

49. G. Istrate. The phase transition in random Horn satisfiability and its algorithmic implications. In *Proc. 10th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 925–926, 1999.

50. S. Jha, Y. Lu, M. Minea, and E. M. Clarke. Equivalence checking using abstract BDDs. In *Proc. 1997 Int'l Conf. on Computer Design*, pages 332–337, 1997.

51. Roberto J. Bayardo Jr. and Robert Schrag. Using CSP look-back techniques to solve real-world SAT instances. In *AAAI/IAAI*, pages 203–208, 1997.

52. R. M. Karp. The transitive closure of a random digraph. *Random Structures and Algorithms*, 1(1):73–93, 1990.

53. H. Kautz and B. Selman. Planning as satisfiability. In *Proc. European Conference on Artificial Intelligence*, pages 359–379, 1992.

54. R. E. Kirk. *Statistics: An introduction (4th ed.)*. Harcourt Brace, Fort Worth, 1999.

55. S. Kirkpatrick, G. Györgi, N. Tishby, and L. Troyansky. The statistical mechanics of k-satisfaction. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 439–446. Morgan Kaufmann Publishers, 1993.

56. S. Kirkpatrick and B. Selman. Critical behavior in the satisfiability of random formulas. *Science*, 264:1297–1301, 1994.

57. P. Kolaitis and T. Raffill. In search of a phase transition in the AC-matching problem. In *7th Principles and Practice of Constraint Programming (CP2001)*, pages 433–450, 2001.

58. T. Larrabee and Y. Tsuji. Evidence for a satisfiability threshold for random 3CNF formulas. Technical report, University of California, Santa Cruz, 1992. Technical report USCS-CRL-92042.

59. Chu Min Li and Anbulagan. Heuristics based on unit propagation for satisfiability problems. In *IJCAI (1)*, pages 366–371, 1997.

60. J. A. Makowsky. Why Horn formulas matter in Computer Science: Initial structures and generic examples. *JCSS*, 34(2-3):266–292, 1987.

61. J. P. Marques Silva and K. A. Sakallah. GRASP–A search algorithm for propositional satisfiability. *IEEE Trans. on Computers*, 48(5):506–521, 1999.

62. D. G. Mitchell and H. J. Levesque. Some pitfalls for experimenters with random SAT. *Artificial Intelligence*, 81(1-2):111–125, 1996.

63. M. Mitzenmacher. Tight thresholds for the pure literal rule. Technical report, Digital System Research Center, Palo Alto, California, 1997. SRC Technical Note 1997 - 011.

64. R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. $2 + p$-SAT: Relation of typical-case complexity to the nature of the phase transition. *Random Structures & Algorithms*, 15(3-4):414–435, 1999.

65. R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Determining computational complexity from characteristic 'phase transitions'. *Nature*, 400:133–137, 1999.

66. G.J. Nam, K.A. Sakallah, and R. A. Rutenbar. Satisfiability-based layout revisited: Detailed routing of complex FPGAs via search-based boolean sat. In *Proc. ACM Int'l Symp. on Field-Programmable Gate Arrays (FPGA'99)*, pages 167–175, 1999.

67. B. Selman and S. Kirkpatrick. Critical behavior in the computational cost of satisfiability testing. *Artificial Intelligence*, 81(1-2):273–295, 1996.

68. B. Selman, D. G. Mitchell, and H. J. Levesque. Generating hard satisfiability problems. *Artificial Intelligence*, 81(1-2):17–29, 1996.

69. B.M. Smith and M.E. Dyer. Locating the phase transition in binary constraint satisfaction problems. *Artificial Intelligence Journal*, 8(1–2):155–181, 1996.

70. F. Somenzi. CUDD: CU Decision Diagram package. release 2.3.0., 1998. Dept. of Electrical and Computer Engineering. University of Colorado at Boulder.

71. D. Stauffer and A. Aharony. *Introduction to percolation theory.* Taylor and Francis, London, 1994.

72. P. Svenson and M.G. Nordahl. Relaxation in graph coloring and satisfiability problems. *Phys. Rev. E*, 59(4):3983–3999, 1999.

73. The VIS Group. VIS: A system for verification and synthesis. In *Proc. 8th Int'l Conf. on Computer Aided Verification (CAV '96)*, pages 428–432, 1996. LNCS 1102. Ed. by R. Alur and T. Henziger.

74. H. Theil. The measurement of inequality by component of income. *Economics Letter*, 2, 1979.

75. J.S. Turner. Almost all k-colorable graphs are easy to color. *J. of Algorithms*, 9:63–82, 1988.

76. T. E. Uribe and M. E. Stickel. Ordered binary decision diagrams and the Davis-Putnam procedure. In *First Int'l Conf. on Constraints in Computational Logics*, volume 845 of *Lecture Notes in Computer Science*, pages 34–49, Munich, September 1994. Springer-Verlag.

77. M.Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification (preliminary report). In *Proc. 1st IEEE Symp. on Logic in Computer Science*, pages 332–344, 1986.

78. C.P. Williams and T. Hogg. Exploiting the deep structure of constraint problems. *Artificial Intelligence*, 70:73–117, 1994.

79. D. B. Wilson. On the critical exponents of random k-SAT. *Random Structures and Algorithms*. to appear.