Belief propagation algorithms for constraint satisfaction problems

by

Elitza Nikolaeva Maneva

B.S. (California Institute of Technology) 2001

A dissertation submitted in partial satisfaction of the
requirements for the degree of

Doctor of Philosophy
in
Computer Science
and the Designated Emphasis
in
Communication, Computation, and Statistics

in the

GRADUATE DIVISION
of the
UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:
Professor Alistair Sinclair, Chair
Professor Christos Papadimitriou
Professor Elchanan Mossel

Fall 2006

The dissertation of Elitza Nikolaeva Maneva is approved:

Chair	Date
	Date
	Date

University of California, Berkeley

Fall 2006

Belief propagation algorithms for constraint satisfaction problems

Abstract

Belief propagation algorithms for constraint satisfaction problems

by

Elitza Nikolaeva Maneva
Doctor of Philosophy in Computer Science
and the Designated Emphasis in Communication, Computation, and Statistics

University of California, Berkeley

Professor Alistair Sinclair, Chair

We consider applications of belief propagation algorithms to Boolean constraint satisfaction problems (CSPs), such as 3-SAT, when the instances are chosen from a natural distribution—the uniform distribution over formulas with prescribed ratio of the number of clauses to the number of variables. In particular, we show that survey propagation, which is the most effective heuristic for random 3-SAT problems with density of clauses close to the conjectured satisfiability threshold, is in fact a belief propagation algorithm. We define a parameterized distribution on partial assignments, and show that applying belief propagation to this distribution recovers a known family of algorithms ranging from survey propagation to standard belief propagation on the uniform distribution over satisfying assignments. We investigate the resulting lattice structure on partial assignments, and show how the new distributions can be viewed as a "smoothed" version of the uniform distribution over satisfying assignments, which is a first step towards explaining the superior performance of survey propagation over the naive application of belief propagation. Furthermore, we use this lattice structure to obtain a conditional improvement on the upper bound for the satisfiability threshold.

The design of survey propagation is associated with the structure of the solution space of random 3-SAT problems. In order to shed light on the structure of this space for the case of general Boolean CSPs we study it in Schaefer's framework. Schaefer's dichotomy theorem splits Boolean CSPs into polynomial time solvable and NP-complete problems. We show that with respect to some structural properties such as the diameter of the solutions space and the hardness of deciding its connectivity, there are two kinds of Boolean CSPs, but the boundary of the new dichotomy differs significantly from Schaefer's.

Finally, we present an application of a method developed in this thesis to the source-coding problem. We use the dual of good low-density parity check codes. For the compression step we define an appropriate distribution on partial assignments and apply belief propagation to it, using the same technique that was developed to derive survey propagation as a belief propagation algorithm. We give experimental evidence that this method yields performance very close to the rate distortion limit.

Contents

List of Figures

List of Tables

Acknowledgments

I have been extremely lucky to have had the opportunity to work with many outstanding researchers who have also been great mentors for me. I owe this to UC Berkeley, which is magically able to attract the best in everything. I have my adviser Alistair Sinclair to thank for bringing this research topic to my attention, for the support, and for guiding me with a lot of care even when we happened to be far away. I think I benefited a lot from his patience and the depth with which he attacked both research problems and practical issues. I am also thankful for the extremely high quality of the classes he taught. I want to thank Christos Papadimitriou for being a great inspirational figure for me from the moment I arrived at Berkeley. I also owe him a big "thank you" for coming up with a very nice question, the answer to which is now a whole chapter in this thesis. I would like to thank Elchanan Mossel and Martin Wainwright for initiating our collaboration, and thus getting this dissertation rolling. They have both been extremely supportive and endless sources of great advice. I am also grateful to Federico Ardila for getting involved in the combinatorial aspects of this thesis, and teaching me about lattices.

I also want to thank Michael Luby and Amin Shokrollahi for introducing me to the belief propagation algorithm during the course on "Data-Transport Protocols" in Spring 2003. For my understanding of the survey propagation algorithm I owe a lot to Marc Mézardand Andrea Montanari, and the MSRI semester on Probability, Algorithms and Statistical Physics.

My summer internship at IBM in 2005 was also very important for this thesis. I am grateful to Phokion Kolaitis for being at the same time a teacher, collaborator and a mentor to me.

I have been blessed with wonderful fellow graduate students as well. I want to give special thanks to Sam Riesenfeld, Andrej Bogdanov, and Kamalika Chaudhuri. I had fun working on homeworks and projects with them, and I have learned a lot from them. I remember fondly our brainstorming sessions in Brewed Awakening. I also want to thank Parikshit Gopalan for the great job he did during our collaboration at IBM.

Looking much further back, I want to thank the teacher who made me fall in love with math and who taught me the most important things - Rumi Karadjova. Without her talent for teaching, my life would have been completely different. I am also thankful to my two math-school classmates Eddie Nikolova and Adi Karagiozova for working on their PhD degrees in the same area as me in universities as prestigious as UC Berkeley and doing a great job, because they gave me real faith that I was not here purely by accident.

I would like to acknowledge my undergraduate school—the California Institute of Technology —for the vast amount of opportunities they gave me absolutely for free.

I also want to thank Keshav for sharing with me almost the whole journey of becoming a researcher over the last nine years. I learned twice as much by living through both of our experiences at the same time. For the last stretch, which was not trivial, I would like to thank Vikrant and Evimaria for being there for me.

Finally, I want to thank my parents for teaching me to aim high.

Chapter 1

Introduction

A lot of computational problems encountered in science and industry can be cast as *constraint satisfaction problems* (CSPs): a large number of variables have to be assigned values from a given domain so that a large number of simple constraints are satisfied. For example, scheduling the flights at an airport involves assigning a gate to each flight so that flights arriving or departing from the same gate are not less than half an hour apart, unless they use the same plane. A particular problem in the class of constraint satisfaction problems is specified by the domain for the variables and the kind of constraints that can be imposed. An instance of the problem is also called a *formula*.

The case that is the focus of this thesis, is that of variables with Boolean domain $\{0, 1\}$. In 1971 Cook proved that one of these problems, which is known as 3-satisfiability or 3-SAT, is as hard as any problem that can be solved by a non-deterministic Turing machine in polynomial time, thus defining the notion of NP-completeness [Coo71]. The constraints of 3-SAT are disjunctions of 3 variables and/or negations of variables, for example $(x_1 \vee \bar{x}_2 \vee x_3)$ is the constraint that an assignment with $x_1 = 0$, $x_2 = 1$ and $x_3 = 0$ is not satisfying. The application of a particular constraint to a set of variables is also called a *clause*.

In 1978 Schaefer determined the computational complexity of all Boolean constraint satisfaction problems [Sch78]. He showed that all of these problems fall in only two classes: problems that are NP-complete, and problems for which there is a polynomial time algorithm. He also defined simple criteria for checking for a given problem to which of the two classes it belongs.

Both Cook's and Schaefer's work, as well as most of the work on constraint satisfaction problems that followed, focuses on the worst-case complexity of the problems. A more optimistic view is the study of “typical” instances of constraint satisfaction problems. What constitutes a typical instance, of course, depends on the domain of application. The question of modeling what is a typical instance is an interesting and important one, however even in the context of the simplest models that we can think of, we are still at the stage of developing a toolbox for the design and analysis of algorithms for that model. The goal of this thesis is precisely the development of such tools.

The model that we consider here is the following: the total number of clauses is set to αn where n is the number of variables, and α is a positive constant that we will call the *density* (as it can be thought of as the number of clauses per variable). Each clause is generated by choosing a constraint independently and uniformly at random from the set of all possible constraints in the problem and applying it to a random set of variables (of size corresponding to the constraint). For example, in the case of the 3-SAT problem, a clause is generated by choosing 3 random variables, negating each one independently with probability $1/2$, and taking their disjunction.

The first thing we need to understand about a model for generating random instances is what is the probability that a random formula is satisfiable. In the above model it is clear that this probability is non-increasing with respect to the density, since adding more clauses cannot increase the number of satisfying assignments. In the case of 3-SAT, it is conjectured that there is a particular critical density α_c such that for any $\epsilon > 0$ random formulas of density $\alpha_c - \epsilon$ have satisfying assignments with high probability (i.e. probability going to 1 as n goes to infinity), and random formulas of density $\alpha_c + \epsilon$ have no satisfying assignment with high probability. A slightly weaker statement was proved by Friedgut in 1999 [Fri99]. In particular, he showed that there exists a function $\alpha_c(n)$ such that random formulas of density $\alpha_c(n) - \epsilon$ have satisfying assignments with high probability, and random formulas of density $\alpha_c(n) + \epsilon$ have no satisfying assignment with high probability. Some generalizations of this result to other random constraint satisfaction problems have been given by Molloy [Mol03] and by Creignou and Daudé [CD04]. The precise value of the critical density is known only for a few problems: for example for 2-SAT—which is defined the same way as 3-SAT, but each constraint is only on two variables—the critical density is $\alpha_c = 1$ [Goe96, CR92, dlV92] . Generalizing this result to k-SAT for $k \geq 3$ is a major challenge in this area. Achlioptas and Peres showed that as k becomes large $\alpha_c = 2^k \log 2 - O(k)$ [AP03]. For 3-SATthe best known bounds are $3.52 \leq \alpha_c \leq 4.51$ [DBM00, KKL00].

In the last decade, in addition to the theoretical computer science community, these questions have also been tackled by the statistical physics community, albeit by very different methods. One of the main objective of this thesis is to bridge a gap in the methods of the two communities, and build a basis for further cross-fertilization.

In statistical physics constraint satisfaction problems represent a particular example of a spin system. A lot of the phenomena observed in the context of random CSPs are universal in complex physical systems. For example, the transition from a satisfiable regime to an unsatisfiable regime at a critical density α_c is just one example of what is known as a phase transition. More generally, a phase transition is a change in the macroscopic properties of a thermodynamical system when a single parameter of the system is changed by a small amount.

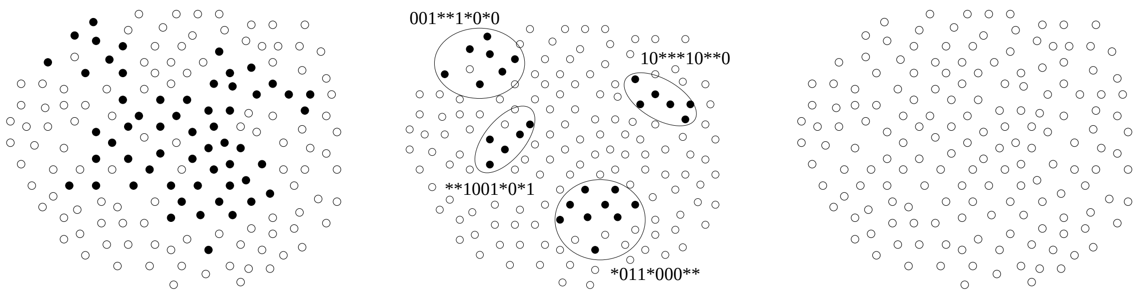
Sophisticated approximation methods that have been developed in the last twenty years—such as the cavity method and the replica ansatz [MPV87, MP03]—provide a general technique for calculating the satisfiability threshold of random constraint satisfaction problems. In particular, the threshold for 3-SAT has been estimated to be $\alpha_c \approx 4.267$ [MPZ02]. Unfortunately, this technique has not yet been proved to yield rigorous results. There is a lot of research effort directed towards turning these estimates into rigorous statements and the confidence in their accuracy among researchers is growing.

The performance of classical algorithms for constraint satisfaction problems such as DPLL [DLL62] and random walks [Pap91] in the context of random instances is also commonly analyzed using statistical physics methods [SM04, CM04]. Their performance appears to be related to physical properties of the system, such as the existence of multiple "states" or "phases", which is claimed to impede random walk algorithms and to cause exponential blow-up in the search tree of DPLL. Such multiple states are claimed to exist for example in the case of random 3-SAT formulas with density close to the satisfiability threshold. There is no mathematical definition of a state in this context. Informally, a system is considered to have a single state when the influence on a particular site (variable or particle) v by other sites diminishes rapidly with their distance from v. Distance here is measured in terms of the graph of interactions, which is known as a factor graph. In the case of a formula this is a bipartite graph with two kinds of vertices—for variables and for clauses—such that there is an edge between a variable node and a clause node if and only if the variable appears in that clause. On the other hand, a system is said to have multiple states when there are long-range correlations between sites that are far apart. A *state* then can be thought of as a subspace of the space of configurations, in which the values of variables that are far away in the factor graph of the formula are uncorrelated.

A very exciting recent algorithmic development has resulted precisely from this view of multiple states, which in physics is known as the "replica symmetry breaking ansatz". The ground-breaking contribution of Mézard, Parisi and Zecchina [MPZ02], as described in an article published in "Science", is the development of a new algorithm for solving k-SAT problems. A particularly dramatic feature of this method, known as *survey propagation* (SP), is that it appears to remain effective at solving very large instances of random k-SAT problems—even with densities very close to the satisfiability threshold, a regime where previously known algorithms typically fail. We will not go into the ideas behind the algorithm in depth here, but refer the reader to the physics literature [MZ02, BMZ03, MPZ02] for details.

In physical systems with multiple states a particular state usually consists of configurations that are similar. In the case of constraint satisfaction problems this general fact has led to the idea that the solutions belonging to a certain state are close in Hamming distance. Therefore, one simple way to think of the transition from a single-state regime to a multiple-state regime is in terms of the geometry of the space of solutions. In particular, the main assumption is the existence of a critical value α_d for the density (for 3-SAT, $\alpha_d \approx 3.92$), smaller than the threshold density α_c, at which the structure of the space of solutions of a random formula changes. For densities below α_d the space of solutions is highly connected—in particular, it is possible to move from one solution to any other by flipping a constant number of variables at a time, and staying at all times in a satisfying assignment. For densities above α_d, the space of solutions breaks up into clusters, so that moving from a satisfying assignment within one cluster to some other assignment within another cluster requires flipping some constant fraction of the variables simultaneously. Informally, one can think of a graph on the satisfying assignments where two assignments are connected if they are constant distance apart. Then below α_d this graph has a single connected component, while above α_d there are (exponentially) many. Since this graph on satisfying assignments is not well-defined we will continue to refer to the components as clusters. Figure 1.1 illustrates how the structure of the space of solutions evolves as the density of a random formula increases. It is important to emphasize that there is no precise connection between the two concepts—the combinatorial concept of a cluster of solutions and the probabilistic concept of a state as a subspace of the configuration space in which there are no long-range correlations.

Within each cluster, a distinction can be made between *frozen* variables—ones that do not change their value within the cluster—and *free* variables that do change their value in the cluster. A concise description of a cluster is an assignment of $\{0, 1, *\}$ to the variables with the frozen variables taking their frozen value, and the free variables taking the joker or wild-card value $*$. The original argument for the clustering assumption was the analysis of simpler satisfiability problems, such as XOR-SAT, where the existence of clusters can be demonstrated by rigorous methods [MRTZ03]. More recently, Mora, Mézard and Zecchina [MMZ05] as well as Achlioptas and Ricci-Tersenghi [ART06] have demonstrated via rigorous methods that for $k \geq 8$ and some clause density below the unsatisfiability threshold, clusters of solutions do indeed exist.



(a) $0 < \alpha < \alpha_d$ (b) $\alpha_d < \alpha < \alpha_c$ (c) $\alpha_c < \alpha$

Figure 1.1. The black dots represent satisfying assignments, and white dots unsatisfying assignments. Distance is to be interpreted as the Hamming distance between assignments. (a) For low densities the space of satisfying assignments is well connected. (b) As the density increases above α_d the space is believed to break up into an exponential number of clusters, each containing an exponential number of assignments. These clusters are separated by a "sea" of unsatisfying assignments. (c) Above α_c all assignments become unsatisfying.

Before we describe the survey propagation algorithm, it is helpful to first understand another algorithm, which is much better known, namely *belief propagation* (BP) [Pea88, YFW03]. Both belief propagation and survey propagation are examples of message-passing algorithms. This is a large class of algorithms with the common trait that messages with statistical information are passed along the edges of a graph of interactions. The goal of belief propagation is to compute the marginal distribution of a single variable in a joint distribution that can be factorized (i.e. a Markov random field). Such a distribution is represented as a factor graph: a bipartite graph with nodes for the variables and for the factors, where a factor node is connected by an edge to every variable that it depends on. The algorithm proceeds in rounds. In every round messages are sent along both direction of every edge. The outgoing message from a particular node is calculated based on the incoming messages to this node in the previous round from all other neighbors of the node. When the messages converge to a fixed point or a prescribed number of rounds has passed, the marginal distribution for every variable is estimated based on the fixed incoming messages into the variable node. The rule for computing the messages is such that if the graph is acyclic, then the estimates of the marginal distributions are exact. To understand these rules it is convenient to think of the graph as a rooted tree. It is easy to verify that the marginal distribution at the root can be computed from the marginal distributions of the roots of the subtrees below it, which can then be thought of as messages coming up the tree recursively.

Little is known about the behavior of belief propagation on general graphs. However it is applied successfully in many areas where the graphs have cycles, most notably in computer vision

[FPC00, CF02] and coding theory [RU01, KFL01, YFW05].

Belief propagation can be applied to constraint satisfaction problems in the following way: the uniform distribution on satisfying assignment is a Markov random field represented by the factor graph of the formula. Thus belief propagation can be used to estimate the probability that a given variable is 1 or 0 in a random satisfying assignment. Suppose the estimates are p_1 versus p_0. If this estimate were exact it would never be a mistake to assign a variable 1 (or 0) if $p_1 > 0$ (or $p_0 > 0$). Since p_0 and p_1 are just estimates, the most reasonable strategy is to choose the variable with largest value of $|p_0 - p_1|$ and assign it 1 if $p_1 > p_0$ and 0 otherwise. After a variable is assigned, belief propagation is applied again, and the process is repeated until all variables are assigned. This strategy of assigning variables one by one is called *decimation*. Belief propagation with decimation successfully finds satisfying assignments of random 3-SAT formulas with clause density lower than approximately 3.92. For formulas with higher clause density the belief propagation equations do not converge to a fixed point. This is consistent with the hypothesis from statistical physics that in the regime with $\alpha \geq 3.92$ there are multiple states, because, in general, belief propagation is not expected to produce good results when there are long-range correlations present. Intuitively, the reason is that there is an underlying assumption behind the message-passing rules of the algorithm that the messages arriving from different neighbors of a variable are essentially independent.

Survey propagation is designed to circumvent the issue of long-range correlations. In contrast to belief propagation, the survey propagation algorithm has been derived only for specific problems. In the original derivation for 3-SAT [MPZ02, BMZ03], the messages are interpreted as "surveys" taken over the clusters in the solution space, and provide information about the fraction of clusters in which a given variable is free or frozen. Decimation by survey propagation results in a partial assignment to the variables, which determines a particular cluster of assignments. An assignment for the rest of the variables is found using an algorithm that works in the single-state regime, such as the random-walk algorithm Walk-SAT. This strategy is successful in practice for formulas with density of clauses very close to the satisfiability threshold ($\alpha \leq 4.25$).

Prior to the work presented here, the relationship between survey propagation and belief propagation was not understood. We show that survey propagation can be interpreted as an instantiation of belief propagation, and thus as a method for computing (approximations) to marginal distributions in a certain Markov random field (MRF). The starting point of this thesis is precisely creating this bridge between the two methods. The rest of the results presented here are motivated by this connection.

1.1 Summary of results

Survey propagation as a belief propagation algorithm. We start by presenting a novel conceptual perspective on survey propagation. We introduce a new family of Markov random fields that are associated with a given k-SAT problem and show how a range of algorithms—including survey propagation as a special case—can all be recovered as instances of the belief propagation algorithm, as applied to suitably restricted MRFs within this family.

The configurations in our extended MRFs have a natural interpretation as *partial satisfying assignments* (i.e. assignments in $\{0, 1, *\}^n$) in which a subset of variables are assigned 0 or 1 in such a way that the remaining formula does not contain any empty or unit clauses. These partial assignments include as a subset the summaries of clusters illustrated in Figure 1.1. The assignments are weighted depending on the number of unassigned variables and on the number of assigned variables that are not the unique satisfying variable of any fully assigned clause. The latter are called *unconstrained variables*. The distribution has two parameters $\omega_o, \omega_* \in [0, 1]$. The probability of any assignment $\mathbf{x} \in \{0, 1, *\}^n$ is

$$\Pr[\mathbf{x}] \propto \omega_o^{n_o(\mathbf{x})} \times \omega_*^{n_*(\mathbf{x})}, \tag{1.1}$$

where $n_*(\mathbf{x})$ is the number of unassigned variables, and $n_o(\mathbf{x})$ is the number of unconstrained variables in $\mathbf{x}$. Survey propagation corresponds to setting the parameters as $\omega_* = 1$ and $\omega_o = 0$, whereas the original naive application of belief propagation corresponds to setting the parameters to $\omega_* = 0$, $\omega_o = 1$.

To provide some geometrical intuition for our results, it is convenient to picture these partial assignments as arranged in layers depending on the number of assigned variables, so that the top layer consists of fully assigned satisfying configurations. Figure 1.2 provides an idealized illustration of the space of partial assignments viewed in this manner. For random formulas with clause density in the regime where multiple clusters are present, the set of fully assigned configurations is separated into disjoint clusters that cause local message-passing algorithms like belief propagation to break down. Our results suggest that the introduction of partial satisfying assignments yields a *modified search space* that is far less fragmented, thereby permitting a local algorithm like belief propagation to find solutions.

We consider a natural partial ordering associated with this enlarged space, and we refer to minimal elements in this partial ordering as *cores*. We prove that any core is a fixed point of survey propagation ($\omega_* = 1$, $\omega_o = 0$). This fact indicates that each core represents a summary of one

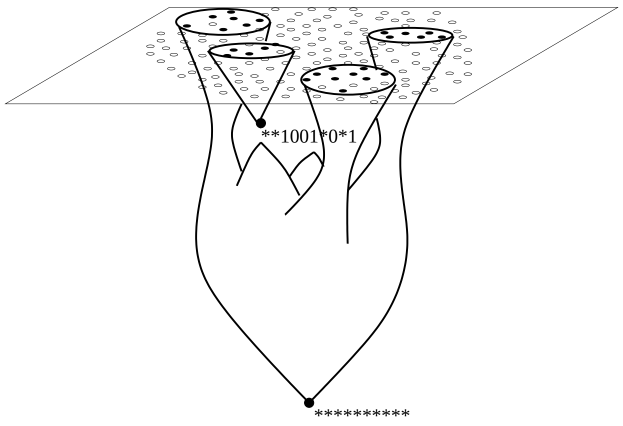


Figure 1.2. The set of fully assigned satisfying configurations occupy the top plane, and are arranged into clusters. Enlarging to the space of partial assignments leads to a new space with better connectivity. Minimal elements in the partial ordering are known as cores. Each core corresponds to one or more clusters of solutions from the top plane. In this example, one of the clusters has as a core a non-trivial partial assignment, whereas the others are connected to the all-$*$ assignment.

cluster of solutions. However, our experimental results for $k = 3$ indicate that the solution space of a random formula typically has only a trivial core (i.e., the empty assignment). This observation motivates a deeper study of the full family of Markov random fields for the range $0 \leq \omega_*, \omega_o \leq 1$, as well as the associated belief propagation algorithms. Accordingly, we study the lattice structure of the partial assignments, and prove a combinatorial identity that reveals how the distribution for $\omega_*, \omega_o \in (0, 1)$ can be viewed as a "smoothed" version of the MRF with $(\omega_*, \omega_o) = (0, 1)$. Our experimental results on the corresponding belief propagation algorithms indicate that they are most effective for values of the pair (ω_*, ω_o) close to and not necessarily equal to $(1, 0)$. The near-core assignments which are the ones of maximum weight in this case, may correspond to quasi-solutions of the cavity equations, as defined by Parisi [Par02].

The fact that survey propagation is a form of belief propagation was first conjectured by Braunstein et al. [BMZ03], and established independently of our work by Braunstein and Zecchina [BZ04]. In other independent work, Aurell et al. [AGK05] provided an alternative derivation of $\mathrm{SP}(1)$ that established a link to belief propagation. However, both of these papers treat only the case $(\omega_*, \omega_o) = (1, 0)$, and do not provide a combinatorial interpretation based on an underlying Markov random field. The results established here are a strict generalization, applying to the full range of $\omega_*, \omega_o \in [0, 1]$. Moreover, the structures intrinsic to our Markov random fields—namely cores and lattices—place the survey propagation algorithm on a combinatorial foundation. As we discuss later, this combinatorial perspective has already inspired subsequent work [ART06] on survey propagation for satisfiability problems.

A new method for bounding the satisfiability threshold. The family of Markov random fields on partial assignments that we define in association with the survey propagation algorithm can also be used to study the satisfiability threshold. In particular, the sum of their weights (where the weight of $\mathbf{x}$ is defined as $\omega_*^{n_*(\mathbf{x})}\omega_o^{n_o(\mathbf{x})}$) is always at least 1 when the formula is satisfiable. Therefore, showing that the expected value of the sum of their weights vanishes implies that formulas are with high probability unsatisfiable. This is an example of the first-moment method. Applying this idea directly unfortunately does not yield an improvement on the best upper bound of the satisfiability threshold, which is currently 4.51 [KKL00]. However, if we consider only assignments that have non-trivial cores, it is possible to show that above density $\alpha \geq 4.46$ they do not exist with high probability.

Classifying Boolean CSPs according to connectivity of the solution space. The original derivation of survey propagation as well as our analysis of the algorithm focus on the k-SAT problem. In fact, the replica symmetry breaking analysis can be done for other Boolean constraint satisfaction problems, and respectively the algorithm can be derived. However, before doing the analysis and solving approximately the corresponding distributional equations by population dynamics, there is no way to know which problems lead to symmetry breaking, i.e. the presence of multiple states below the satisfiability threshold. For example, it is known that for 2-SAT there is only a single state for any clause density below the satisfiability threshold, whereas for k-SAT with $k \geq 3$ this is not the case. Ultimately, we would like to be able to make more general (and rigorous) statements about phase properties and the performance of algorithms both for larger classes of problems, as well as larger classes of random models.

As was already mentioned, the worst-case complexity of all Boolean constraint satisfaction problems was determined by Schaefer almost three decades ago. He proved a remarkable *dichotomy theorem* stating that the satisfiability problem is in P for certain classes of Boolean formulas, while it is NP-complete for all other classes. This result pinpoints the computational complexity of all well-known variants of SAT, such as 3-SAT, HORN 3-SAT, NOT-ALL-EQUAL 3-SAT, and 1-IN-3-SAT. Much less is known about algorithms and computational hardness of *random* instances of Boolean constraint satisfaction problems. Identifying common properties between such problems is an intriguing goal, which has lead to some conjectures as well as rebuttals (e.g. [MZK$^+$99, ACIM01]).

In this thesis, we explore the phenomenon of clustering of solutions in the solution space as illustrated in Figure 1.1. To make the definition of clusters mathematically precise, we define two solutions of a given n-variable Boolean formula φ to be neighbors if and only if they differ in exactly one variable. Under this definition, clusters are simply the connected components of the subgraph of the n-dimensional hypercube that is induced by the solutions of φ. We denote this subgraph by $G(\varphi)$. We consider questions relating to this graph only from a worst-case viewpoint; however, even under this condition we get a non-trivial classification of Boolean constraint satisfaction problems into two classes with very different properties.

We address both algorithmic problems related to the solution space as well as the structural properties of Boolean satisfiability problems. We study the computational complexity of the following problems (i) Is $G(\varphi)$ connected? (ii) Given two solutions $\mathbf{s}$ and $\mathbf{t}$ of φ, is there a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(\varphi)$? We call these the *connectivity problem* and the *st-connectivity problem* respectively. On the structural side, we study the diameter of the solution graph of Boolean constraint satisfaction problems.

We identify two broad classes of relations with respect to the structure of the solution graphs of Boolean formulas built using these relations. The boundary between these two classes differs from the boundary in Schaefer's dichotomy. Schaefer showed that the satisfiability problem is solvable in polynomial time precisely for formulas built from Boolean relations all of which are bijunctive, or all of which are Horn, or all of which are dual Horn, or all of which are affine. We identify new classes of Boolean relations, called *tight* relations, that properly contain the classes of bijunctive, Horn, dual Horn, and affine relations. The solution graphs of formulas built from tight relations are characterized by certain simple structural properties. On the other hand we find *non-tight* sets of relations; formulas built from such sets of relations can express any solution graph.

The main step in the proof of Schaefer's dichotomy theorem is a result of independent interest known as Schaefer's expressibility theorem. The crux of our results is a different expressibility theorem which we call the *Faithful Expressibility Theorem* (FET). At a high level, this theorem asserts that for any Boolean relation with a solution graph G, we can construct a formula using any non-tight set of relations, such that its solution graph is isomorphic to G after certain adjacent vertices are merged. In addition to being an interesting structural result in its own right, the FET implies that all non-tight relations have the same computational complexity for both the connectivity and the st-connectivity problems. It also shows that the diameters of the solution graphs of formulas obtainable from such relations are polynomially related.

As a consequence of the FET we establish three dichotomy results. The first is a dichotomy theorem for the st-connectivity problem: we show that st-connectivity is solvable in linear time for formulas built from tight relations, and is PSPACE-complete in all other cases. The second is a dichotomy theorem for the connectivity problem: it is in coNP for formulas built from tight relations, and PSPACE-complete in all other cases. Finally, we establish a structural dichotomy theorem for the diameter of the connected components of the solution space of Boolean formulas. This result asserts that, in the PSPACE-complete cases, the diameter of the connected components can be exponential, but in all other cases it is linear.

Source coding via generalized belief propagation. The methodology of partial assignments that we developed to describe survey propagation as a belief propagation algorithm may also open the door to other problems where a complicated landscape prevents local search algorithms from finding good solutions. As a concrete example, we show that related ideas can be leveraged to perform lossy data compression at near-optimal (Shannon limit) rates.

As was mentioned earlier, the belief propagation algorithm is commonly used for the decoding of graphical error-correcting codes such as LDPC (low-density parity check) codes. It is natural to expect that the dual problem of data compression can also be tackled using this algorithm. However, attempts in that direction have not led to a working algorithm—the messages generally do not converge. The intuition is that while in the case of error-correcting codes there is one codeword that is most attractive, in the case of data compression there are many equally good compressions and the messages keep oscillating between them.

Here we propose another approach that is very similar to the one that we took in the analysis of the survey propagation algorithm. An extended MRF on $\{0, 1, *\}$ assignments is defined for LDGM (low-density generator matrix) codes. The belief propagation messages are derived in the same way as for the MRF for the k-SAT problem. We implement this algorithm and present experimental evidence that it has very promising performance, at least in the special case of a Bernoulli source.

1.2 Organization

The next chapter contains general background that will be used throughout the thesis—in particular, the precise definitions of constraint satisfaction problems and of the belief propagation algorithm. The connection between survey propagation and the belief propagation algorithm is established in Chapter 3; this chapter is based on joint work with Elchanan Mossel and Martin

Wainwright [MMW05]. The combinatorial structure of the new MRF and our application of the first-moment method to it is given in Chapter 4; this chapter is based on unpublished joint work with Alistair Sinclair, and also on work with Federico Ardila and Elchanan Mossel. In Chapter 5 we present our dichotomy results on the connectivity of the space of solutions of general Boolean constraint satisfaction problems; this chapter is based on joint work with Parikshit Gopalan, Phokion Kolaitis, and Christos Papadimitriou [GKMP06]. The application of the method developed in Chapter 3 to the source-coding problem is presented in Chapter 6; this chapter is based on joint work with Martin Wainwright [WM05].

Chapter 2

Technical preliminaries

The central theme of this thesis is the application of an inference heuristic known as belief propagation to constraint satisfaction problems where the problem instance is chosen from a particular probability distribution. In this chapter we introduce both the basic concepts relating to Boolean constraint satisfaction and the general belief propagation algorithm.

2.1 Boolean constraint satisfaction problems

2.1.1 Definitions

A *logical relation* R of arity $k \geq 1$ is defined as a non-empty subset of $\{0, 1\}^k$. Let $\mathcal{S}$ be a finite set of logical relations. A CNF$(\mathcal{S})$-*formula* over a set of variables $V = \{x_1, \dots, x_n\}$ is a finite conjunction $C_1 \wedge \cdots \wedge C_m$ of clauses built using relations from $\mathcal{S}$, variables from V, and the constants 0 and 1; this means that each C_i is an expression of the form $R(\xi_1, \dots, \xi_k)$, where $R \in \mathcal{S}$ is a relation of arity k, and each ξ_j is a variable in V or one of the constants 0, 1.

The *satisfiability problem* SAT$(\mathcal{S})$ associated with a finite set $\mathcal{S}$ of logical relations asks: given a CNF$(\mathcal{S})$-formula φ, is φ satisfiable? All well known restrictions of Boolean satisfiability, such as 3-SAT, NOT-ALL-EQUAL 3-SAT (also written as NAE-3-SAT), and POSITIVE 1-IN-3-SAT, can be cast as SAT$(\mathcal{S})$ problems, for a suitable choice of $\mathcal{S}$. For instance, POSITIVE 1-IN-3SAT is SAT$(\{R_{1/3}\})$, where $R_{1/3} = \{100, 010, 001\}$. The most common of these problems is k-SAT, which is SAT$(\{D_0, D_1, \dots, D_k\})$, where $D_r = \{0, 1\}^k \backslash \{(0^r\ 1^{k-r})\}$. A CNF$(\mathcal{S}_k)$-formula is also referred to as a k-CNF formula.

We will also write the clauses of a k-CNF formula in the standard notation, for example

$(x_1 \vee \bar{x}_2 \vee x_3)$ corresponds to $D_1(x_2, x_1, x_3)$.

2.1.2 Computational hardness

In 1978 Schaefer [Sch78] identified the worst-case complexity of *every* satisfiability problem $\textsc{Sat}(\mathcal{S})$. He determined several basic classes of relations that lead to polynomial time solvable satisfiability problems:

Definition 1. Let R be a logical relation.

1. R is *bijunctive* if it is the set of solutions of a 2-CNF formula.
2. R is *Horn* if it is the set of solutions of a Horn formula, where a Horn formula is a CNF formula such that each conjunct has at most one positive literal.
3. R is *dual Horn* if it is the set of solutions of a dual Horn formula, where a dual Horn formula is a CNF formula such that each conjunct has at most one negative literal.
4. R is *affine* if it is the set of solutions of a system of linear equations over $\mathbb{Z}_2$.

A set of logical relations $\mathcal{S}$ is called *Schaefer* if at least one of the following conditions holds: every relation in $\mathcal{S}$ is bijunctive, or every relation in $\mathcal{S}$ is Horn, or every relation in $\mathcal{S}$ is dual Horn, or every relation in $\mathcal{S}$ is affine.

Theorem 1. (Schaefer's Dichotomy Theorem [Sch78]) *If $\mathcal{S}$ is Schaefer, then $\textsc{Sat}(\mathcal{S})$ is in* P*; otherwise, $\textsc{Sat}(\mathcal{S})$ is* NP*-complete.*

Furthermore, there is a cubic algorithm for determining, given a finite set $\mathcal{S}$ of relations, whether $\textsc{Sat}(\mathcal{S})$ is in P or NP-complete (the input size is the sum of the sizes of relations in $\mathcal{S}$). Schaefer relations can be characterized in terms of *closure* properties [Sch78]. A relation R is bijunctive if and only if it is closed under the *majority* operation (if $\mathbf{a}, \mathbf{b}, \mathbf{c} \in R$, then $\mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \in R$, where $\mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ is the vector whose i-th bit is the majority of a_i, b_i, c_i). A relation R is Horn if and only if it is closed under $\vee$ (if $\mathbf{a}, \mathbf{b} \in R$, then $\mathbf{a} \vee \mathbf{b} \in R$, where, $\mathbf{a} \vee \mathbf{b}$ is the vector whose i-th bit is $a_i \vee b_i$). Similarly, R is dual Horn if and only if it is closed under $\wedge$. Finally, R is affine if and only if it is closed under $\mathbf{a} \oplus \mathbf{b} \oplus \mathbf{c}$.

While Schaefer's theorem completely identifies the worst-case complexity of all Boolean CSP, much less is known about the hardness of finding a solution if a formula is chosen from some natural probability distribution on $\mathrm{CNF}(\mathcal{S})$. Most of the existing work on algorithms for random instances of constraint satisfaction problems has been on the k-SAT problem. By Schaefer's theorem k-SAT is NP-complete for $k \geq 3$, and in P for $k \leq 2$. The most natural distribution on k-CNF formulas, and one that has been studied the most, is the following: for a fixed constant $\alpha > 0$, choose $m = \alpha n$ k-clauses uniformly at random by first choosing a random set of k variables, and then choosing a random relation out of $\mathcal{S}_k$. It is common to refer to α as the *density* of the formula. It is clear that a random formula becomes harder to satisfy as α increases. In 1999 Friedgut proved the following theorem:

Theorem 2. (Friedgut's Theorem [Fri99]) *For every $k \geq 2$ there exists a function $\alpha_c(n)$ such that for every $\epsilon > 0$:*

$$\begin{aligned}
\Pr[\,\textit{a random } k\text{-CNF } \textit{formula of density } \alpha_c(n) - \epsilon \textit{ is satisfiable}] &\rightarrow 1 \\
\Pr[\,\textit{a random } k\text{-CNF } \textit{formula of density } \alpha_c(n) + \epsilon \textit{ is satisfiable}] &\rightarrow 0
\end{aligned}$$

The function $\alpha_c(n)$ is the *threshold function* for k-SAT. It is conjectured that $\alpha_c(n)$ does not depend on n. For $k = 2$ it is known that $\alpha_c(n) = 1$ [Goe96, CR92, dlV92]. For larger k only bounds on the threshold function are known. In particular, for $k = 3$, it is known that $3.52 \leq \alpha_c(n) \leq 4.51$ [KKL00, DBM00]. For general k, it is easy to see that $\alpha_c(n) \leq 2^k \ln 2$, and an almost matching lower bound $\alpha_c(n) \geq 2^k \ln 2 - \frac{(k+1)\ln 2 + 3}{2}$ was proved in [AP03].

Other random Boolean constraint satisfaction problems have also been studied. For example for 1-IN-k-SAT the threshold has been found to be $1/\binom{k}{2}$ [ACIM01]. The same work provides bounds for the satisfiability threshold of NAE-3-SAT.

2.2 Belief propagation

Belief propagation is a widely-used algorithm for computing approximations to marginal distributions in general Markov random fields [YFW03, KFL01]. It has been applied widely in statistical inference, computer vision, and more recently in error-correcting codes. It also has a variational interpretation as an iterative method for attempting to solve a non-convex optimization problem based on the Bethe approximation [YFW03].

2.2.1 Definition

Belief propagation is an inference algorithm for a particular kind of factorized joint probability distribution. The distribution is represented as a graph, and the algorithm proceeds by passing messages along the edges of the graph according to a set of message-passing rules.

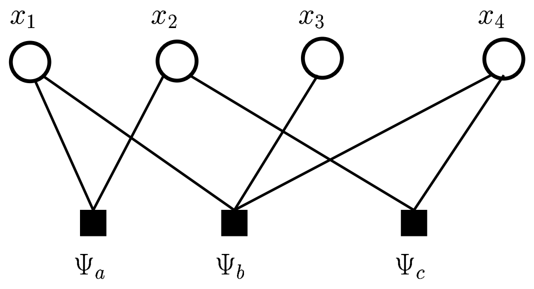


Figure 2.1. An example of a factor graph. Round nodes correspond to variables, while square nodes correspond to functions. The distribution corresponding to this graph is factorized as: $p(x_1, x_2, x_3, x_4) = \frac{1}{Z}\, \Psi_a(x_1, x_2) \times \Psi_b(x_1, x_3, x_4) \times \Psi_c(x_2, x_4)$.

Let $x_1, x_2, \ldots, x_n$ be variables taking values in a finite domain D. Subsets $V(a) \subset \{1, \ldots, n\}$ are indexed by $a \in C$, where $|C| = m$. Given a subset $S \subseteq \{1, 2, \ldots, n\}$, we define $\mathbf{x}_S := \{x_i \mid i \in S\}$. Consider a probability distribution p over $x_1, \ldots, x_n$ that can be factorized as

$$p(x_1, x_2, \ldots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} \Psi_i(x_i) \prod_{a \in C} \Psi_a\left(\mathbf{x}_{V(a)}\right), \tag{2.1}$$

where $\Psi_i(x_i)$ and $\Psi_a\left(\mathbf{x}_{V(a)}\right)$ are non-negative real functions, referred to as *compatibility functions*, and

$$Z := \sum_{x_1, \ldots, x_n} \left[\prod_{i=1}^{n} \Psi_i(x_i) \prod_{a \in C} \Psi_a\left(\mathbf{x}_{V(a)}\right) \right]$$

is the normalization constant or *partition function*. A factor graph representation of this probability distribution is a bipartite graph with vertices V corresponding to the variables, called *variable nodes*, and vertices C corresponding to the sets $V(a)$, called *function nodes*. There is an edge between a variable node i and function node a if and only if $i \in V(a)$. We define also $C(i) := \{a \in C : i \in V(a)\}$.

Suppose that we wish to compute the marginal probability of a single variable i, namely:

$$p(x_i) = \sum_{x_1 \in D} \cdots \sum_{x_{i-1} \in D} \sum_{x_{i+1} \in D} \cdots \sum_{x_n \in D} p(x_1, \ldots, x_n).$$

The *belief propagation* or *sum-product* algorithm is an efficient algorithm for computing the marginal probability distribution of each variable, assuming that the factor graph is acyclic [KFL01]. Suppose the tree is rooted at x_i. The essential idea is to use the distributive property of the sum and product operations to compute independent terms for each subtree recursively. This recursion can be cast as a message-passing algorithm, in which messages are passed up the tree. In particular, let the vector $M_{i\to a}$ denote the message passed by variable node i to function node a; similarly, the quantity $M_{a\to i}$ denotes the message that function node a passes to variable node i.

The messages from function nodes to variable nodes are updated in the following way:

$$M_{a\to i}(x_i) \quad \propto \quad \sum_{\mathbf{x}_{V(a)\setminus\{i\}}} \left[\psi_a\left(\mathbf{x}_{V(a)}\right) \prod_{j\in V(a)\setminus\{i\}} M_{j\to a}(x_j) \right]. \tag{2.2}$$

The messages from variable nodes to function nodes are updated as follows:

$$M_{i\to a}(x_i) \quad \propto \quad \psi_i(x_i) \prod_{b\in C(i)\setminus\{a\}} M_{b\to i}(x_i). \tag{2.3}$$

It is straightforward to show that for a factor graph without cycles, these updates will converge after a linear number of iterations. Upon convergence, the local marginal distributions at variable nodes and function nodes can be computed, using the message fixed point $\widehat{M}$, as follows:

$$F_i(x_i) \quad \propto \quad \psi_i(x_i) \prod_{b\in C(i)} \widehat{M}_{b\to i}(x_i) \tag{2.4a}$$

$$F_a\left(\mathbf{x}_{V(a)}\right) \quad \propto \quad \psi_a\left(\mathbf{x}_{V(a)}\right) \prod_{j\in V(a)} \widehat{M}_{j\to a}(x_j). \tag{2.4b}$$

The same updates, when applied to a general graph, are no longer exact due to the presence of cycles. However, for certain problems, including error-control coding, applying belief propagation to a graph with cycles gives excellent results. The algorithm is initialized by sending random messages on all edges, and is run until the messages converge to fixed values, or if the messages do not converge, until some fixed number of iterations [KFL01].

2.2.2 Application to constraint satisfaction problems

Given a constraint satisfaction problem we can describe it as a factorized distribution in the following way. For any clause C_a define a function on the set of variables that it constrains $\mathbf{x}_{V(a)}$, such that $\psi_a(\mathbf{x}_{V(a)}) = 1$ if the clause is satisfied and 0 otherwise. For example, for the k-SAT problem the function corresponding to clause $a \in C$ is $\psi_a(\mathbf{x}) = 1 - \prod_{i\in V(a)} \delta(J_{a,i}, x_i)$, where $J_{a,i}$ is 1 if variable x_i is negated in clause a, 0 otherwise, and $\delta(x,y)$ is 1 if $x = y$ and 0 otherwise.

Using these functions, let us define a probability distribution over binary sequences as

$$p(\mathbf{x}) \quad := \quad \frac{1}{Z} \prod_{a\in C} \psi_a(\mathbf{x}_{V(a)}), \tag{2.5}$$

where $Z := \sum_{\mathbf{x}\in\{0,1\}^n} \prod_{a\in C} \psi_a(\mathbf{x})$ is the normalization constant. Note that this definition makes sense if and only if the k-SAT instance is satisfiable, in which case the distribution (2.5) is simply the uniform distribution over satisfying assignments.

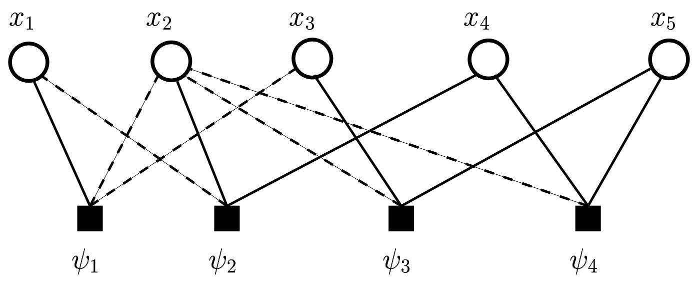


Figure 2.2. Factor graph representation of a 3-SAT problem on $n = 5$ variables with $m = 4$ clauses, in which circular and square nodes correspond to variables and clauses respectively. Solid and dotted edges correspond to positive and negative literals respectively. This graph corresponds to the formula $(x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee x_5) \wedge (\bar{x}_2 \vee x_4 \vee x_5)$.

This Markov random field representation (2.5) of any satisfiable formula motivates a marginalization-based approach to finding a satisfying assignment. In particular, suppose that we had an oracle that could compute exactly the marginal probability

$$p(x_i) = p(x_i) = \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_n} p(x_1, x_2, \ldots, x_n),$$

for a particular variable x_i. Note that this marginal reveals the existence of satisfying assignments with $x_i = 0$ (if $p(x_i = 0) > 0$) or $x_i = 1$ (if $p(x_i = 1) > 0$). Therefore, a satisfying assignment could be obtained by a recursive marginalization-decimation procedure, consisting of computing the marginal $p(x_i)$, appropriately setting x_i (i.e., decimating), and then recursing on the smaller formula.

Of course, exact marginalization is NP-hard; however, reducing the problem of finding a satisfying assignment to a marginalization problem allows one to use the belief propagation algorithm as an efficient heuristic. Even though the BP algorithm is not exact, a reasonable approach is to set the variable that has the largest bias towards a particular value, and repeat. We refer to the resulting algorithm as the “naive belief propagation algorithm”. This approach finds a satisfying assignment for α up to approximately 3.92 for $k = 3$; for higher α, however, the iterations for BP typically fail to converge [MPZ02, AGK05, BMZ03].

Chapter 3

Survey propagation as a belief propagation algorithm

As described in the introduction, survey propagation is an algorithm based on analysis via the cavity method and the 1-step replica symmetry breaking ansatz of statistical physics. A theoretical understanding of these methods is the object of a lot of current research, but still far from our grasp. This chapter provides a new conceptual perspective on the survey propagation algorithm, drawing a connection to the better understood belief propagation algorithm.

Although survey propagation can be generalized to other Boolean constraint satisfaction problems, for the sake of consistency with the rest of the literature on survey propagation we present it in the context of the k-SAT problem.

3.1 Description of survey propagation

In contrast to the naive BP approach, a marginalization-decimation approach based on *survey propagation* appears to be effective in solving random k-SAT problems even close to the satisfiability threshold [MPZ02, BMZ03]. Here we provide an explicit description of what we refer to as the $\mathrm{SP}(\rho)$ family of algorithms, where setting the parameter $\rho = 1$ yields the pure form of survey propagation. For any given $\rho \in [0, 1]$, the algorithm involves updating messages from clauses to variables, as well as from variables to clauses. Each clause $a \in C$ passes a real number $\eta_{a \to i} \in [0, 1]$ to each of its variable neighbors $i \in V(a)$ In the other direction, each variable $i \in V$ passes a triplet of real numbers $\Pi_{i \to a} = (\Pi^u_{i \to a}, \Pi^s_{i \to a}, \Pi^*_{i \to a})$ to each of its clause neighbors $a \in C(i)$ (that is the set of clauses that impose constraints on variable x_i). .

The set $C(i)$ of clauses can be decomposed into two disjoint subsets

$$C^-(i) := \{a \in C(i) \ : \ J_{a,i} = 1\}, \qquad C^+(i) := \{a \in C(i) \ : \ J_{a,i} = 0\},$$

according to whether the clause is satisfied by $x_i = 0$ or $x_i = 1$ respectively. Moreover, for each pair $(a, i) \in E$, the set $C(i)\backslash\{a\}$ can be divided into two (disjoint) subsets, depending on whether their preferred assignment of x_i *agrees* (in which case $b \in C_a^s(i)$) or *disagrees* (in which case $b \in C_a^u(i)$) with the preferred assignment of x_i corresponding to clause a. More formally, we define

$$C_a^s(i) := \{b \in C(i)\backslash\{a\} \ : \ J_{a,i} = J_{b,i} \ \}, \qquad C_a^u(i) := \{b \in C(i)\backslash\{a\} \ : \ J_{a,i} \neq J_{b,i} \ \}.$$

It will be convenient, when discussing the assignment of a variable x_i with respect to a particular clause a, to use the notation $s_{a,i} := 1 - J_{a,i}$ and $u_{a,i} := J_{a,i}$ to indicate, respectively, the values that are *satisfying* and *unsatisfying* for the clause a.

The precise form of the updates are given in Figure 3.1.

Message from clause a to variable i:

$$\eta_{a\to i} = \prod_{j \in V(a)\backslash\{i\}} \left[\frac{\Pi^u_{j\to a}}{\Pi^u_{j\to a} + \Pi^s_{j\to a} + \Pi^*_{j\to a}} \right]. \tag{3.1}$$

Message from variable i to clause a:

$$\Pi^u_{i\to a} = \left[1 - \rho \prod_{b\in C_a^u(i)} (1-\eta_{b\to i})\right] \prod_{b \in C_a^s(i)} (1-\eta_{b\to i}). \tag{3.2a}$$

$$\Pi^s_{i\to a} = \left[1 - \prod_{b\in C_a^s(i)} (1-\eta_{b\to i})\right] \prod_{b \in C_a^u(i)} (1-\eta_{b\to i}). \tag{3.2b}$$

$$\Pi^*_{i\to a} = \prod_{b\in C_a^s(i)} (1-\eta_{b\to i}) \prod_{b \in C_a^u(i)} (1-\eta_{b\to i}). \tag{3.2c}$$

Figure 3.1: $\mathrm{SP}(\rho)$ message updates

Although we have omitted the time step index for simplicity, equations (3.1) and (3.2) should be interpreted as defining a recursion on (η, Π). The initial values for η are chosen randomly in the interval $(0, 1)$.

The idea of the ρ parameter is to provide a smooth transition from the original naive belief propagation algorithm to the survey propagation algorithm. As shown in [BMZ03], setting $\rho = 0$

yields the belief propagation updates applied to the probability distribution (2.5), whereas setting $\rho = 1$ yields the pure version of survey propagation.

3.1.1 Intuitive "warning" interpretation

To gain intuition for these updates, it is helpful to consider the pure SP setting of $\rho = 1$. As described by Braunstein et al. [BMZ03], the messages in this case have a natural interpretation in terms of probabilities of warnings. In particular, at time $t = 0$, suppose that the clause a sends a warning message to variable i with probability $\eta^0_{a\to i}$, and a message without a warning with probability $1 - \eta^0_{a\to i}$. After receiving all messages from clauses in $C(i)\backslash\{a\}$, variable i sends a particular symbol to clause a saying either that it can't satisfy it ("u"), that it can satisfy it ("s"), or that it is indifferent ("$*$"), depending on what messages it got from its other clauses. There are four cases:

1. If variable i receives warnings from $C^u_a(i)$ and no warnings from $C^s_a(i)$, then it cannot satisfy a and sends "u".

2. If variable i receives warnings from $C^s_a(i)$ but no warnings from $C^u_a(i)$, then it sends an "s" to indicate that it is inclined to satisfy the clause a.

3. If variable i receives no warnings from either $C^u_a(i)$ or $C^s_a(i)$, then it is indifferent and sends "$*$".

4. If variable i receives warnings from both $C^u_a(i)$ and $C^s_a(i)$, a contradiction has occurred.

The updates from clauses to variables are especially simple: in particular, any given clause sends a warning if and only if it receives "u" symbols from all of its other variables.

In this context, the real-valued messages involved in the pure $\mathrm{SP}(1)$ all have natural probabilistic interpretations. In particular, the message $\eta_{a\to i}$ corresponds to the probability that clause a sends a warning to variable i. The quantity $\Pi^u_{j\to a}$ can be interpreted as the probability that variable j sends the "u" symbol to clause a, and similarly for $\Pi^s_{j\to a}$ and $\Pi^*_{j\to a}$. The normalization by the sum $\Pi^u_{j\to a} + \Pi^s_{j\to a} + \Pi^*_{j\to a}$ reflects the fact that the fourth case is a failure, and hence is excluded a priori from the probability distribution

Suppose that all of the possible warning events were independent. In this case, the SP message update equations (3.1) and (3.2) would be the correct estimates for the probabilities. This independence assumption is valid on a graph without cycles, and in that case the SP updates do have a rigorous probabilistic interpretation. It is not clear if the equations have a simple interpretation in the case $\rho \neq 1$.

3.1.2 Decimation based on survey propagation

Supposing that these survey propagation updates are applied and converge, the overall conviction of a value at a given variable is computed from the incoming set of equilibrium messages as

$$\begin{aligned}
\mu_i(1) &\propto \left[1 - \rho \prod_{b \in C^+(i)} (1 - \eta_{b \to i})\right] \prod_{b \in C^-(i)} (1 - \eta_{b \to i}). \\
\mu_i(0) &\propto \left[1 - \rho \prod_{b \in C^-(i)} (1 - \eta_{b \to i})\right] \prod_{b \in C^+(i)} (1 - \eta_{b \to i}). \\
\mu_i(*) &\propto \prod_{b \in C^+(i)} (1 - \eta_{b \to i}) \prod_{b \in C^-(i)} (1 - \eta_{b \to i}).
\end{aligned}$$

In order to be consistent with the interpretation of $\{\mu_i(0), \mu_i(*), \mu_i(1)\}$ as (approximate) marginal probabilities, they are normalized to sum to one. The *bias* of a variable node is defined as $B(i) := |\mu_i(0) - \mu_i(1)|$.

The marginalization-decimation algorithm based on survey propagation [BMZ03] consists of the following steps:

1. Run $\text{SP}(1)$ on the SAT problem. Extract the fraction β of variables with the largest biases, and set them to their preferred values.

2. Simplify the SAT formula, and return to Step 1.

Once the maximum bias over all variables falls below a pre-specified tolerance, the Walk-SAT algorithm is applied to the formula to find the remainder of the assignment (if possible). Intuitively, the goal of the initial phases of decimation is to find a cluster; once inside the cluster, the induced problem is considered easy to solve, meaning that any "local" algorithm should perform well within a given cluster.

3.2 Markov random fields over partial assignments

In this section, we show how a large class of message-passing algorithms—including the $\text{SP}(\rho)$ family as a particular case—can be recovered by applying the well-known belief propagation algorithm to a novel class of Markov random fields (MRFs) associated with any k-SAT problem. We begin by introducing the notion of a partial assignment, and then define a family of MRFs over these assignments.

3.2.1 Partial assignments

Suppose that the variables $\mathbf{x} = (x_1, \dots, x_n)$ are allowed to take values in $\{0, 1, *\}$, which we refer to as a *partial assignment*. An $*$ (star) assignment should be thought of as either an undecided variable, or as joker state, i.e. this variable's value is not essential to the satisfiability.

Definition 2. A partial assignment to $\mathbf{x}$ is *invalid* for a clause a if either

(a) all variables are unsatisfying (i.e., $x_i = u_{a,i}$ for all $i \in V(a)$), or

(b) all variables are unsatisfying except for one index $j \in V(a)$, for which $x_j = *$.

Otherwise, the partial assignment is valid for clause a, and we denote this event by $\mathrm{VAL}_a(\mathbf{x}_{V(a)})$. We say that a partial assignment is *valid* for a formula if it is valid for all of its clauses.

The motivation for deeming case (a) invalid is clear, in that any partial assignment that does not satisfy the clause must be excluded. Note that case (b) is also invalid, since (with all other variables unsatisfying) the variable x_j is effectively forced to $s_{a,i}$, and so cannot be assigned the $*$ symbol.

For a valid partial assignment, the subset of variables that are assigned either 0 or 1 values can be divided into *constrained* and *unconstrained* variables in the following way:

Definition 3. We say that a variable x_i is the *unique satisfying variable* for a clause if it is assigned $s_{a,i}$ whereas all other variables in the clause (i.e., the variables $\{x_j \ : \ j \in V(a) \backslash \{i\}\}$) are assigned $u_{a,j}$. A variable x_i is *constrained* by clause a if it is the unique satisfying variable.

We let $\mathrm{CON}_{i,a}(\mathbf{x}_{V(a)})$ denote an indicator function for the event that x_i is the unique satisfying variable in the partial assignment $\mathbf{x}_{V(a)}$ for clause a. A variable is *unconstrained* if it has 0 or 1 value, and is not constrained by any clause. Thus for any partial assignment the variables are divided into stars, constrained and unconstrained variables. We define the three sets

$$\begin{aligned}
S_*(\mathbf{x}) &:= \{i \in V : x_i = *\} \\
S_c(\mathbf{x}) &:= \{i \in V : x_i \text{ constrained}\} \\
S_o(\mathbf{x}) &:= \{i \in V : x_i \text{ unconstrained}\}
\end{aligned}$$

of $*$, constrained and unconstrained variables respectively. Finally, we use $n_*(x)$, $n_c(x)$ and $n_o(x)$ to denote the respective sizes of these three sets.

Various probability distributions can be defined on valid partial assignments by giving different weights to stars, constrained and unconstrained variables, which we denote by ω_c, ω_* and ω_o respectively. Since only the ratio of the weights matters, we set $\omega_c = 1$, and treat ω_o and ω_* as free non-negative parameters (we generally take them in the interval $[0, 1]$). We define the weights of partial assignments in the following way: invalid assignments $\mathbf{x}$ have weight $W(\mathbf{x}) = 0$, and for any valid assignment $\mathbf{x}$, we set

$$W(\mathbf{x}) := \omega_o^{n_o(\mathbf{x})} \times \omega_*^{n_*(\mathbf{x})}. \tag{3.3}$$

Our primary interest is the probability distribution given by $p_W(\mathbf{x}) \propto W(\mathbf{x})$. In contrast to the earlier distribution p, it is important to observe that this definition is valid for any SAT problem, whether or not it is satisfiable, as long as $\omega_* \neq 0$, since the all-$*$ vector is always a valid partial assignment. Note that if $\omega_o = 1$ and $\omega_* = 0$ then the distribution $p_W(\mathbf{x})$ is the uniform distribution on satisfying assignments. Another interesting case that we will discuss is that of $\omega_o = 0$ and $\omega_* = 1$, which corresponds to the uniform distribution over valid partial assignments without unconstrained variables.

3.2.2 Markov random fields

Given our set-up thus far, it is not at all obvious whether or not the distribution p_W can be decomposed as a Markov random field based on the original factor graph. Interestingly, we find that p_W does indeed have such a Markov representation for any choices of $\omega_o, \omega_* \in [0, 1]$. Obtaining this representation requires the addition of another dimension to our representation, which allows us to assess whether a given variable is constrained or unconstrained. We define the *parent set* of a given variable x_i, denoted by P_i, to be the set of clauses for which x_i is the unique satisfying variable. Immediate consequences of this definition are the following:

(a) If $x_i = 0$, then we must have $P_i \subseteq C^-(i)$.

(b) If $x_i = 1$, then there must hold $P_i \subseteq C^+(i)$.

(c) The setting $x_i = *$ implies that $P_i = \emptyset$.

Note also that $P_i = \emptyset$ means that x_i cannot be constrained. For each $i \in V$, let $\mathcal{P}(i)$ be the set of all possible parent sets of variable i. Due to the restrictions imposed by our definition, P_i

must be contained in either $C^+(i)$ or $C^-(i)$ but not both. Therefore, the cardinality[1] of $\mathcal{P}(i)$ is $2^{|C^-(i)|} + 2^{|C^+(i)|} - 1$.

Our extended Markov random field is defined on the Cartesian product space $\mathcal{X}_1 \times \ldots \times \mathcal{X}_n$, where $\mathcal{X}_i := \{0, 1, *\} \times \mathcal{P}(i)$. The distribution factorizes as a product of compatibility functions at the variable and clause nodes of the factor graph, which are defined as follows:

Variable compatibilities: Each variable node $i \in V$ has an associated compatibility function of the form:

$$\Psi_i(x_i, P_i) \;:=\; \begin{cases} \omega_o & : \quad P_i = \emptyset, x_i \neq * \\ \omega_* & : \quad P_i = \emptyset, x_i = * \\ 1 & : \quad \text{for any other valid } (P_i, x_i) \end{cases} \tag{3.4}$$

The role of these functions is to assign weight to the partial assignments according to the number of unconstrained and star variables, as in the weighted distribution p_W.

Clause compatibilities: The compatibility functions at the clause nodes serve to ensure that only valid assignments have non-zero probability, and that the parent sets $P_{V(a)} := \{P_i \;:\; i \in V(a)\}$ are consistent with the assignment on $x_{V(a)}$. More precisely, we require that the partial assignment $x_{V(a)}$ is valid for a (denoted by $\text{VAL}_a(x_{V(a)}) = 1$) and that for each $i \in V(a)$, exactly one of the two following conditions holds:

(a) $a \in P_i$ and x_i is constrained by a or

(b) $a \notin P_i$ and x_i is not constrained by a.

The following compatibility function corresponds to an indicator function for the intersection of these events:

$$\Psi_a\big(\mathbf{x}_{V(a)}, P_{V(a)}\big) := \text{VAL}_a(\mathbf{x}_{V(a)}) \times \prod_{i \in V(a)} \delta\big(\text{Ind}\, a \in P_i,\;\; \text{CON}_{a,i}(\mathbf{x}_{V(a)})\big). \tag{3.5}$$

We now form a Markov random field over partial assignments and parent sets by taking the product of variable (3.4) and clause (3.5) compatibility functions

$$p_{gen}(\mathbf{x}, P) \propto \prod_{i \in V} \Psi_i(x_i, P_i) \;\; \prod_{a \in C} \Psi_a\big(\mathbf{x}_{V_a}, P_{V(a)}\big). \tag{3.6}$$

With these definitions $p_{gen} = p_W$.

[1]Note that it is necessary to subtract one so as not to count the empty set twice.

3.2.3 Survey propagation as an instance of belief propagation

We now consider the form of the belief propagation (BP) updates as applied to the MRF p_{gen} defined by equation (3.6). We refer the reader to Section 2.2 for the definition of the BP algorithm on a general factor graph. The main result of this section is to establish that the $\mathrm{SP}(\rho)$ family of algorithms are equivalent to belief propagation as applied to p_{gen} with suitable choices of the weights ω_o and ω_*. In the interests of readability, most of the technical lemmas will be presented in the appendix.

We begin by introducing some notation necessary to describe the BP updates on the extended MRF. The BP message from clause a to variable i, denoted by $M_{a\to i}(\cdot)$, is a vector of length $|\mathcal{X}_i| = 3 \times |\mathcal{P}(i)|$. Fortunately, due to symmetries in the variable and clause compatibilities defined in equations (3.4) and (3.5), it turns out that the clause-to-variable message can be parameterized by only three numbers, $\{M^u_{a\to i}, M^s_{a\to i}, M^*_{a\to i}\}$, as follows:

$$M_{a\to i}(x_i, P_i) \quad = \quad \begin{cases} M^s_{a\to i} & \text{if } x_i = s_{a,i},\ P_i = S \cup \{a\} \text{ for some } S \subseteq C^s_a(i), \\ M^u_{a\to i} & \text{if } x_i = u_{a,i},\ P_i \subseteq C^u_a(i), \\ M^*_{a\to i} & \text{if } x_i = s_{a,i},\ P_i \subseteq C^s_a(i) \text{ or } x_i = *\ , P_i = \emptyset, \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

where $M^s_{a\to i}, M^u_{a\to i}$ and $M^*_{a\to i}$ are elements of $[0,1]$.

Now turning to messages from variables to clauses, it is convenient to introduce the notation $P_i = S \cup \{a\}$ as a shorthand for the event

$$a \in P_i \quad \text{and} \quad S = P_i \backslash \{a\} \subseteq C^s_a(i),$$

where it is understood that S could be empty. In Lemma 3, we show that the variable-to-clause message $M_{i\to a}$ is fully specified by values for pairs (x_i, P_i) of six general types:

$$(s_{a,i}, S \cup \{a\}),\ (s_{a,i}, \emptyset \neq P_i \subseteq C^s_a(i)),\ (u_{a,i}, \emptyset \neq P_i \subseteq C^u_a(i)),\ (s_{a,i}, \emptyset),\ (u_{a,i}, \emptyset),\ (*, \emptyset).$$

The BP updates themselves are most compactly expressed in terms of particular linear combinations of such basic messages, defined in the following way:

$$R^s_{i\to a} \quad := \quad \sum_{S \subseteq C^s_a(i)} M_{i\to a}(s_{a,i}, S \cup \{a\}) \tag{3.8a}$$

$$R^u_{i\to a} \quad := \quad \sum_{P_i \subseteq C^u_a(i)} M_{i\to a}(u_{a,i},\ P_i) \tag{3.8b}$$

$$R^*_{i\to a} \quad := \quad \sum_{P_i \subseteq C^s_a(i)} M_{i\to a}(s_{a,i}, P_i) + M_{i\to a}(*, \emptyset). \tag{3.8c}$$

Note that $R^s_{i\to a}$ is associated with the event that x_i is the unique satisfying variable for clause a; $R^u_{i\to a}$ with the event that x_i does not satisfy a; and $R^*_{i\to a}$ with the event that x_i is neither unsatisfying nor uniquely satisfying (i.e., either $x_i = *$, or $x_i = s_{a,i}$ but is not the only variable that satisfies a).

With this terminology, the BP algorithm on the extended MRF can be expressed in terms of a recursion on the triplets $(M^s_{a\to i}, M^u_{a\to i}, M^*_{a\to i})$ and $(R^s_{i\to a}, R^u_{i\to a}, R^*_{i\to a})$ as described in Figure 3.2.

Messages from clause a to variable i:

$$\begin{aligned}
M^s_{a\to i} &= \prod_{j\in V(a)\setminus\{i\}} R^u_{j\to a} \\
M^u_{a\to i} &= \prod_{j\in V(a)\setminus\{i\}} (R^u_{j\to a} + R^*_{j\to a}) + \sum_{k\in V(a)\setminus\{i\}} (R^s_{k\to a} - R^*_{k\to a}) \prod_{j\in V(a)\setminus\{i,k\}} R^u_{j\to a} \\
&\qquad - \prod_{j\in V(a)\setminus\{i\}} R^u_{j\to a} \\
M^*_{a\to i} &= \prod_{j\in V(a)\setminus\{i\}} (R^u_{j\to a} + R^*_{j\to a}) - \prod_{j\in V(a)\setminus\{i\}} R^u_{j\to a}.
\end{aligned}$$

Messages from variable i to clause a:

$$\begin{aligned}
R^s_{i\to a} &= \prod_{b\in C^u_a(i)} M^u_{b\to i}\Big[\prod_{b\in C^s_a(i)} (M^s_{b\to i} + M^*_{b\to i})\Big] \\
R^u_{i\to a} &= \prod_{b\in C^s_a(i)} M^u_{b\to i}\Big[\prod_{b\in C^u_a(i)} (M^s_{b\to i} + M^*_{b\to i}) - (1-\omega_o) \prod_{b\in C^u_a(i)} M^*_{b\to i}\Big] \\
R^*_{i\to a} &= \prod_{b\in C^u_a(i)} M^u_{b\to i}\Big[\prod_{b\in C^s_a(i)} (M^s_{b\to i} + M^*_{b\to i}) - (1-\omega_o) \prod_{b\in C^s_a(i)} M^*_{b\to i}\Big] \\
&\qquad + \omega_* \prod_{b\in C^s_a(i)\cup C^u_a(i)} M^*_{b\to i}.
\end{aligned}$$

Figure 3.2: BP message updates on extended MRF

Next we provide the derivation of these BP equations on the extended MRF.

Lemma 3 (Variable to clause messages)**.** *The variable to clause message vector $M_{i\to a}$ is fully specified by values for pairs (x_i, P_i) of the form:*

$$\{(s_{a,i}, S\cup\{a\}),\ \ (s_{a,i}, \emptyset \neq P_i \subseteq C^s_a(i)),\ \ (u_{a,i}, \emptyset \neq P_i \subseteq C^u_a(i)),\ \ (s_{a,i}, \emptyset),\ \ (u_{a,i}, \emptyset),\ \ (*, \emptyset)\}.$$

Specifically, the updates for these five pairs take the following form:

$$\begin{aligned}
M_{i\to a}(s_{a,i},\ P_i = S \cup \{a\}) &= \prod_{b\in S} M^s_{b\to i} \prod_{b\in C^s_a(i)\backslash S} M^*_{b\to i} \prod_{b\in C^u_a(i)} M^u_{b\to i} && \text{(3.11a)}\\
M_{i\to a}(s_{a,i},\ \emptyset \neq P_i \subseteq C^s_a(i)) &= \prod_{b\in P_i} M^s_{b\to i} \prod_{b\in C^s_a(i)\backslash P_i} M^*_{b\to i} \prod_{b\in C^u_a(i)} M^u_{b\to i} && \text{(3.11b)}\\
M_{i\to a}(u_{a,i},\ \emptyset \neq P_i \subseteq C^u_a(i)) &= \prod_{b\in P_i} M^s_{b\to i} \prod_{b\in C^u_a(i)\backslash P_i} M^*_{b\to i} \prod_{b\in C^s_a(i)} M^u_{b\to i} && \text{(3.11c)}\\
M_{i\to a}(s_{a,i},\ P_i = \emptyset) &= \omega_o \prod_{b\in C^s_a(i)} M^*_{b\to i} \prod_{b\in C^u_a(i)} M^u_{b\to i} && \text{(3.11d)}\\
M_{i\to a}(u_{a,i},\ P_i = \emptyset) &= \omega_o \prod_{b\in C^u_a(i)} M^*_{b\to i} \prod_{b\in C^s_a(i)} M^u_{b\to i} && \text{(3.11e)}\\
M_{i\to a}(*, P_i = \emptyset) &= \omega_* \prod_{b\in C(i)\backslash\{a\}} M^*_{b\to i}. && \text{(3.11f)}
\end{aligned}$$

Proof. The form of these updates follows immediately from the definition (3.4) of the variable compatibilities in the extended MRF, and the BP message update (2.3). □

Next, we compute the specific forms of the linear sums of messages defined in equation (3.8). First, we use the definition (3.8a) and Lemma 3 to compute the form of $R^s_{i\to a}$:

$$\begin{aligned}
R^s_{i\to a} &= \sum_{S\subseteq C^s_a(i)} M_{i\to a}(s_{a,i}, P_i = S \cup \{a\})\\
&= \sum_{S\subseteq C^s_a(i)} \prod_{b\in S} M^s_{b\to i} \prod_{b\in C^s_a(i)\backslash S} M^*_{b\to i} \prod_{b\in C^u_a(i)} M^u_{b\to i}\\
&= \prod_{b\in C^u_a(i)} M^u_{b\to i} \Big[\prod_{b\in C^s_a(i)} (M^s_{b\to i} + M^*_{b\to i}) \Big].
\end{aligned}$$

Similarly, the definition (3.8b) and Lemma 3 allows us compute the following form of $R^u_{i\to a}$:

$$\begin{aligned}
R^u_{i\to a} &= \sum_{S\subseteq C^u_a(i)} M_{i\to a}(u_{a,i}, P_i = S)\\
&= \sum_{S\subseteq C^u_a(i), S\neq\emptyset} \prod_{b\in S} M^s_{b\to i} \prod_{b\in C^u_a(i)\backslash S} M^*_{b\to i} \prod_{b\in C^s_a(i)} M^u_{b\to i} + \omega_o \prod_{b\in C^u_a(i)} M^*_{b\to i} \prod_{b\in C^s_a(i)} M^u_{b\to i}\\
&= \prod_{b\in C^s_a(i)} M^u_{b\to i} \Big[\prod_{b\in C^u_a(i)} (M^s_{b\to i} + M^*_{b\to i}) - (1-\omega_o) \prod_{b\in C^u_a(i)} M^*_{b\to i} \Big].
\end{aligned}$$

Finally, we compute $R^*_{i\to a}$ using the definition (3.8c) and Lemma 3:

$$\begin{aligned}
R^*_{i\to a} &= \Big[\sum_{S\subseteq C^s_a(i)} M_{i\to a}(s_{a,i}, P_i = S)\Big] + M_{i\to a}(*, P_i = \emptyset) \\
&= \Big[\sum_{S\subseteq C^s_a(i), S\neq\emptyset} \prod_{b\in S} M^s_{b\to i} \prod_{b\in C^s_a(i)\backslash S} M^*_{b\to i} \prod_{b\in C^u_a(i)} M^u_{b\to i}\Big] \\
&\quad +\omega_o \prod_{b\in C^s_a(i)} M^*_{b\to i} \prod_{b\in C^u_a(()i)} M^u_{b\to i} + \omega_* \prod_{b\in C^s_a(i)} M^*_{b\to i} \prod_{b\in C^u_a(i)} M^*_{b\to i} \\
&= \prod_{b\in C^u_a(i)} M^u_{b\to i}\Big[\prod_{b\in C^s_a(i)} (M^s_{b\to i} + M^*_{b\to i}) - (1-\omega_o) \prod_{b\in C^s_a(i)} M^*_{b\to i}\Big] \\
&\quad +\omega_* \prod_{b\in C^s_a(i)\cup C^u_a(i)} M^*_{b\to i}.
\end{aligned}$$

Lemma 4 (Clause to variable messages)**.** *The updates of messages from clauses to variables in the extended MRF take the following form:*

$$M^s_{a\to i} = \prod_{j\in V(a)\backslash\{i\}} R^u_{j\to a} \tag{3.12a}$$

$$M^u_{a\to i} = \prod_{j\in V(a)\backslash\{i\}} (R^u_{j\to a} + R^*_{j\to a}) \tag{3.12b}$$

$$+ \sum_{k\in V(a)\backslash\{i\}} (R^s_{k\to a} - R^*_{k\to a}) \prod_{j\in V(a)\backslash\{i,k\}} R^u_{j\to a} - \prod_{j\in V(a)\backslash\{i\}} R^u_{j\to a} \tag{3.12c}$$

$$M^*_{a\to i} = \prod_{j\in V(a)\backslash\{i\}} (R^u_{j\to a} + R^*_{j\to a}) - \prod_{j\in V(a)\backslash\{i\}} R^u_{j\to a}. \tag{3.12d}$$

Proof. (i) We begin by proving equation (3.12a). When $x_i = s_{a,i}$ and $P_i = S\cup\{a\}$ for some $S \subseteq C^s_a(i)$, then the only possible assignment for the other variables at nodes in $V(a)\backslash\{i\}$ is $x_j = u_{a,j}$ and $P_j \subseteq C^u_a(j)$. Accordingly, using the BP update equation (2.2), we obtain the following update for $M^s_{a\to i} = M_{a\to i}(s_{a,i}, P_i = S\cup\{a\})$:

$$\begin{aligned}
M^s_{a\to i} &= \prod_{j\in V(a)\backslash\{i\}} \sum_{P_j\subseteq C^u_a(j)} M_{j\to a}(u_{a,j}, P_j) \\
&= \prod_{j\in V(a)\backslash\{i\}} R^u_{j\to a}.
\end{aligned}$$

(ii) Next we prove equation (3.12d). In the case $x_i = *$ and $P_i = \emptyset$, the only restriction on the other variables $\{x_j :\ j\in V(a)\backslash\{i\}\}$ is that they are not all unsatisfying. The weight assigned to the event that they are all unsatisfying is

$$\sum_{\left\{S_j\subseteq C_a^u(j)\,:\,j\in V(a)\backslash\{i\}\right\}} \prod_{j\in V(a)\backslash\{i\}} M_{j\to a}(u_{a,j},S_j) = \prod_{j\in V(a)\backslash\{i\}} \Big[\sum_{S_j\subseteq C_a^u(j)} M_{j\to a}(u_{a,j},S_j)\Big] = \prod_{j\in V(a)\backslash\{i\}} R^u_{j\to a}. \tag{3.13}$$

On the other hand, the weight assigned to the event that each is either unsatisfying, satisfying or $*$ can be calculated as follows. Consider a partition $J^u \cup J^s \cup J^*$ of the set $V(a)\backslash\{i\}$, where J^u, J^s and J^* corresponds to the subsets of unsatisfying, satisfying and $*$ assignments respectively. The weight $W(J^u, J^s, J^*)$ associated with this partition takes the form

$$\sum_{\left\{S_j\subseteq C_a^u(j)\,:\,j\in J^u\right\}} \sum_{\left\{S_j\subseteq C_a^s(j)\,:\,j\in J^s\right\}} \prod_{j\in J^u} M_{j\to a}(u_{a,j},S_j) \prod_{j\in J^s} M_{j\to a}(s_{a,j},S_j) \prod_{j\in J^*} M_{j\to a}(*,\emptyset).$$

Simplifying by distributing the sum and product leads to

$$\begin{aligned} W(J^u,J^s,J^*) &= \prod_{j\in J^u}\Big[\sum_{S_j\subseteq C_a^u(j)} M_{j\to a}(u_{a,j},S_j)\Big] \prod_{j\in J^s}\Big[\sum_{S_j\subseteq C_a^s(j)} M_{j\to a}(s_{a,j},S_j)\Big] \\ &\quad \prod_{j\in J^*} M_{j\to a}(*,\emptyset) \\ &= \prod_{j\in J^u} R^u_{j\to a} \prod_{j\in J^s}\left[R^*_{j\to a} - M_{j\to a}(*,\emptyset)\right] \prod_{j\in J^*} M_{j\to a}(*,\emptyset). \end{aligned}$$

Now summing the $W(J^u, J^s, J^*)$ over all partitions $J^u \cup J^s \cup J^*$ of $V(a)\backslash\{i\}$ yields

$$\begin{aligned} &\sum_{J^u\cup J^s\cup J^*} W(J^u,J^s,J^*) \\ &= \sum_{J^u\subseteq V(a)\backslash\{i\}} \prod_{j\in J^u} R^u_{j\to a} \\ &\quad \sum_{J^s\cup J^*=V(a)\backslash\{J^u\cup i\}} \Big\{\prod_{j\in J^s}\left[R^*_{j\to a} - M_{j\to a}(*,\emptyset)\right] \prod_{j\in J^*} M_{j\to a}(*,\emptyset\Big\} \qquad (3.14) \\ &= \sum_{J^u\subseteq V(a)\backslash\{i\}} \prod_{j\in J^u} R^u_{j\to a} \prod_{j\in V(a)\backslash\{J^u\cup i\}} R^*_{j\to a} \\ &= \prod_{j\in V(a)\backslash\{i\}} \left[R^u_{j\to a} + R^*_{j\to a}\right], \qquad (3.15) \end{aligned}$$

where we have used the binomial identity twice. Overall, equations (3.13) and (3.15) together yield that

$$M^*_{a\to i} = \prod_{j\in V(a)\backslash\{i\}} \left[R^u_{j\to a} + R^*_{j\to a}\right] - \prod_{j\in V(a)\backslash\{i\}} R^u_{j\to a},$$

which establishes equation (3.12d).

(iii) Finally, turning to equation (3.12b), for $x_i = u_{a,i}$ and $P_i \subseteq C_a^u(i)$, there are only two possibilities for the values of $x_{V(a)\setminus\{i\}}$:

(a) either there is one satisfying variable and everything else is unsatisfying, or

(b) there are at least two variables that are satisfying or $*$.

We first calculate the weight $W(A)$ assigned to possibility (a), again using the BP update equation (2.2):

$$\begin{aligned} W(A) &= \sum_{k\in V(a)\setminus\{i\}} \sum_{S^k\subseteq C_a^s(k)} M_{k\to a}(s_{a,k}, S^k \cup \{a\}) \prod_{j\in V(a)\setminus\{i,k\}} \sum_{S^j\subseteq C_a^u(j)} M_{j\to a}(u_{j,a}, S^j) \\ &= \sum_{k\in V(a)\setminus\{i\}} R^s_{k\to a} \prod_{j\in V(a)\setminus\{i,k\}} R^u_{j\to a}. \end{aligned}$$

We now calculate the weight $W(B)$ assigned to possibility (b) in the following way. From our calculations in part (ii), we found that the weight assigned to the event that each variable is either unsatisfying, satisfying or $*$ is $\prod_{j\in V(a)\setminus\{i\}} \left[R^u_{j\to a} + R^*_{j\to a}\right]$. The weight $W(B)$ is given by subtracting from this quantity the weight assigned to the event that there are *not* at least two $*$ or satisfying assignments. This event can be decomposed into the disjoint events that either all assignments are unsatisfying (with weight $\prod_{j\in V(a)\setminus\{i\}} R^u_{j\to a}$ from part (ii)); or that exactly one variable is $*$ or satisfying. The weight corresponding to this second possibility is

$$\begin{aligned} &\sum_{k\in V(a)\setminus\{i\}} \left[M_{k\to a}(*, \emptyset) + \sum_{S^k\subseteq C_a^s(k)} M_{k\to a}(s_{k,a}, S^k)\right] \prod_{j\in V(a)\setminus\{i,k\}} \sum_{S^j\subseteq C_j^u(a)} M_{j\to a}(u_{j,a}, S^j) \\ &\qquad = \sum_{k\in V(a)\setminus\{i\}} R^*_{k\to a} \prod_{j\in V(a)\setminus\{i,k\}} R^u_{j\to a}. \end{aligned}$$

Combining our calculations so far we have

$$W(B) = \prod_{j\in V(a)\setminus\{i\}} \left[R^u_{j\to a} + R^*_{j\to a}\right] - \sum_{k\in V(a)\setminus\{i\}} R^*_{k\to a} \prod_{j\in V(a)\setminus\{i,k\}} R^u_{j\to a} - \prod_{j\in V(a)\setminus\{i\}} R^u_{j\to a}.$$

Finally, summing together the forms of $W(A)$ and $W(B)$ from and then factoring yields the desired equation (3.12b). □

Since the messages are interpreted as probabilities, we only need their ratio, and we can normalize them to any constant. At any iteration, approximations to the local marginal probabilities at each variable node $i \in V$ are given by (up to a normalization constant):

$$\begin{aligned}
F_i(0) &\propto \prod_{b\in C^+(i)} M^u_{b\to i}\Big[\prod_{b\in C^-(i)}(M^s_{b\to i}+M^*_{b\to i})-(1-\omega_o)\prod_{b\in C^-(i)} M^*_{b\to i}\Big]\\
F_i(1) &\propto \prod_{b\in C^-(i)} M^u_{b\to i}\Big[\prod_{b\in C^+(i)}(M^s_{b\to i}+M^*_{b\to i})-(1-\omega_o)\prod_{b\in C^+(i)} M^*_{b\to i}\Big]\\
F_i(*) &\propto \omega_*\prod_{b\in C(i)} M^*_{b\to i}
\end{aligned}$$

The following theorem establishes that the $\mathrm{SP}(\rho)$ family of algorithms is equivalent to belief propagation on the extended MRF:

Theorem 5. *For all $\omega_* \in [0,1]$, the BP updates on the extended (ω_*, ω_o)-MRF p_{gen} are equivalent to the $\mathrm{SP}(\omega_*)$ family of algorithms under the following restrictions:*

(a) the constraint $\omega_o + \omega_* = 1$ is imposed, and

(b) all messages are initialized such that $M^u_{a\to i} = M^*_{a\to i}$ for every edge (a,i).

Proof. Under the constraint $\omega_o + \omega_* = 1$, if we initialize $M^u_{a\to i} = M^*_{a\to i}$ on every edge, then there holds $R^s_{i\to a} = R^*_{i\to a}$ and consequently $M^u_{a\to i} = M^*_{a\to i}$ remains true at the next iteration. Initializing the parameters in this way and imposing the normalization $M^u_{a\to i} + M^*_{a\to i} = 1$ leads to the following recurrence equations:

$$M^s_{a\to i} = \frac{\prod_{j\in V(a)\setminus\{i\}} R^u_{j\to a}}{\prod_{j\in V(a)\setminus\{i\}}(R^*_{j\to a}+R^u_{j\to a})},$$

where

$$\begin{aligned}
R^u_{i\to a} &= \prod_{b\in C^s_a(i)}(1-M^s_{b\to i})\Big[1-\omega_*\prod_{b\in C^u_a(i)}(1-M^s_{b\to i})\Big],\\
R^*_{i\to a} &= \prod_{b\in C^u_a(i)}(1-M^s_{b\to i}).
\end{aligned}$$

These updates are equivalent to $\mathrm{SP}(\omega_*)$ by setting $\eta_{a\to i} = M^s_{a\to i}$, $\Pi^u_{i\to a} = R^u_{i\to a}$, and $\Pi^s_{i\to a} + \Pi^*_{i\to a} = R^*_{i\to a}$. □

Remarks:

1. As mentioned earlier, setting $\omega_o = 1$ and $\omega_* = 0$ amounts to forbidding any $*$ symbols, so that the only partial assignments with positive probability are ordinary (i.e., $\{0,1\}$) satisfying

assignments. Moreover, the extended MRF in (3.6) assigns all satisfying assignments the same probability, since both constrained and unconstrained variables are given weight one. Thus, the extended MRF reduces to the standard MRF in (2.5), and the $\mathrm{SP}(0)$ algorithm reduces to the naive belief propagation algorithm described in section 2.2. The other extreme entails setting $\omega_o = 0$ and $\omega_* = 1$, which gives rise to the pure form $\mathrm{SP}(1)$ of survey propagation. For this setting of the parameters, unconstrained variables are not allowed, and again all partial assignments with positive probability have the same weight. Braunstein and Zecchina [BZ04] consider this special case of our extended MRF, though from a rather different perspective.

2. Given the link between SP and extended MRFs, it is natural to study combinatorial and probabilistic properties of the latter objects. In Section 3.3, we show how so-called "cores" arise as fixed points of $\mathrm{SP}(1)$, and we prove a weight-preserving identity that shows how the extended MRF for general ρ is a "smoothed" version of the naive MRF.

3. The initial messages have very small influence on the behavior of the algorithm, and they are typically chosen to be uniform random variables in $(0, 1)$. In practice, for $\omega_o + \omega_* = 1$ if we start with different values for $M^u_{a\to i}$ and $M^*_{a\to i}$ they soon converge to become equal.

4. The result allows us to generalize the algorithm to other Boolean CSP.

5. If we restrict our attention to 3-SAT, the equations have simpler form. In particular for a clause a on x_i, x_j, x_k, the messages to variable node i are:

$$\begin{aligned}
M^s_{a\to i} &= R^u_{j\to a} R^u_{k\to a} \\
M^u_{a\to i} &= R^*_{j\to a} R^*_{k\to a} + R^s_{j\to a} R^u_{k\to a} + R^u_{j\to a} R^s_{k\to a} \\
M^*_{a\to i} &= R^*_{j\to a} R^*_{k\to a} + R^*_{j\to a} R^u_{k\to a} + R^u_{j\to a} R^*_{k\to a}.
\end{aligned}$$

3.3 Interpretation of survey propagation

In this section we relate the properties of the family of Markov random fields defined in the previous section, to the assumptions underlying the design of survey propagation. First, we introduce a partial order on the partial assignments and demonstrate the connection between minimal elements in this partial order, clusters of satisfying assignments, and fixed points of the survey propagation algorithm. The definition of this partial order and some of its properties are also essential for the next two chapters. In Section 3.3.2 we summarize the properties of the space of partial assignments in the easy case of low density formulas, where by low density we mean density for which the pure-literal heuristic is successful with high probability. In Section 3.3.3 we show how varying the parameters of our family of Markov random fields results in "smoothing" of the distribution over satisfying assignments, which may be the reason behind the successful convergence of the belief propagation heuristic for some of the parameters. Finally, we discuss experimental observations for 3-SAT, and some recent related work for k-SAT with $k \geq 8$.

3.3.1 Partial assignments and clustering

We begin by defining an acyclic directed graph on all valid partial assignments. The vertex set of the directed graph G consists of all valid partial assignments. The edge set is defined in the following way: for a given pair of valid partial assignments $\mathbf{u}$ and $\mathbf{v}$, the graph includes a directed edge from $\mathbf{u}$ to $\mathbf{v}$ if there exists an index $i \in V$ such that (i) $u_j = v_j$ for all $j \neq i$; and (ii) $v_i = *$ and $u_i \neq *$. We label the edge between $\mathbf{u}$ and $\mathbf{v}$ with the index i, corresponding to the fact that $\mathbf{v}$ is obtained from $\mathbf{u}$ by adding one extra $*$ in position i.

This directed graph G has a number of properties:

(a) Valid partial assignments can be separated into different levels based on their number of star variables. In particular, assignment $\mathbf{u}$ is in level $n_*(\mathbf{u})$. Thus, every edge is from an assignment in level $l-1$ to one in level l, where l is at most n.

(b) The out-degree of any valid partial assignment $\mathbf{u}$ is exactly equal to its number of unconstrained variables $n_o(\mathbf{u})$, and the outgoing edges are labeled by the set $S_o(\mathbf{u})$.

(c) It is an acyclic graph so that its structure defines a partial ordering; in particular, we write $\mathbf{v} < \mathbf{u}$ if there is a directed path in G from $\mathbf{u}$ to $\mathbf{v}$. Notice that all directed paths from $\mathbf{u}$ to $\mathbf{v}$ are labeled by indices in the set $T = \{i \in V : u_i \neq v_i = *\}$, and only the order in which they appear is different.

The minimal assignments in this ordering are precisely all the positive probability assignments in the distribution p_W for $(\omega_o, \omega_*) = (0, 1)$. Theorem 5 leads to viewing the pure form of survey propagation $\mathrm{SP}(1)$ as performing an approximate marginalization over such assignments. Before we explore their properties we need several definitions.

For any valid partial assignment $\mathbf{u}$ and a subset $S \subseteq V$, let $\gamma_S(\mathbf{u})$ be the minimal $\mathbf{v} < \mathbf{u}$, such that the path from $\mathbf{u}$ to $\mathbf{v}$ is labeled only by indices in S. For $S = V$ the assignment $\gamma_V(\mathbf{u})$ is minimal in the ordering.

Definition 4. The *core* of a valid partial assignment $\mathbf{u}$ is the assignment $\gamma_V(\mathbf{u})$.

The following proposition asserts that there always exists a unique $\gamma_S(\mathbf{u})$.

Proposition 6. *For any valid assignment* $\mathbf{u}$ *and* $S \subseteq V$*, there is a unique minimal* $\mathbf{v} < \mathbf{u}$ *such that the path from* $\mathbf{u}$ *to* $\mathbf{v}$ *is labeled only by indices in* S*.*

Proof. To establish the uniqueness statement, suppose that there are two minimal such assignments $\mathbf{v^1}$ and $\mathbf{v^2}$, and the paths from $\mathbf{u}$ to $\mathbf{v^1}$ and $\mathbf{v^2}$ are labeled by sets of indices $T_1, T_2 \subseteq S$ respectively. If $T_1 = T_2$ then $\mathbf{v^1} = \mathbf{v^2}$, so let us assume that T_1 and T_2 are distinct. Without loss of generality, we may take $T_1 \backslash T_2 \neq \emptyset$. Consider a particular path from $\mathbf{u}$ to $\mathbf{v^1}$, with labels $t_1, t_2, \ldots t_r$, where $r = |T_1|$. Let t_i be the first label such that $t_i \notin T_2$. Then its corresponding variable is unconstrained when the variables indexed by $\{t_1, \ldots t_{i-1}\} \cup S_*(\mathbf{u}) \subseteq T_2 \cup S_*(\mathbf{u})$ are assigned $*$, therefore it is unconstrained in $\mathbf{v^2}$. This implies that there exists an edge out of $\mathbf{v^2}$ that is labeled by $t_i \in S$, which contradicts the assumption that $\mathbf{v^2}$ is minimal. ☐

Next, we show that satisfying assignments that belong to the same cluster of solutions have the same core.

Proposition 7. *If satisfying assignments* $\mathbf{a}, \mathbf{b} \in \{0,1\}^n$ *belong to the same cluster of solutions then their cores are the same.*

Proof. It suffices to prove the statement for $\mathbf{a}$ and $\mathbf{b}$ that differ in a single variable x_i. Since x_i can take both 0 and 1 value it is unconstrained in both $\mathbf{a}$ and $\mathbf{b}$. Therefore there is a path from $\mathbf{a}$ to $\gamma_V(\mathbf{a})$ that starts with an edge labeled by i, and the same is true for $\mathbf{b}$. These paths converge after the first edge, therefore $\gamma_V(\mathbf{a}) = \gamma_V(\mathbf{b})$. ☐

Definition 5. A *cover* is a valid partial assignment $\mathbf{v} \in \{0,1,*\}^n$ that contains no unconstrained variables.

In particular the core of any satisfying assignment is a cover. On the other hand not all cover assignments are cores.

Given a satisfying assignment $\mathbf{a}$ it is easy to find the core corresponding to its cluster simply by going down the partial order - that is by replacing unconstrained variables by $*$. We refer to this process as *coarsening*. The following result connects fixed points of $\mathrm{SP}(1)$ to core assignments. In particular $\mathrm{SP}(1)$, when suitably initialized, simply strips the valid assignment $\mathbf{a}$ down to its core $\gamma_V(\mathbf{a})$, as in the coarsening process.

Proposition 8. *For a valid assignment* $\mathbf{w}$*, let* $\mathrm{SP}(1)$ *be initialized by:*

$$\Pi^u_{i\to a} = \delta(w_i, u_{a,i}), \quad \Pi^s_{i\to a} = \delta(w_i, s_{a,i}), \quad \Pi^*_{i\to a} = 0.$$

Then within a finite number of steps, the algorithm converges and the output fields are

$$\mu_i(b) = \delta(v_i, b),$$

where $\mathbf{v} = \gamma_V(\mathbf{w})$ *and* $b \in \{0, 1, *\}$.

Proof. We say that a variable i belongs to the core if $v_i \neq *$. We say that a clause a belongs to the core if all the variables in the clause belong to the core. We first show by induction that

I. If a and i belong to the core and x_i is not the unique satisfying variable for a in assignment $\mathbf{w}$ then $\Pi^u_{i\to a} = \delta(w_i, u_{a,i})$ and $\Pi^s_{i\to a} = \delta(w_i, s_{a,i})$, and

II. If a and i belong to the core and x_i is unique satisfying variable for a then $\eta_{a\to i} = 1$.

Clearly, property I holds at time 0. Therefore, it suffices to prove that if property I holds at time t then so does II. and that if property II holds at time t then property I holds at time $t+1$.

Suppose that property I holds at time t. Let a and i belong to the core such that x_i is the unique satisfying variable of the clause a. By the induction hypothesis for all $j \in V(a) \setminus \{i\}$ it holds that $\Pi^u_{j\to a} = \delta(w_j, u_{a,j}) = 1$. This implies that $\eta_{a\to i} = 1$ as needed.

Suppose that property II holds at time t. Let a and i belong to the core such that x_i is not unique satisfying for a. By the assumption, it follows that there exists b which belongs to the core such that x_i is the unique satisfying variable for b. This implies by the induction hypothesis that $\eta_{b\to i} = 1$. It is now easy to see that at update $t+1$: $\Pi^u_{i\to a} = \delta(w_i, u_{a,i})$ and $\Pi^s_{i\to a} = \delta(w_i, s_{a,i})$. Note that the claim above implies that for all times t and all i such that $v_i \neq *$, it holds that $\mu_i(b) = \delta(v_i, b)$.

Let $i_1, i_2, \ldots, i_s$ be a "coarsening-path" from $\mathbf{w}$ to $\mathbf{v}$. In other words, the variable x_{i_1} is not uniquely satisfying any clause. Once, this variable is set to $*$, the variable x_{i_2} is not uniquely satisfying any clause etc. We claim that for all $1 \leq t \leq s$, for all updates after time t and for all clauses a such that $i_t \in V(a)$ it holds that $\eta_{a\to i_t} = 0$. The proof follows easily by induction on t. This in turn implies that if for all updates after time t $\mu_{i_t}(b) = \delta(v_i, *)$, from which the result follows. ☐

3.3.2 Connectivity of the space of solutions of low-density formulas

The pure-literal (PL) heuristic [RPF99] is the following algorithm: assign 1 to a variable if it only appears positively in a clause, or 0 if it only appears negatively in a clause, reduce the formula, and repeat the procedure.

It is known that there is a phase transition for the event that the pure-literal heuristic succeeds [RPF99]. That is, for every $k \geq 2$ there exists a constant α_{PL} such that for any $\epsilon > 0$:

$$\begin{aligned} \Pr[\text{ PL heuristic succeeds for a random } k\text{-CNF formula of density } \alpha_{\mathrm{PL}} - \epsilon] &\rightarrow 1 \\ \Pr[\text{ PL heuristic succeeds for a random } k\text{-CNF formula of density } \alpha_{\mathrm{PL}} + \epsilon] &\rightarrow 0 \end{aligned}$$

For 2-SAT this transitions is at $\alpha_{\mathrm{PL}} = 1 = \alpha_c$, i.e. it coincides with the satisfiability threshold. For 3-SAT the transition for the PL heuristic is at 1.63. The exact value is computable for any fixed k, as well as asymptotically [BFU93, LMS98, Mol04]. In this section we consider random formulas of density below α_{PL}, or more generally, formulas for which the pure-literal heuristic succeeds.

Proposition 9. *If a k-CNF formula for $k \geq 2$ is such that the pure-literal heuristic successfully finds a solution, then:*

1. *the solution space consists of a single connected component,*
2. *the only cover is $(*^n)$, and*
3. *([ABS03]) the standard random walk algorithm introduced in [Pap91] finds a solution in polynomial time.*

Proof. Suppose the solution space is disconnected. We choose two satisfying assignments $\mathbf{a}$ and $\mathbf{b}$ that belong to different components, and differ in a minimal number of variables. Let $D = \{i : a_i \neq b_i\}$. The pure-literal heuristic assigns variables one at a time. Let x_j be the first variable from D that is assigned by the pure-literal heuristic. The formula before x_j is assigned, is such that x_j appears either only positive or only negative. Without loss of generality we can assume that it does not appear negated, and $a_j = 0$, $b_j = 1$. This implies that if we flip the value of x_j in the assignment $\mathbf{a}$, then the new assignment will still be satisfying; at the same time it is in the same connected component as $\mathbf{a}$, and it is closer to $\mathbf{b}$ than $\mathbf{a}$ is. This contradicts the way we chose $\mathbf{a}$ and $\mathbf{b}$.

Suppose there is a non-trivial cover $\sigma \in \{0, 1, *\}^n$. Without loss of generality let $\sigma = (1^r *^{n-r})$ for some $0 < r \leq n$. The pure-literal heuristic assigns the variables in some order, and again without loss of generality we can assume that $x_1, \ldots, x_r$ are assigned in that order. Since x_1 appears positive in at least one clause containing only variables from $x_1, \ldots, x_r$, it has to be assigned 1. Similarly x_2 appears positive in a constraining clause, and furthermore x_1 couldn't be one of the negated variables in that clause, because it was pure when it was assigned by PL. Therefore x_2 is also assigned 1. In general x_i appears positive in a constraining clause, which contains neither of $x_1, \ldots, x_{i-1}$. For $i = r$ this leads to a contradiction, since this clause has to be a unit clause. (If we allow unit clauses, the statement of the theorem does not hold.) □

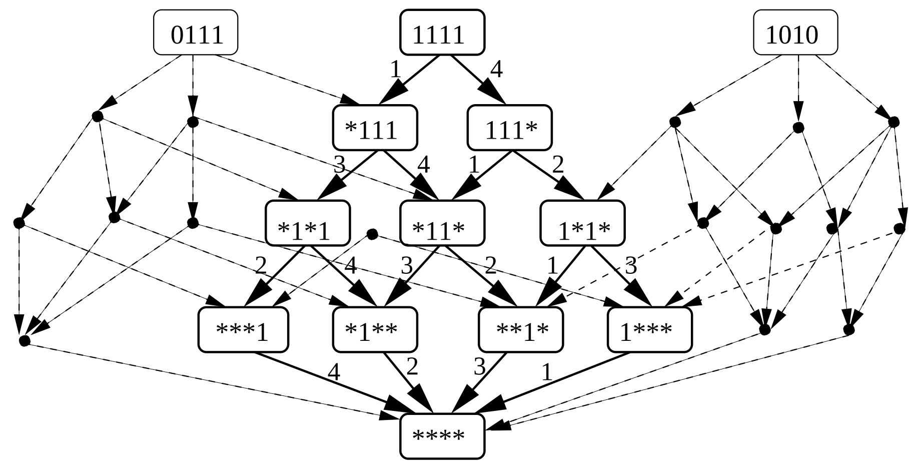
3.3.3 Role of the parameters of the Markov random field

For any satisfying assignment $\mathbf{a} \in \{0, 1\}^n$, the assignments below it in the partial order contribute their weight in the distribution towards assigning the variables in agreement with $\mathbf{a}$. The subgraph of these partial assignments can have different properties, as depicted in Figure 3.3. Only a subset of the partial assignments is shown, since even for small formulas the space of partial assignments is quite large. For the first formula all satisfying assignments have a trivial core. For the second one, on the other hand, there are assignments with non-trivial cores.

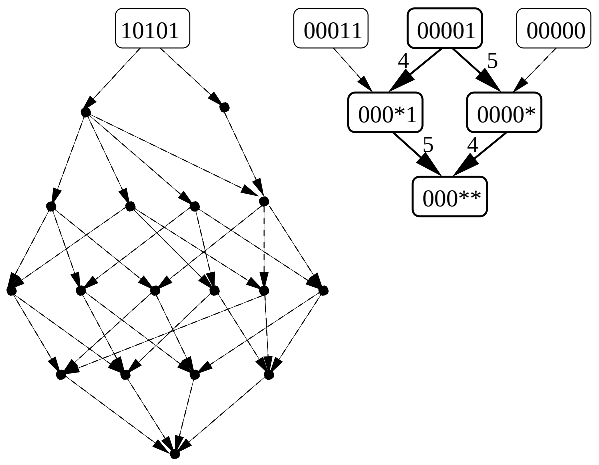
One of the benefits of our analysis is that it suggests a large pool of algorithms to be investigated, for example by varying the values of ω_o and ω_*. A "good" setting of these parameters should place significant weight on precisely those valid assignments that can be extended to satisfying assignments. At the same time, the parameter setting clearly affects the level of connectivity in the space of valid assignments. Connectivity most likely affects the performance of belief propagation, as well as any other algorithm that we may apply to compute marginal probabilities or sample from the distribution.

Figure 3.4(a) shows the performance of belief propagation on the extended MRF for different values of (ω_o, ω_*), and applied to particular random formula with $n = 10,000$, $k = 3$ and $\alpha = 4.2$. The most successful pairs in this case were $(0.05, 0.95)$, $(0.05, 0.9)$, $(0.05, 0.85)$, and $(0.05, 0.8)$. For these settings of the parameters the decimation steps reached a solution, so a call to Walk-SAT was not needed. For weights satisfying $\omega_o + \omega_* > 1$, the behavior is very predictable: even when the algorithm converges, the choices that it makes in the decimation steps lead to a contradiction. Note that there is a sharp transition in algorithm behavior as the weights cross the line $\omega_o + \omega_* = 1$, which is representative of the more general behavior.

The following result provides some justification for the excellent performance in the regime $\omega_o + \omega_* \leq 1$.



(a)



(b)

Figure 3.3. Portion of the directed graph on partial assignments for two different formulas: (a) $(\bar{x}_1 \vee \bar{x}_2 \vee x_3) \wedge (x_2 \vee \bar{x}_3 \vee \bar{x}_4)$. Highlighted is the lattice below the satisfying assignment $z = (1, 1, 1, 1)$, whose core is trivial (i.e., $\gamma_V(z) = (*, *, *, *)$). (b) $(\bar{x}_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2 \vee x_3) \wedge (x_2 \vee \bar{x}_3 \vee x_1) \wedge (x_2 \vee \bar{x}_3 \vee x_5) \wedge (x_1 \vee x_5 \vee \bar{x}_4)$. The satisfying assignment $z = (0, 0, 0, 0, 1)$ has the non-trivial core $\gamma_V(z) = (0, 0, 0, *, *)$. For the same formula there are other satisfying assignments, e.g. $(1, 0, 1, 0, 1)$ which have a trivial core.

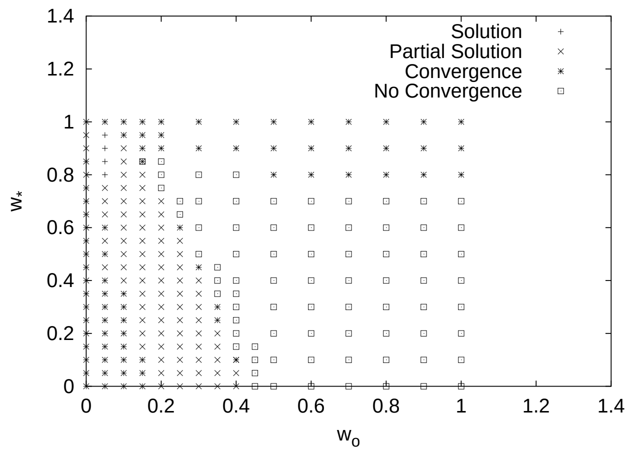


Figure 3.4. Performance of BP for different choices of (ω_o, ω_*) as applied to a particular randomly chosen formula with $n = 10000$, $k = 3$, $\alpha = 4.2$. Four distinct cases can be distinguished: (i) BP converges and the decimation steps yields a complete solution, (ii) BP converges and the decimation steps yield a partial solution, completed by using Walk-SAT, (iii) BP converges, but the decimation steps do not lead to a solution, and (iv) BP does not converge.

Theorem 10. (Weight Preservation Theorem) *If* $\omega_o + \omega_* = 1$*, then* $\sum_{y \leq x} W(y) = \omega_*^{n_*(x)}$ *for any valid assignment* x*. If* $\omega_o + \omega_* < 1$*, then* $\sum_{y \leq x} W(y) \geq (\omega_*)^{n_*(x)}$ *for any valid assignment* x*.*

We defer the proof of this theorem to the next chapter.

The Weight Preservation Theorem has a very natural interpretation in terms of a “smoothing” operation. In particular, the (ω_o, ω_*)-MRF may be regarded as a smoothed version of the uniform distribution over satisfying assignments, in which the uniform weight assigned to each satisfying assignment is spread over the lattice associated with it. Note, however, that any partial assignment that belongs to two or more lattices is assigned a weight only once. Otherwise, the transformation would be a convolution operation in a strict sense.

3.3.4 Coarsening experiments for 3-SAT

We have performed a large number of the following experiments:

1. starting with a satisfying assignment $\mathbf{u}$, change a random one of its unconstrained variables to $*$,

2. repeat until there are no unconstrained variables.

This coarsening procedure, is equivalent to taking a random path from $\mathbf{u}$ in G, by choosing at each step a random outgoing edge. Any such path terminates at the core $\gamma_V(\mathbf{u})$. It is interesting

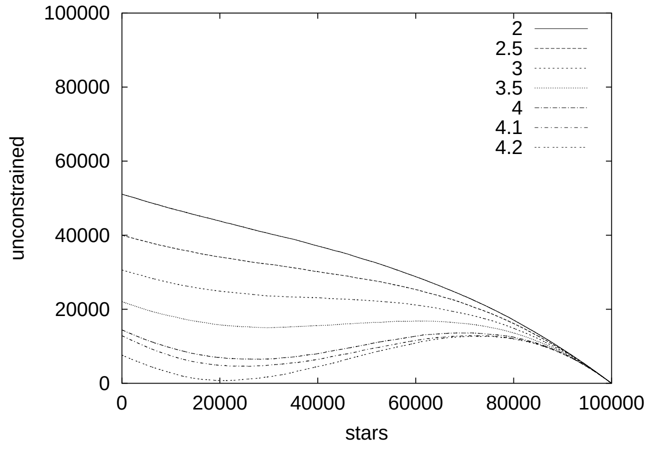


Figure 3.5. Evolution of the number of unconstrained variables in the coarsening process: start with a satisfying assignment, change a random unconstrained variable to $*$ and repeat. Plotted is the result of an experiment for $n = 100,000$, for random formulas with $k = 3$ and $\alpha = \{2, 2.5, 3, 3.5, 4, 4.1, 4.2\}$. In particular, core assignments are on the x-axis, and satisfying assignments are on the y-axis.

to examine at each step of this process the number of unconstrained variables (equivalently, the number of outgoing edges in the graph G). For 3-SAT problems, Figure 3.5 shows the results of such experiments for $n = 100,000$, and using different values of α. The plotted curves are the evolution of the number of unconstrained variables as the number of $*$'s increases. We note that for $n = 100$ and α close to the threshold, satisfying assignments often correspond to core assignments; a similar observation was also made by Braunstein and Zecchina [BZ04]. In contrast, for larger n, this correspondence is rarely the case. Rather, the generated curves suggest that $\gamma_V(\mathbf{u})$ is almost always the all-$*$ assignment, and moreover that for high density α, there is a critical level in G where the out-degrees are very low. Increasing α results in failure of the algorithm itself, rather than in the formation of real core assignments.

The fact that we do not observe core assignments for $k = 3$, and yet the algorithm is successful, means that an alternative explanation is required. Accordingly, we propose studying the behavior of $\mathrm{SP}(\rho)$ for $\rho \in (0, 1)$. Our experimental results, consistent with similar reports from Kirkpatrick [Kir04], show that $\mathrm{SP}(\rho)$ tends to be most effective in solving k-SAT for values of $\rho < 1$. If so, the good behavior of $\mathrm{SP}(1)$ may well follow from the similarity of $\mathrm{SP}(1)$ updates to $\mathrm{SP}(\rho)$ updates for $\rho \approx 1$.

3.3.5 Related work for k-SAT with $k \geq 8$

Interestingly, as mentioned earlier, for $k \geq 9$ there are values for $\alpha < \alpha_c$ such that this coarsening procedure provably results in a non-trivial core assignment with high probability, as was proved by Achlioptas and Ricci-Tersenghi in [ART06]. Furthermore, for $k \geq 8$ they prove that there are an exponential number of components of solutions, such that assignments in different components differ in a constant fraction of the variables. The existence of such components was proved also by Mora, Mezard and Zecchina [MMZ05].

3.4 Future directions

As we described, associated with any k-SAT problem is a large family of Markov random fields over partial assignments, as specified by the parameter ρ (or more generally, the parameters ω_o and ω_*). Further analysis of survey propagation and its generalizations requires a deeper understanding of the following two questions. First, for what parameter choices do the marginal probabilities of the associated Markov random field *yield useful information* about the structure of satisfiability assignments? Second, for what parameter choices do efficient message-passing algorithms like belief propagation *yield accurate approximations* to these marginal probabilities? Our results show that the success of SP-like algorithms depends on a delicate balance between these two factors. (For instance, the marginal probabilities of the uniform distribution over SAT assignments clearly contain useful information, but belief propagation fails to yield good approximations for sufficiently large clause densities.) More generally, these questions fall in a broader collection of issues, all related to a deeper understanding of satisfiability problems and especially the relationship between finite satisfiability problems and their asymptotic analysis.

Chapter 4

Towards bounding the satisfiability threshold of 3-SAT

All of the known rigorous upper bounds for the satisfiability threshold of 3-SAT are based on the first moment method [KMPS94, DB97, KKKS98, JSV00, KKS$^+$01, DBM03]. The corresponding upper bounds in this sequence of results are: 4.758, 4.64, 4.601, 4.596, 4.571, 4.506. The general method is to consider a random variable X equal to the number of satisfying assignments of a particular kind. These satisfying assignments are such that at least one exists if the formula is satisfiable. Showing that above a certain density in the random 3-SAT model $\mathrm{E}[X] \to 0$ implies that $\Pr[X = 0]$ goes to 1 and consequently the probability that the formula is unsatisfiable also goes to 1. For example [DB97] takes X to be the number of *negatively prime solutions* - these are solutions for which every variable assigned 1 is constrained.

The Markov random fields defined in Chapter 3 provide a richer class of choices for X that can be explored. In particular, the Weight Preservation Theorem (Theorem 10) implies that we can choose X to be the total weight of partial assignments, or the total weight of partial assignments that can be reached from a satisfying assignment, or the total weight of partial assignments that can be reached from positively prime solutions, etc. The intuition is that this idea may allow us to avoid the overcouning due to formulas with a lot of solutions in a single cluster, by merging a large part of the weight of such solutions.

Although we are not able to improve the current best upper bound for the threshold (which is 4.51), we obtain the following partial result:

Theorem 11. *For random instances of 3-SAT with density greater than* 4.46*, with high probability* *there exists* no *satisfying assignment that has a non-trivial core.*

In light of the results of [ART06], proving that assignments with trivial cores do not exist with high probability appears to be a considerably easier task, and a promising direction for future work.

In the next section we prove the Weight Preservation Theorem which is necessary for the proof of Theorem 11. A more detailed discussion of the properties of lattices such as the ones formed by the partial assignments is given in section 4.2. Section 4.3 is devoted to the proof of Theorem 11.

4.1 Weight preservation theorem

Recall that the statement of the theorem is the following:

Weight Preservation Theorem: *If $\omega_o + \omega_* = 1$, then $\sum_{\mathbf{y}\leq\mathbf{x}} W(\mathbf{y}) = \omega_*^{n_*(\mathbf{x})}$ for any valid assignment $\mathbf{x}$. If $\omega_o + \omega_* < 1$, then $\sum_{\mathbf{y}\leq\mathbf{x}} W(\mathbf{y}) \geq (\omega_*)^{n_*(\mathbf{x})}$ for any valid assignment $\mathbf{x}$.*

Proof. We start with the case $\omega_o + \omega_* = 1$. Let A denote the set of partial assignments $\mathbf{z}$ such that $z_j \in \{x_j, *\}$ for all $j \in V$. We refer to these as the set of assignments consistent with $\mathbf{x}$. Let $B = \{\mathbf{y} : \mathbf{y} \leq \mathbf{x}\}$ be the set of valid assignments that are reachable from $\mathbf{x}$. Notice that all $\mathbf{y} \in B$ are valid and consistent with $\mathbf{x}$, but not every valid assignment in A is reachable from $\mathbf{x}$. We will let $S_*(\mathbf{z})$ denote the set of variables assigned $*$ both for valid and invalid assignments $\mathbf{z}$.

We define a map between all assignments consistent with $\mathbf{x}$ and the set of reachable ones. Let $\sigma : A \to B$ be defined as

$$\sigma(\mathbf{z}) := \gamma_{S_*(\mathbf{z})}(\mathbf{x}).$$

Notice that if $\mathbf{y} \in B$ then $\sigma(\mathbf{y}) = \mathbf{y}$. The map is, of course, many-to-one. We define what we later show is the reverse map. For $\mathbf{y} \in B$ let

$$\tau(\mathbf{y}) := \{\mathbf{z} \in A : S_*(\mathbf{z}) = S_*(\mathbf{y}) \cup T, T \subseteq S_c(\mathbf{y})\}.$$

Lemma 12. *For any $\mathbf{y} \in B$ and $\mathbf{z} \in A$, $\mathbf{z} \in \tau(\mathbf{y})$ if and only if $\sigma(\mathbf{z}) = \mathbf{y}$.*

Proof. Let $\mathbf{z} \in \tau(\mathbf{y})$ so that $S_*(\mathbf{z}) = S_*(\mathbf{y}) \cup T$ for some $T \subseteq S_c(\mathbf{y})$. $\sigma(\mathbf{z}) = \gamma_{S_*(\mathbf{z})}(\mathbf{x})$ is the minimal valid assignment such that the path from $\mathbf{x}$ to it is labeled only by elements in $S_*(\mathbf{z})$. We will show that $\mathbf{y}$ satisfies these properties, and therefore by proposition 6, $\mathbf{y} = \sigma(\mathbf{z})$. Any path from $\mathbf{x}$ to $\mathbf{y}$ (which exists since $\mathbf{y} \in B$) is labeled by $S_*(\mathbf{y})\backslash S_*(\mathbf{x}) \subseteq S_*(\mathbf{z})$. Furthermore, for

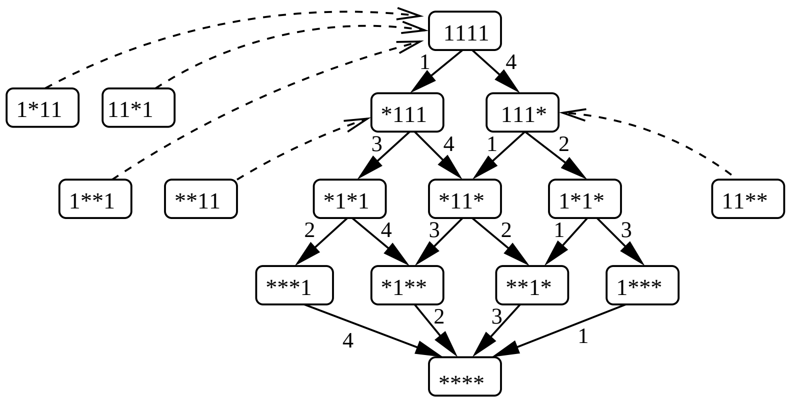


Figure 4.1. The directed graph G and the map σ for the formula $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x_2} \vee \bar{x_3} \vee x_4)$ and the satisfying assignment $(0, 0, 1, 0)$. The solid arrows denote edges in G and the dashed arrows denote σ.

every $i \in S_*(\mathbf{z})$, $i \notin S_o(\mathbf{y})$ so there is no outgoing edge from $\mathbf{y}$ labeled by an element in $S_*(\mathbf{z})$. Therefore $\mathbf{y}$ is minimal.

Let $\mathbf{y} = \sigma(\mathbf{z}) = \gamma_{S_*(\mathbf{z})}(\mathbf{x})$. By proposition 6 there is no $i \in S_*(\mathbf{z})$ such that $i \in S_o(\mathbf{y})$. Therefore $S_*(\mathbf{z}) \subseteq S_*(\mathbf{y}) \cup S_c(\mathbf{y})$. Further we have that $S_*(\mathbf{y}) \subseteq S_*(\mathbf{z}) \cup S_*(\mathbf{x}) = S_*(\mathbf{z})$, therefore $S_*(\mathbf{z}) = S_*(\mathbf{y}) \cup T$ for some $T \subseteq S_c(\mathbf{y})$. Hence $\mathbf{z} \in \tau(\mathbf{y})$. ☐

For a set of partial assignments X let $W(X) = \sum_{\mathbf{x} \in X} W(\mathbf{x})$. Let $W^{\emptyset}(\mathbf{z}) = \omega_*^{n_*(\mathbf{z})} \times \omega_o^{n - n_*(\mathbf{z})}$, denote the weight of any partial assignment, if the formula had no clauses. For such a formula all partial assignments are valid. Observe that if we restrict our attention to the assignments that are consistent with $\mathbf{x}$,

$$\begin{aligned} W^{\emptyset}(A) &= \sum_{\mathbf{z} \in A} W^{\emptyset}(\mathbf{z}) \\ &= \sum_{S \subseteq V \setminus S_*(\mathbf{x})} \omega_*^{|S_*(\mathbf{x})| + |S|} \times \omega_o^{n - |S_*(\mathbf{x})| - |S|} \\ &= \omega_*^{|S_*(\mathbf{x})|} \times (\omega_* + \omega_o)^{n - |S_*(\mathbf{x})|} \\ &= \omega_*^{n_*(\mathbf{x})} \end{aligned}$$

We show that when clauses are added to the formula, the total weight under $\mathbf{x}$ is preserved as long as $\mathbf{x}$ is still valid. In particular when an assignment $\mathbf{z}$ that is consistent with $\mathbf{x}$ becomes invalid, it passes its weight to an assignment that is still valid, namely $\sigma(\mathbf{z})$, which has fewer $*$

variables than $\mathbf{z}$.

$$\begin{aligned}
W(\mathbf{y}) &= \omega_*^{n_*(\mathbf{y})} \times \omega_o^{n_o(\mathbf{y})} \times 1^{n_c(\mathbf{y})} \\
&= \omega_*^{n_*(\mathbf{y})} \times \omega_o^{n_o(\mathbf{y})} \times (\omega_* + \omega_o)^{n_c(\mathbf{y})} \qquad (4.1) \\
&= \sum_{T \subseteq S_c(\mathbf{y})} \omega_*^{n_*(\mathbf{y})+|T|} \times \omega_o^{n_o(\mathbf{y})+n_c(\mathbf{y})-|T|} \\
&= \sum_{T \subseteq S_c(\mathbf{y})} W^{\emptyset}(\mathbf{z} : S_*(\mathbf{z}) = S_*(\mathbf{y}) \cup T) \\
&= W^{\emptyset}(\{\mathbf{z} : S_*(\mathbf{z}) = S_*(\mathbf{y}) \cup T, T \subseteq S_c(\mathbf{y})\}) \\
&= W^{\emptyset}(\tau(\mathbf{y})).
\end{aligned}$$

Finally, we have:

$$\sum_{\mathbf{y} \leq \mathbf{x}} W(\mathbf{y}) = \sum_{\mathbf{y} \leq \mathbf{x}} W^{\emptyset}(\tau(\mathbf{y})) = W^{\emptyset}(A) = \omega_*^{n_*(\mathbf{x})}$$

where we used the fact that the sets $\tau(\mathbf{y})$ for $\mathbf{y} \in B$ partition A by lemma 12.

The proof of the case $\omega_o + \omega_* < 1$ is similar except that equation (4.1) becomes an inequality:

$$W(\mathbf{y}) = \omega_o^{n_o(\mathbf{y})} \times \omega_*^{n_*(\mathbf{y})} \times 1^{n_c(\mathbf{y})} \geq \sum_{T \subseteq S_c(S)} W^{\emptyset}(\tau(\mathbf{y})).$$

When an assignment $\mathbf{z}$ that is consistent with $\mathbf{x}$ becomes invalid, it passes more than its own weight to $\sigma(\mathbf{z})$. $\square$

4.2 Lattices of partial assignments

Let $\mathcal{L}$ denote a collection of subsets of a ground set P, i.e. $\mathcal{L} \subseteq 2^P$.

Definition 6. $\mathcal{L}$ is called a *lattice* if for any two sets $S_1, S_2 \in \mathcal{L}$ there exists a unique infimum, and a unique supremum of S_1 and S_2 in $\mathcal{L}$ denoted by $S_1 \wedge S_2$ and $S_1 \vee S_2$.

More explicitly, there exist sets $\overline{S}, \underline{S} \in \mathcal{L}$, such that $\overline{S} \supseteq S_1, S_2$, $\underline{S} \subseteq S_1, S_2$, and for any $S \supseteq S_1, S_2$ it holds that $S \supseteq \overline{S}$, and similarly for any $S \subseteq S_1, S_2$ it holds that $S \subseteq \underline{S}$.

Definition 7. A lattice is *distributive* if $(S_1 \wedge S_2) \vee S_3 = (S_1 \vee S_3) \wedge (S_2 \vee S_3)$ and $(S_1 \vee S_2) \wedge S_3 = (S_1 \wedge S_3) \vee (S_2 \vee S_3)$.

There are a number of equivalent definitions of distributive lattice. One property that is of interest to us is that a distributive lattices can be thought of as the set of order ideals of a partially ordered set.

Definition 8. For a partially ordered set P, a subset S is an *order ideal* if for every $a, b \in P$ with $a \leq b$, it holds that $b \in S$ implies $a \in S$.

It is a well known fact that the set of order ideals of a partially ordered set is a distributive lattice, and that for every distributive lattice one can construct a partially ordered set that generates this lattice.

4.2.1 Implication lattices

Just as distributive lattices can be generated as the ideals of a partial order on a set P, we will show that the lattice of partial assignments consistent with a given satisfying assignment can be generated by a generalized partial order.

Definition 9. An *implication* is a pair (S, a), where $S \subset P$, $a \in P \backslash S$.

An implication (S, a) should be thought of as a subset S implying an element a. We will call $|S|$ the order of the implication. A partial order in particular is a set of implications of order 1 that does not contain cycles.

Definition 10. Let $\mathcal{I}$ be a collection of implications on P. An *implication ideal* of $(P, \mathcal{I})$ is a subset $H \subseteq P$ such that for every $(S, a) \in \mathcal{I}$, if $S \subseteq H$ then $a \in H$.

Notice that the intersection of implication ideals is an implication ideal, whereas their union may not be one. Also P is always an implication ideal. This implies that the set of implication ideals is a lattice (see [Sta97]), and we will call it an *implication lattice*. We will show that any collection of subsets of P that contains P and is closed under intersection is the set of implication ideals for some collection of implications. Therefore implication lattices are equivalent to what is known in the literature as an *alignment*.

Theorem 13. *For any collection $\mathcal{H}$ of subsets of P that satisfies (1) $P \in \mathcal{H}$; and (2) $\mathcal{H}$ is closed under intersection; we can construct a collection of implications $\mathcal{I}$ on P, so that $\mathcal{H}$ is the set of implication ideals on $(P, \mathcal{I})$.*

Proof. We start with $\mathcal{I}$ being the complete set of implications on P. For every $S \in \mathcal{H}$ remove from $\mathcal{I}$ all implications (F, a), where $F \subseteq S$ and $a \notin S$. We claim that $\mathcal{H}$ is the set of implication ideals of the resulting collection of implications $\mathcal{I}$.

It is immediate by the construction that every $S \in \mathcal{H}$ is an implication ideal. It remains to show that there are no other implication ideals. Suppose $D \subset P$ is an ideal for $\mathcal{I}$. That means that all implications (F, a) with $F \subseteq D$ and $a \notin D$ have been removed. In particular (D, a) has been removed for all $a \notin D$. This implies that for every $a \notin D$ there exists $S_a \in \mathcal{H}$ such that $D \subseteq S_a$ and $a \notin S_a$. Since an implication lattice is closed under intersection, the intersection of all such S_a is in $\mathcal{H}$. However their intersection is exactly D, so $D \in \mathcal{H}$. ☐

The set of implications constructed above is the maximal set of implications that generates $\mathcal{H}$. It would be interesting to characterize the class of implication lattices that can be generated by implications of particular order. In the case of order-1 implications the resulting implication lattice is a distributive lattice, even when the order contains cycles.

4.2.2 Balanced lattices

Definition 11. Let $\mathcal{L}$ be a collection of subsets of a non-empty set P. We will say that an element $a \in P$ is *removable* in a set $S \in \mathcal{L}$ if the set $S \backslash \{a\}$ is also in the collection $\mathcal{L}$. We denote the set of removable elements of S by $U(S)$ and the non-removable by $C(S)$.

Definition 12. We will say that a set S is *accessible* from P if there is a sequence of elements $a_1, \ldots, a_r \in P$ such that $P \backslash S = \{a_1, \ldots, a_r\}$ and $P \backslash \{a_1, a_2, \ldots, a_i\} \in \mathcal{L}$ for every $i = 1, \ldots, r$.

Definition 13. A *balanced* lattice is an implication lattice $\mathcal{L}$ on P such that every set $S \in \mathcal{L}$ is accessible from P.

Theorem 14. *Let $\mathcal{L} \subseteq 2^P$ be a collection of subsets of a non-empty set P. The following statements are equivalent:*

1. *$\mathcal{L}$ is an implication lattice in which every set $S \in \mathcal{L}$ is accessible from P.*
2. *The intervals $(U(S), S)$ for $S \in \mathcal{L}$ partition the boolean lattice 2^P, i.e. for every $D \subseteq P$ there is a unique $S \in \mathcal{L}$ such that $U(S) \subseteq D \subseteq S$.*
3. *For any field and any set of values $(t_a : a \in P)$ in the field:*

$$\sum_{S \in \mathcal{L}} \prod_{a \in U(S)} t_a \times \prod_{a \notin S} (1 - t_a) = 1 \tag{4.2}$$

Proof. **1. implies 2.** Consider any subset $D \subseteq P$. It suffices to show that there is a unique set $S \in \mathcal{L}$, $S \leq P$ such that $U(S) \subseteq D \subseteq S$. To find such an S we start with P and remove removable elements from $P \backslash D$ until there are none left. The set S we reach this way may not be unique, since it may depend on the order in which we remove the elements. However it certainly satisfies $U(S) \subseteq D \subseteq S \leq P$. To show that it is unique, suppose there are two such sets T and S. Let their corresponding sequences of order ideals be $P = T_0 \subset T_1 \subset T_2 \subset \ldots \subset T_k = T$ and $P = S_0 \subset S_1 \subset S_2 \subset \ldots \subset S_k = S$. Let i be the smallest index for which $T \not\subset S_i$, and a is the unique element in $S_{i-1} - S_i$. a is removable in S_{i-1}, and since $a \notin D$, a is not removable in T. Since a is not removable in T there must exist an implication (F, a) for some $F \subseteq T$. However, since $T \subseteq S_{i-1}$ this implication would mean that a is not removable in S_{i-1} either. This is a contradiction.

2. implies 1. By Theorem 13 it suffices to show that P in $\mathcal{L}$, that $\mathcal{L}$ is closed under intersection, and that every $S \in \mathcal{L}$ is accessible from P.

We define the map $\phi : 2^P \to \mathcal{L}$, as follows: for every $D \in\subseteq P$, let $\phi(D)$ be the unique element $S \in \mathcal{L}$ such that $U(S) \subseteq D \subseteq S$.

The first condition obviously holds, because $\phi(P) = P$ is in $\mathcal{L}$. Next we show that every set $S \in \mathcal{L}$ is accessible from any superset $T \supseteq S$ that also belongs to $\mathcal{L}$. It suffices to prove that there exists an element $a \in T \backslash S$ that is removable in T. Suppose no element in $T \backslash S$ is removable in S. Then $\phi(S)$ could be either S or T since both satisfy the conditions of the map. Since the map is unique, we have reached a contradiction.

Finally, we need to prove that $\mathcal{L}$ is closed under intersection. We need a simple lemma:

Lemma 15. *If an element $a \in S \in \mathcal{L}$ is removable in S then it is removable in every $T \in \mathcal{L}$, for which $a \in T \subset S$.*

Proof. It suffices to prove the statement for $|T| = |S| - 1$. Let $b \in P$ be the element of S that is not in T, i.e. $T = S \backslash \{b\}$. Since a is removable in S, the set $Q = S \backslash \{a\}$ is in $\mathcal{L}$. Now, suppose a is not removable in T. Then the set $S \backslash \{a, b\} \notin \mathcal{L}$, and it can map both to $S \backslash \{a\}$ and $S \backslash \{b\}$. This is a contradiction by the uniqueness of ϕ. □

For any $M \in \mathcal{L}$, and $D \subseteq M$ let $\phi_M(D)$ denote the minimal set in $\mathcal{L}$ reachable from M by removing elements from $M \backslash D$ sequentially, going only through sets in $\mathcal{L}$ in the process. $\phi_M(D)$ is well-defined because the order of removing the elements does not matter. Suppose there were two such minimal sets S and T. Without loss of generality $T \backslash S \neq \emptyset$. Let their corresponding sequences of sets in $\mathcal{L}$ be $M = T_0 \subset T_1 \subset T_2 \subset \ldots \subset T_k = T$ and $M = S_0 \subset S_1 \subset S_2 \subset \ldots \subset S_k = S$. Let i be the smallest index for which $T \not\subseteq S_i$, and a is the unique element in $S_{i-1} - S_i$. Clearly, a is removable in S_{i-1}. Since $T \subseteq S_{i-1}$ by lemma 15 a is removable in T. However this contradicts the fact that T is a minimal set, because $a \notin D$. Therefore $\phi_S(D)$ is well defined. Notice that $\phi_S(D)$ satisfies $U(\phi_S(D)) \subseteq D \subseteq \phi_S(D)$, therefore $\phi_S(D) = \phi(D)$ for every $S \in \mathcal{L}$ and $D \subseteq S$.

Since any set that is both subset of S_1 and S_2 is contained in $S_1 \cap S_2$, in order to prove that $S_1 \wedge S_2 = S_1 \cap S_2$ it suffices to prove that $S_1 \cap S_2 \in \mathcal{L}$. Let's denote by S: $\phi(S_1 \cap S_2) = \phi_{S_1}(S_1 \cap S_2) = \phi_{S_2}(S_1 \cap S_2) = S \in \mathcal{L}$. The first definition for S implies $S \supset S_1 \cap S_2$, the second implies $S \subseteq S_1$, and the third $S \subseteq S_2$. Thus the only possible value for S is $S = S_1 \cap S_2$. Therefore $S_1 \cap S_2 \in \mathcal{L}$.

2. implies 3. Let $\overline{t_a} = 1 - t_a$. Observe that

$$\sum_{D \subseteq P} \prod_{a \in D} t_a \prod_{a \notin D} \overline{t_a} = \prod_{a \in P} (t_a + \overline{t_a}) = 1$$

Since for every D there is a unique S such that $U(S) \subseteq D \subseteq S$, it suffices to prove that for every $S \in \mathcal{L}$:

$$\prod_{a \in U(S)} t_a \times \prod_{a \notin S} \overline{t_a} = \sum_{D\,:\,U(S) \subseteq D \subseteq S} \prod_{a \in D} t_a \prod_{a \notin D} \overline{t_a}$$

This is easily seen to be true because:

$$\begin{aligned}
\sum_{D\,:\,U(S) \subseteq D \subseteq S} \prod_{a \in D} t_a \prod_{a \notin D} \overline{t_a} &= \prod_{a \in U(S)} t_a \prod_{a \notin S} \overline{t_a} \sum_{R \subseteq C(S)} \prod_{a \in R} t_a \prod_{a \in C(S) \setminus R} \overline{t_a} \\
&= \prod_{a \in U(S)} t_a \prod_{a \notin S} \overline{t_a} \prod_{a \in C(S)} (t_a + \overline{t_a}) \\
&= \prod_{a \in U(S)} t_a \prod_{a \notin S} \overline{t_a}
\end{aligned}$$

3. implies 2. Consider any set $D \subseteq P$, and set $t_a = 1$ if $a \in D$ and 0 otherwise. The equality becomes:

$$\begin{aligned}
1 &= \sum_{S \in \mathcal{L}} \prod_{a \in U(S)} t_a \times \prod_{a \notin S} \overline{t_a} \\
&= \sum_{S \in \mathcal{L}\,:\,U(S) \subseteq D \subseteq S} 1
\end{aligned}$$

Therefore there is exactly one set $S \in \mathcal{L}$ for which $U(S) \subseteq D \subseteq S$. $\square$

4.3 Bound on the threshold for solutions with cores

We will prove that for density above 4.46 with high probability there are no covers that can be extended to satisfying assignments. We do this in three steps. Let the *size* of the cover be the number of variables assigned 0 or 1 value. First we show that covers with sub-linear size do not exist with high probability. Next we show that covers of linear size either $\leq 0.25n$ or $\geq 0.7n$ do not exist with high probability. Finally we show that covers that can be extended to satisfying assignments (i.e. cores) of size in the intermediate range from $0.25n$ to $0.7n$ do not exist with high probability.

4.3.1 Typical size of covers

The following lemma establishes that covers and consequently cores, if they exist, with high probability have at least a certain linear fraction $c(\alpha, k)$ of the variables assigned 0 or 1 value.

Lemma 16. *Let ϕ be a random k-sat formula with $m = \alpha n$ clauses where $k \geq 3$. Then for all positive integers C it holds that*

$$\Pr[\, \phi \textit{ has a cover with } C \textit{ clauses}\,] \quad \leq \quad \left(\frac{e^2 \alpha C^{k-2}}{n^{k-2}} \right)^C , \tag{4.3}$$

Consequently, if we define $c(\alpha, k) := (\alpha e^2)^{-1/(k-2)}$, then with probability tending to one as $n \to +\infty$, there are no covers of size strictly less than $c(\alpha, k)\, n$.

Proof. Suppose that the formula ϕ has a cover with C clauses. Note that the variables in these clauses all lie in some set of at most C variables. Thus the probability that a cover with C clauses exists is bounded by the probability that there is a set of C clauses whose variables lie in some set of size $\leq C$. This probability is bounded by

$$\binom{m}{C} \binom{n}{C} \left(\frac{C}{n} \right)^{Ck} ,$$

which can be upper bounded by

$$\left(\frac{em}{C} \right)^C \left(\frac{en}{C} \right)^C \left(\frac{C}{n} \right)^{Ck} = \left(\frac{e^2 \alpha C^{k-2}}{n^{k-2}} \right)^C ,$$

as needed. $\square$

In particular for $k = 3$, covers have size at least $\frac{1}{\alpha \times e^2} n$.

4.3.2 Ruling out small covers and large covers

We prove that a random formula with density $\alpha \geq 4.46$ does not have covers of size $\leq 0.25n$ or of size $\geq 0.7n$ with high probability.

Lemma 17. *The function f is defined as follows:*

$$\begin{aligned} f(a,r,\alpha) &= \ln \frac{2^a}{a^a(1-a)^{1-a}} + \frac{a}{e^r-1}\left((e^r(1-r)-1)\ln(e^r-1)+(r-1)re^r\right) \\ &+ \alpha \ln\left(\left(\frac{3a^3}{8d}\right)^d \left(\frac{4-a^2(3-a)}{4(1-d)}\right)^{1-d}\right), \end{aligned}$$

where d denotes $\frac{re^r a}{(e^r-1)\alpha}$.

*If $a \in [0,1]$ and $\alpha > 0$ are such that for every $r > 0$ with $\frac{re^r a}{(e^r-1)\alpha} \leq 1$, it holds that $f(a,r,\alpha) < 0$, then with high probability random 3-*SAT *formulas of density α do not have covers of size an.*

Proof. For any $s \in \{0,1,\ldots,n\}$, let X_s denote the number of cover assignments σ of size s.

$$\begin{aligned} \mathrm{E}[X_s] &= \mathrm{E}\left[\sum_{\sigma\in\{0,1,*\}^n} \mathrm{Ind}\left[\sigma \text{ is valid } \cap\ (n_*(\sigma)=n-s)\ \cap\ (n_o(\sigma)=0)\right]\right] \\ &= \sum_{\sigma\in\{0,1,*\}^n : n_*(\sigma)=n-s} \Pr[\sigma \text{ is valid } \cap\ (n_o(\sigma)=0)] \\ &= \binom{n}{s} 2^s \ \Pr[\sigma = (0^s *^{n-s}) \text{ is valid and } x_1,\ldots,x_s \text{ are constrained}] \end{aligned}$$

In order to claim that covers of size an do not exist at some density α it suffices to show that $\lim_{n\to\infty}\frac{1}{n}\ln \mathrm{E}[X_s] \leq 0$. Let $a = s/n$. We denote by P the probability that $\sigma = (0^s *^{n-s})$ is valid and all of its assigned variables are constrained. P is equivalent to the probability of the following event in an experiment with $m = \alpha n$ balls thrown uniformly and independently at random into $2^3\binom{n}{3}$ bins. There are 3 kinds of bins:

1. Bins of type 1 should be empty. These are the clauses that are not allowed:
   - $(x_{i_1} \vee x_{i_2} \vee x_{i_3})$, with $i_1, i_2, i_3 \in \{1,2,\ldots,s\}$,
   - $(x_{i_1} \vee x_{i_2} \vee \bar{x}_j)$, with $i_1, i_2 \in \{1,2,\ldots,s\}$ and $j > s$,
   - $(x_{i_1} \vee x_{i_2} \vee x_j)$, with $i_1, i_2 \in \{1,2,\ldots,s\}$ and $j > s$,

The total number of these is

$$\binom{s}{3} + 2(n-s)\binom{s}{2} = n^3 \left(\frac{a^3}{6} + a^2(1-a)\right) + o(n^3).$$

2. Bins of type 2 correspond to constraining clauses: $(x_{i_1} \vee x_{i_2} \vee \bar{x}_t)$, with $i_1, i_2, t \in \{1, 2, \ldots, s\}$. For each variable x_t there are $\binom{s-1}{2} = n^2 \, \frac{a^2}{2} + o(n^2)$ clauses that could constrain it. Equivalently, there has to be at least one ball in one of those bins for every x_t with $t \in \{1, 2, \ldots, s\}$.

3. There are no constraints for the remaining bins of type 3. Their total number is:

$$2^3\binom{n}{3} - n^3 \left(\frac{a^3}{6} + a^2(1-a)\right) - n^3 \, \frac{a^3}{2} + o(n^3)$$

$$= \frac{n^3}{3}\,(4 - a^2(3-a)) + o(n^3)$$

Suppose $m' = dm$ of the clauses we choose are of type 2, and the remaining $m - m'$ are of type 3. The probability of this event is:

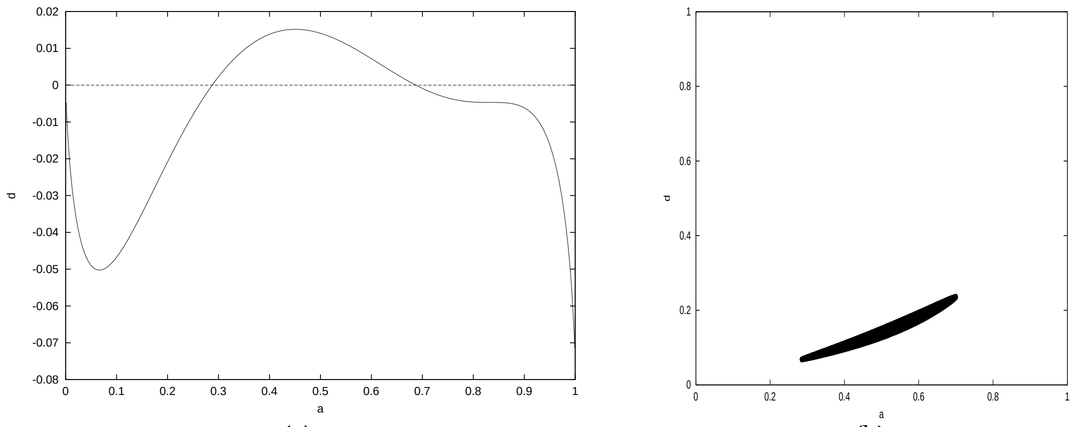
$$p_{m'} = \binom{m}{m'} \left(\frac{3\,a^3}{8} + o(1)\right)^{m'} \left(1 - \frac{a^2\,(3-a)}{4} + o(1)\right)^{m-m'} .$$

The probability that the m' clauses of type 2 are such that there is one of each kind, is the same as the coupon collectors probability of success, with $s = an$ different coupons, and $m' = (d\alpha)n$ trials. We will use the following general fact [Chv91]: Let $q(cN, N)$ denote the probability of collecting N coupons within cN trials. If $c < 1$, $q(cN, N) = 0$. Otherwise, as N goes to infinity $q(cN, N)$, grows like $g(c)^N$, where $g(c) = (e^{r_0} - 1)\left(\frac{c}{er_0}\right)^c$, and r_0 is the solution of $\frac{re^r}{e^r - 1} = c$. More precisely, $\lim_{N\to\infty} \frac{1}{N} \ln q(cN, N) = \ln g(c)$.

$$\begin{aligned} P &= \sum_{m'} \binom{m}{m'} \left(\frac{3\,a^3}{8} + o(1)\right)^{m'} \left(1 - \frac{a^2\,(6-2a)}{8} + o(1)\right)^{m-m'} q(m', an) \\ &\leq m \max_{m'} \left\{ \binom{m}{m'} \left(\frac{3\,a^3}{8} + o(1)\right)^{m'} \left(1 - \frac{a^2\,(3-a)}{4} + o(1)\right)^{m-m'} q(m', an) \right\} \end{aligned}$$

Finally,

$$E[X_s] \leq \binom{n}{an} 2^{an} \max_{m'} \left\{ \tbinom{m}{m'} \left(\tfrac{3\,a^3}{8} + o(1)\right)^{m'} \left(1 - \tfrac{a^2\,(3-a)}{4} + o(1)\right)^{m-m'} q(m', an) \right\}$$



(a) (b)

Figure 4.2. (a) The value of $\max_d f(a,d,4.46)$.
(b) The (a,d) pairs for which $f(a,d,4.46) > -0.001$.

$$\begin{aligned}
\lim_{n\to\infty} \frac{1}{n} \ln \mathrm{E}[X_s] \quad &\le \quad \ln\left(\frac{2^a}{a^a(1-a)^{1-a}}\right) \\
&\quad + \max_{d\in[0,1]} \left\{ \alpha \ln \frac{\left(\frac{3\ a^3}{8}\right)^d \left(1-\frac{a^2\ (3-a)}{4}\right)^{1-d}}{d^d\ (1-d)^{1-d}} + a \ln g\left(\frac{d\alpha}{a}\right) \right\} \\
&= \quad \max_{d\in[0,1]} f(a,r,\alpha)
\end{aligned}$$

To show that $\lim_{n\to\infty} \frac{1}{n} \ln \mathrm{E}[X_s] \le 0$ it suffices to show that $f(a,r,\alpha) < 0$ for every r such that $d \in [0,1]$. $\square$

Figure 4.2 shows the value of the function f for $\alpha = 4.46$ for every $a \in (0,1)$, maximized over d. The maximum here has been estimated by taking the maximum value of the function for $r = 0, 0.001, 0.002, \ldots,$ such that $d \le 1$. The function is negative for $a \in (0, 0.25]$ and for $a \in [0.7, 1]$.

4.3.3 Ruling out cores of intermediate size

We prove that a random formula of density $\alpha > 4.46$ does not have cores of size between $0.25n$ and $0.7n$ with high probability.

Lemma 18. *The functions h and g are defined as follows:*

$$\begin{aligned} h(a,r,\alpha,\rho,b) &= \ln \frac{(1-a)^{1-a}\, 2^b\, \rho^{1-a-b}}{b^b\,(1-a-b)^{1-a-b}} \\ &\quad - \frac{\alpha\,(1-d)\,b\,(b(6-5b-3a)+12(1-a-b)a)}{2(4-a^2(3-a))} \\ &\quad + b\ln\left(1-\rho e^{-\frac{3\alpha(1-d)b(b+2a)}{2(4-a^2(3-a))}}\right) \\ g(a,r,\alpha) &= f(a,r,\alpha) + \min_{\rho\in[0,1]}\ \max_{b\in[0,1-a]} h(a,r,\alpha,\rho,b). \end{aligned}$$

*If $a \in [0,1]$ and $\alpha > 0$ are such that for every $r > 0$ with $\frac{re^r a}{(e^r-1)\alpha} \leq 1$, it holds that $g(a,r,\alpha) < 0$, then with high probability random 3-*SAT *formulas of density α do not have cores of size an.*

Proof. Let Y_s denote the number of covers of size $s = an$ that can be extended to satisfying assignments. We will show that for s in the range from $0.25n$ to $0.7n$, $\mathrm{E}[Y_s]$ goes to 0 exponentially fast. It will be convenient to denote by $\varphi_\sigma(x_{S^*(\sigma)})$ the formula that is obtained by substituting the variables that have $0/1$ assignments in σ.

$$\begin{aligned} \mathrm{E}[Y_s] &= \sum_{\sigma\in\{0,1,*\}^n : n_*(\sigma)=n-s} \Pr[\sigma \text{ is valid } \cap\ (n_o(\sigma)=0)\ \cap\ \varphi_\sigma(x_{S^*(\sigma)}) \text{ is satisfiable}] \\ &= \binom{n}{s} 2^s\ \Pr[\sigma = (0^s *^{n-s}) \text{ is valid } \cap\ (n_o(\sigma)=0)\ \cap\ \varphi_\sigma(x_{S^*(\sigma)}) \text{ is satisfiable}] \\ &= \binom{n}{s} 2^s \times \sum_{m'=an}^{m} p_{m'}\, q(m',s) \\ &\qquad\qquad \times \Pr[\ \varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is satisfiable} \mid m-m' \text{ clauses of type 3}]. \end{aligned}$$

We will bound this probability using the Poisson approximation. There are two related random models. In the first model, that we call the *exact model*, $(m-m')$ clauses are chosen uniformly at random without replacement from all $M = n^3(4-a^2(3-a))/3 + o(n^3)$ clauses of type 3. In the second model, which we call the *Poisson model*, each of the M clauses is included in the formula with probability $p = (m-m')/M = \frac{3\alpha(1-d)}{n^2(4-a^2(3-a))} + \omega\left(\frac{1}{n^2}\right)$. The expected number of clauses in both models is the same, namely $(m-m')$. Let $\mathbb{P}_p$ denote probability in the Poisson model, and $\mathbb{P}_m$ denote probability in the exact model. Let J be a random variable for the number of clauses included in the formula in the Poisson model.

$$\begin{aligned}
\mathbb{P}_p[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is satisfiable}] &= \sum_{i=0}^{M} \mathbb{P}_p[J=i] \times \mathbb{P}_i[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is sat.}] \\
&\geq \sum_{i=0}^{m} \mathbb{P}_p[J=i] \times \mathbb{P}_i[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is sat.}]
\end{aligned}$$

Since the probability of the event decreases as m increases, and $\mathbb{P}_p[J \leq m] \geq 1/2$, we get that:

$$\begin{aligned}
\mathbb{P}_p[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is satisfiable}] &\geq \sum_{i=0}^{m} \mathbb{P}_p[J=i] \times \mathbb{P}_i[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is sat.}] \\
&\geq \mathbb{P}_m[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is sat.}] \times \sum_{i=0}^{m} \mathbb{P}_p[J=i] \\
&\geq \mathbb{P}_m[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is satisfiable}] \times \frac{1}{2}
\end{aligned}$$

Thus, it suffices to bound $\mathbb{P}_p[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \text{ is satisfiable}]$. Any of the first moment techniques for bounding the satisfiability threshold would result in a bound for this probability too. We obtain the strongest result using a novel idea of applying the first moment method to the MRF on partial assignments.

Lemma 19. *In the Poisson model with parameters n, $m = \alpha n$, $m' = dm$, $s = an$ and for $\sigma = (0^s *^{n-s})$, and r such that $d = \frac{re^r a}{(e^r-1)\alpha}$,*

$$\lim_{n\to\infty} \frac{1}{n} \ln \mathbb{P}_p[\,\varphi_\sigma(x_{s+1},\ldots,x_n) \textit{ is satisfiable}] \leq \min_{\rho\in[0,1]} \max_{b\in[0,1-a]} h(a,r,\alpha,\rho,b).$$

Proof. By the weight preservation theorem the probability of satisfiability is bounded from above by the expected value of the partition function for any value of ρ. We bound this expectation.

For any $t \in \{0,1,\ldots,n-s\}$ let Z_t denote the sum of the weights of valid assignments τ for $\varphi_\sigma(x_{s+1},\ldots,x_n)$ such that $n_*(\tau) = 1-s-t$. Then

$$\mathrm{E}[Z] = \sum_{t=0}^{n-s} \mathrm{E}[Z_t] \;\leq\; n \max_{t\in\{0,1,\ldots,n-s\}} \mathrm{E}[Z_t].$$

$$\begin{aligned}
\mathrm{E}[Z_t] &= \sum_{u=0}^{t} \rho^{1-s-t}(1-\rho)^u \sum_{\tau\in\{0,1,*\}^{n-s}} \Pr\left[\tau \text{ is valid } \cap \ (n_*(\tau) = 1-s-t) \ \cap \ (n_o(\tau) = u)\right] \\
&= \rho^{n-s-t} \sum_{u=0}^{t}(1-\rho)^u \sum_{\tau\in\{0,1,*\}^{n-s} \,:\, n_*(\tau)=1-s-t} \Pr[\tau \text{ is valid} \cap (n_o(\tau) = u)] \\
&= \rho^{n-s-t} \sum_{u=0}^{t}(1-\rho)^u \binom{n-s}{t}\binom{t}{u} 2^t \\
&\qquad\qquad \Pr[\tau = (0^t *^{n-s-t}) \text{ is valid and } x_{s+1},\ldots,x_{s+u} \text{ are unconstrained}]
\end{aligned}$$

Next we derive the probability that the assignment $(x_{s+1}, x_{s+2}, \ldots, x_n) = (0^t *^{n-s-t})$ is valid and the first u variables are unconstrained. This is equivalent to the probability that there are no clauses of the following kinds:

- $(x_{i_1} \vee x_{i_2} \vee x_{i_3})$, with $i_1 \in \{1, 2, \ldots, s+t\}$, $i_2, i_3 \in \{s+1, s+2, \ldots, s+t\}$,
- $(x_{i_1} \vee x_{i_2} \vee \bar{x}_j)$, with $i_1 \in \{1, 2, \ldots, s+t\}$, $i_2 \in \{s+1, s+2, \ldots, s+t\}$ and $j > s+t$,
- $(x_{i_1} \vee x_{i_2} \vee x_j)$, with $i_1 \in \{1, 2, \ldots, s+t\}$, $i_2 \in \{s+1, s+2, \ldots, s+t\}$ and $j > s+t$,
- $(x_{i_1} \vee x_{i_2} \vee \bar{x}_j)$, with $i_1 \in \{1, 2, \ldots, s+t\}$, $i_2 \in \{s+1, s+2, \ldots, s+t\}$ and $j \in \{s+1, \ldots, s+u\}$,

and for every $j \in \{s+u+1, \ldots, s+t\}$, the formula contains a clause $(x_{i_1} \vee x_{i_2} \vee \bar{x}_j)$, with $i_1 \in \{1, 2, \ldots, s+t\}$, $i_2 \in \{s+1, \ldots, s+t\}$. In the Poisson model all clauses are independent, so it is easy to put these events together to obtain:

$$\begin{aligned}
P &= \Pr[\tau = (0^t *^{n-s-t}) \text{ is valid and } x_{s+1},\ldots,x_{s+u} \text{ are unconstrained}] \\
&= (1-p)^{\binom{t}{3} + s\binom{t}{2} + 2(n-s-t)\left(\binom{t}{2} + st\right) + u\left(\binom{t}{2}+st\right)} \times \left(1-(1-p)^{\binom{t}{2}+st}\right)^{t-u}
\end{aligned}$$

The expression for the expectation can be simplified:

$$\begin{aligned}
\mathrm{E}[Z_t] &= \rho^{n-s-t}\binom{n-s}{t}2^t\sum_{u=0}^{t}(1-\rho)^u\binom{t}{u}P \\
&= \rho^{n-s-t}\binom{n-s}{t}2^t\,(1-p)^{\binom{t}{3}+s\binom{t}{2}+2(n-s-t)\left(\binom{t}{2}+st\right)} \\
&\qquad \times\sum_{u=0}^{t}\binom{t}{u}(1-\rho)^u\,(1-p)^{u\left(\binom{t}{2}+st\right)}\times\left(1-(1-p)^{\binom{t}{2}+st}\right)^{t-u} \\
&= \rho^{n-s-t}\binom{n-s}{t}2^t\,(1-p)^{\binom{t}{3}+s\binom{t}{2}+2(n-s-t)\left(\binom{t}{2}+st\right)} \\
&\qquad \times\left((1-\rho)(1-p)^{\binom{t}{2}+st}+\left(1-(1-p)^{\binom{t}{2}+st}\right)\right)^t \\
&= \rho^{n-s-t}\binom{n-s}{t}2^t\,(1-p)^{\binom{t}{2}(6n-5t-3s-2)/3+2(n-s-t)st}\left(1-\rho(1-p)^{\binom{t}{2}+st}\right)^t
\end{aligned}$$

Let $t=bn$, and recall that $s=an$, $p=\frac{3\alpha(1-d)}{n^2(4-a^2(3-a))}+\frac{1}{o(n^2)}$.

$$\begin{aligned}
\lim_{n\to\infty}\frac{1}{n}\ln\mathrm{E}[Z_t] &= \ln\frac{(1-a)^{1-a}\,2^b\,\rho^{1-a-b}}{b^b\,(1-a-b)^{1-a-b}} \\
&\quad -\frac{\alpha\,(1-d)\,b\,(b(6-5b-3a)+12(1-a-b)a)}{2(4-a^2(3-a))} \\
&\quad +b\ln\left(1-\rho e^{-\frac{3\alpha(1-d)b(b+2a)}{2(4-a^2(3-a))}}\right) \\
&= h(a,r,\alpha,\rho,b)
\end{aligned}$$

□

Substituting the bound of Lemma 19 into the expression for the expectation yields:

$$\lim_{n\to\infty}\frac{1}{n}\mathrm{E}[Y_s]\le g(a,r,\alpha).$$

Figure 4.3 shows the value of g for $\alpha=4.46$ in the region of values for a and d where $\max_{b\in[0,1-a]}f(a,r,4.46,\rho,b)\ge -0.001$ (for all ρ). This value is always negative, therefore with high probability there are no cores of size s. □

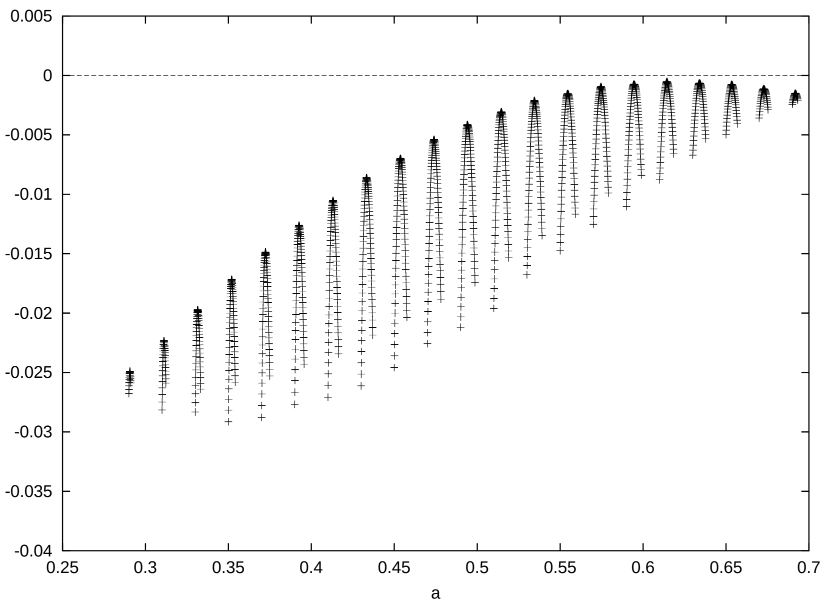


Figure 4.3: (a) The value of $g(a, d, 4.46)$.

Chapter 5

The connectivity of Boolean satisfiability: computational and structural dichotomies

This chapter is dedicated to a study of the connectivity properties of the space of solutions of general Boolean constraint satisfaction problems. Recall that the solutions of a given n-variable Boolean formula φ induce a subgraph $G(\varphi)$ of the n-dimensional hypercube. While there has been an intensive study of the structure of the solution space of Boolean satisfiability problems for random instances, this work seems to be the first to explore this issue from a worst-case viewpoint.

Recall the definitions of Boolean CSPs and Schaefer's dichotomy theorem given in Chapter 2. We find that with respect to the connectivity properties of the solution graphs there are again two kinds of Boolean satisfiability problems. On one hand we find *tight* sets of relations which have "good" structural properties. On the other hand we find *non-tight* sets of relations which can express any solution graph. Surprisingly, the boundary between these two classes differs from the boundary in Schaefer's dichotomy.

The core of Schaefer's theorem is an expressibility result which asserts that every Boolean constraint satisfaction formula can be obtained as a projection over a formula built from clauses from any "hard" set of relations (i.e. a set in which at least one relation is not bijunctive, at least one is not Horn, at least one is not dual Horn, and at least one is not affine). The crux of our results is an expressibility theorem which we refer to as the Faithful Expressibility Theorem. It says that for any Boolean constraint satisfaction formula with a solution graph G, we can construct a formula using any non-tight set of relations, such that its solution graph is isomorphic to G after certain adjacent vertices are merged. An example of the difference between a faithful and an unfaithful expression is shown in Figure 5.1.

As a consequence of the Faithful Expressibility Theorem we obtain three dichotomy results. The first is a dichotomy theorem for the st-connectivity problem. We show that st-connectivity is solvable in linear time for formulas built from tight relations, and PSPACE-complete in all other cases. The second is a dichotomy theorem for the connectivity problem: it is in coNP for formulas built from tight relations, and PSPACE-complete in all other cases. Third, in addition to the two complexity-theoretic dichotomies, we establish a structural dichotomy theorem for the diameter of the connected components of the solution space of Boolean formulas. This result asserts that, in the PSPACE-complete cases, the diameter of the connected components can be exponential, but in all other cases it is linear.

To prove PSPACE-completeness we show that both connectivity and st-connectivity are hard for 3-CNF formulas; this is proved by a reduction from a generic PSPACE computation. This fact together with the Faithful Expressibility Theorem implies PSPACE-completeness for all other non-tight sets of relations.

In the case of tight sets of relation, we show that every component has a unique minimum element, or every component has a unique maximum element, or the Hamming distance coincides with the shortest-path distance in the relation. These properties are inherited by every formula built from a tight set of relations, and yield both small diameter and linear algorithms for st-connectivity.

An intriguing byproduct of our work is that we have identified a broad class of NP-complete satisfiability problems - those built from tight relations - that have simple structural properties, such as linear diameter. It would be interesting to investigate if these properties make random instances built from tight relations easier for Walk-SAT and similar heuristics, and if so, whether such heuristics are amenable to rigorous analysis.

In the next section we will introduce the main concepts precisely, and state our results. We will give the proofs for the two sides of the dichotomy separately, in Sections 5.2 and 5.3 respectively. Finally, we will discuss a few open questions and conjectures in Section 5.4.

5.1 Statements of connectivity theorems

Recall that if φ is a CNF($\mathcal{S}$)-formula with n variables, then $G(\varphi)$ denotes the subgraph of the n-dimensional hypercube induced by the solutions of φ. Thus, the vertices of $G(\varphi)$ are the

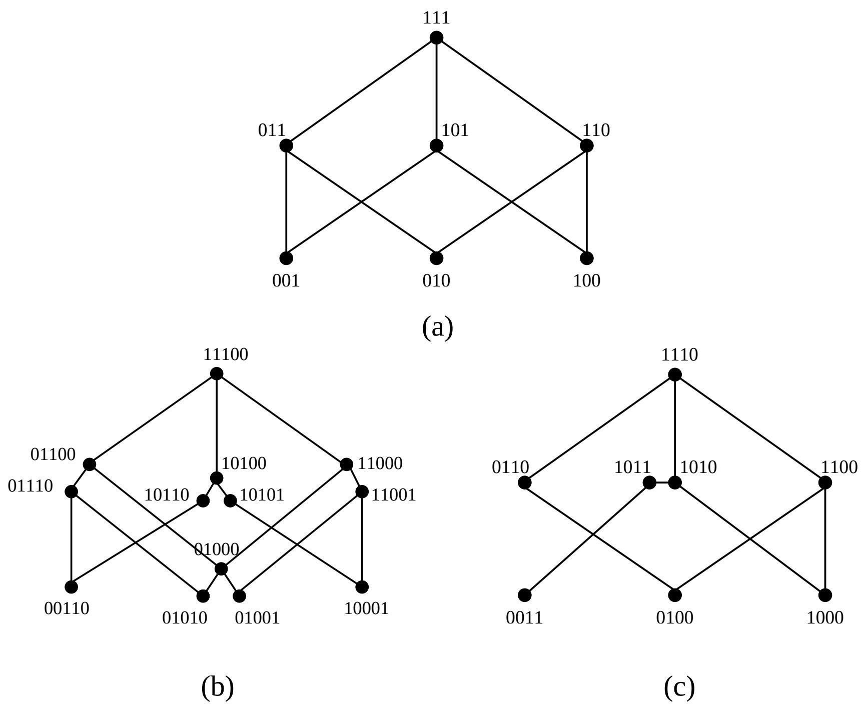


Figure 5.1. Expressing the relation $(x_1 \vee x_2 \vee x_3)$ using NOT-ALL-EQUAL relations.
(a) The graph of $(x_1 \vee x_2 \vee x_3)$;
(b) The graph of a faithful expression: $\varphi(\mathbf{x}, y_1, y_2) = R_{\text{NAE}}(x_1, x_2, y_1) \wedge R_{\text{NAE}}(x_2, x_3, y_2) \wedge R_{\text{NAE}}(y_1, y_2, 1)$.
(c) The graph of an unfaithful expression: $\varphi(\mathbf{x}, y_1) = R_{\text{NAE}}(x_1, x_2, y_1) \wedge R_{\text{NAE}}(\bar{y}_1, x_3, 0) \wedge R_{\text{NAE}}(y_1, x_2, 1)$.
In both cases $(x_1 \vee x_2 \vee x_3) = \exists \mathbf{y}\ \varphi(\mathbf{x}, \mathbf{y})$, but only in the first case the connectivity is preserved.

solutions of φ, and there is an edge between two solutions of $G(\varphi)$ precisely when they differ in a single variable.

We consider the following two algorithmic problems for CNF($\mathcal{S}$)-formulas.

1. The *connectivity* problem $\textsc{Conn}(\mathcal{S})$: given a CNF($\mathcal{S}$)-formula φ, is $G(\varphi)$ connected?

2. The *st-connectivity* problem $\textsc{st-Conn}(\mathcal{S})$: given a CNF($\mathcal{S}$)-formula φ and two solutions $\mathbf{s}$ and $\mathbf{t}$ of φ, is there a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(\varphi)$?

To pinpoint the computational complexity of $\textsc{st-Conn}(\mathcal{S})$ and $\textsc{Conn}(\mathcal{S})$, we need to introduce certain new types of relations.

Definition 14. Let $R \subseteq \{0,1\}^k$ be a logical relation.

1. R is *componentwise bijunctive* if every connected component of $G(R)$ is bijunctive.

2. R is OR-*free* if the relation $\mathrm{OR} = \{01, 10, 11\}$ cannot be obtained from R by setting $k-2$ of the coordinates of R to a constant $\mathbf{c} \in \{0,1\}^{k-2}$. In other words, R is OR-free if $(x_1 \vee x_2)$ is not definable from R by fixing $k-2$ variables.

3. R is NAND-*free* if $(\bar{x}_1 \vee \bar{x}_2)$ is not definable from R by fixing $k-2$ variables.

The next lemma follows from the closure properties of bijunctive, Horn, and dual Horn relations.

Lemma 20. *Let R be a logical relation.*

1. *If R is bijunctive, then R is componentwise bijunctive.*

2. *If R is Horn, then R is* OR-*free.*

3. *If R is dual Horn, then R is* NAND-*free.*

4. *If R is affine, then R is componentwise bijunctive,* OR-*free, and* NAND-*free.*

These containments are proper. For instance, $R_{1/3} = \{100, 010, 001\}$ is componentwise bijunctive, but not bijunctive as $\mathrm{maj}(100, 010, 001) = 000 \notin R_{1/3}$.

We are now ready to introduce the key concept of a *tight* set of relations.

Definition 15. A set $\mathcal{S}$ of logical relations is *tight* if at least one of the following three conditions holds:

1. Every relation in $\mathcal{S}$ is componentwise bijunctive;
2. Every relation in $\mathcal{S}$ is OR-free;
3. Every relation in $\mathcal{S}$ is NAND-free.

In view of Lemma 20, if $\mathcal{S}$ is Schaefer, then it is tight. The converse, however, does not hold. It is also easy to see that there is a polynomial-time algorithm for testing whether a given finite set $\mathcal{S}$ of logical relations is tight.

Schaefer's dichotomy theorem is a corollary of a stronger result known as the expressibility theorem. Similarly, the crux of our results is our *faithful* expressibility theorem. Its exact meaning will be explained in section 5.3, where the notion of faithful expressibility is defined precisely.

Theorem 21. (Faithful Expressibility Theorem) *Let $\mathcal{S}$ be a set of relations that is not tight. Every relation is faithfully expressible from $\mathcal{S}$.*

Using the faithful expressibility theorem, we obtain two dichotomy theorems regarding the computational complexity of $\textsc{Conn}(\mathcal{S})$ and $\textsc{st-Conn}(\mathcal{S})$.

Theorem 22. *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is tight, then* $\textsc{Conn}(\mathcal{S})$ *is in* coNP*; otherwise, it is* PSPACE*-complete.*

Theorem 23. *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is tight, then* $\textsc{st-Conn}(\mathcal{S})$ *is in* P*; otherwise,* $\textsc{st-Conn}(\mathcal{S})$ *is* PSPACE*-complete.*

We also show that if $\mathcal{S}$ is tight, but not Schaefer, then $\textsc{Conn}(\mathcal{S})$ is coNP-complete.

The dichotomy in the computational complexity of $\textsc{Conn}(\mathcal{S})$ and $\textsc{st-Conn}(\mathcal{S})$ is accompanied by a parallel structural dichotomy in the size of the diameter of $G(\varphi)$ (where, for a $\mathrm{CNF}(\mathcal{S})$-formula φ, the *diameter of* $G(\varphi)$ is the maximum of the diameters of the components of $G(\varphi)$).

Theorem 24. *Let $\mathcal{S}$ be a finite set of logical relations. If $\mathcal{S}$ is tight, then for every* $\mathrm{CNF}(\mathcal{S})$*-formula φ, the diameter of $G(\varphi)$ is linear in the number of variables of φ; otherwise, there are* $\mathrm{CNF}(\mathcal{S})$*-formulas φ such that the diameter of $G(\varphi)$ is exponential in the number of variables of φ.*

Our results and their comparison to Schaefer's Dichotomy Theorem are summarized in Table 5.1.

$\mathcal{S}$	SAT($\mathcal{S}$)	ST-CONN($\mathcal{S}$)	CONN($\mathcal{S}$)	Diameter
Schaefer	P	P	coNP	$O(n)$
Tight, non-Schaefer	NP-complete	P	coNP-complete	$O(n)$
Non-tight	NP-complete	PSPACE-complete	PSPACE-complete	$2^{\Omega(\sqrt{n})}$

Table 5.1: Structural and computational dichotomy results

As an example, the set $\mathcal{S} = \{R_{1/3}\}$, where $R_{1/3} = \{100, 010, 001\}$, is tight, but not Schaefer. It follows that SAT($\mathcal{S}$) is NP-complete (recall that this problem is POSITIVE 1-IN-3 SAT), ST-CONN($\mathcal{S}$) is in P, and CONN($\mathcal{S}$) is coNP-complete. Consider also the set $\mathcal{S} = \{R_{\text{NAE}}\}$, where $R_{\text{NAE}} = \{0,1\}^3 \setminus \{000, 111\}$. This set is not tight, hence SAT($\mathcal{S}$) is NP-complete (this problem is POSITIVE NOT-ALL-EQUAL 3-SAT), while both ST-CONN($\mathcal{S}$) and CONN($\mathcal{S}$) are PSPACE-complete.

We conjecture that if $\mathcal{S}$ is Schaefer, then CONN($\mathcal{S}$) is in P. If this conjecture is true, it will follow that the complexity of CONN($\mathcal{S}$) exhibits a *trichotomy*: if $\mathcal{S}$ is Schaefer, then CONN($\mathcal{S}$) is in P; if $\mathcal{S}$ is tight, but not Schaefer, then CONN($\mathcal{S}$) is coNP-complete; if $\mathcal{S}$ is not tight, then CONN($\mathcal{S}$) is PSPACE-complete.

5.2 The easy case of the dichotomy: tight sets of relations

In this section, we explore some structural properties for the solution graphs of tight sets of relations. These properties provide simple algorithms for CONN($\mathcal{S}$) and ST-CONN($\mathcal{S}$) for tight sets $\mathcal{S}$, and also guarantee that for such sets, the diameter of $G(\varphi)$ of CNF($\mathcal{S}$)-formula φ is linear.

We will use $\mathbf{a}, \mathbf{b}, \ldots$ to denote Boolean vectors, and $\mathbf{x}$ and $\mathbf{y}$ to denote vectors of variables. We write $|\mathbf{a}|$ to denote the Hamming weight (number of 1's) of a Boolean vector $\mathbf{a}$. Given two Boolean vectors $\mathbf{a}$ and $\mathbf{b}$, we write $|\mathbf{a} - \mathbf{b}|$ to denote the Hamming distance between $\mathbf{a}$ and $\mathbf{b}$. Finally, if $\mathbf{a}$ and $\mathbf{b}$ are solutions of a Boolean formula φ and lie in the same component of $G(\varphi)$, then we write $d_\varphi(\mathbf{a}, \mathbf{b})$ to denote the shortest-path distance between $\mathbf{a}$ and $\mathbf{b}$ in $G(\varphi)$.

5.2.1 Componentwise bijunctive sets of relations

Lemma 25. *Let $\mathcal{S}$ be a set of componentwise bijunctive relations and φ a* CNF($\mathcal{S}$)*-formula. If $\mathbf{a}$ and $\mathbf{b}$ are two solutions of φ that lie in the same component of $G(\varphi)$, then $d_\varphi(\mathbf{a}, \mathbf{b}) = |\mathbf{a} - \mathbf{b}|$.*

Proof. Consider first the special case in which every relation in $\mathcal{S}$ is bijunctive. In this case, φ is equivalent to a 2-CNF formula and so the space of solutions of φ is closed under majority. We show that there is a path in $G(\varphi)$ from $\mathbf{a}$ to $\mathbf{b}$, such that along the path only the assignments on variables with indices from the set $D = \{i : a_i \neq b_i\}$ change. This implies that the shortest path is of length $|D|$ by induction on $|D|$. Consider any path $\mathbf{a} \to \mathbf{u^1} \to \cdots \to \mathbf{u^r} \to \mathbf{b}$ in $G(\varphi)$. We construct another path by replacing $\mathbf{u^i}$ by $\mathbf{v^i} = \mathrm{maj}\ (\mathbf{a}, \mathbf{u^i}, \mathbf{b})$ for $i = 1, \ldots, r$, and removing repetitions. This is a path because for any i $\mathbf{v^i}$ and $\mathbf{v^{i+1}}$ differ in at most one variable. Furthermore, $\mathbf{v^i}$ agrees with $\mathbf{a}$ and $\mathbf{b}$ for every i for which $a_i = b_i$. Therefore, along this path only variables in D are flipped.

For the general case, we show that every component F of $G(\varphi)$ is the solution space of a 2-CNF formula φ'.

Let F be the component of $G(\varphi)$ which contains $\mathbf{a}$ and $\mathbf{b}$. Let $R \in \mathcal{S}$ be a relation with two components, R_1, R_2 each of which are bijunctive. Consider a clause in φ of the form $R(x_1, \ldots, x_k)$. The projection of F onto $x_1, \ldots, x_k$ is itself connected and must satisfy R. Hence it lies within one of the two components R_1, R_2, assume it is R_1. We replace $R(x_1, \ldots, x_k)$ by $R_1(x_1, \ldots, x_k)$. Call this new formula φ_1. $G(\varphi_1)$ consists of all components of $G(\varphi)$ whose projection on $x_1, \ldots, x_k$ lies in R_1. We repeat this for every clause. Finally we are left with a formula φ' over a set of bijunctive relations. Hence φ' is bijunctive and $G(\varphi')$ is a component of $G(\varphi)$. So the claim follows from the bijunctive case. $\square$

Corollary 26. *Let $\mathcal{S}$ be a set of componentwise bijunctive relations. Then*

1. *For every $\varphi \in \mathrm{CNF}(\mathcal{S})$ with n variables, the diameter of each component of $G(\varphi)$ is bounded by n.*
2. ST-CONN$(\mathcal{S})$ *is in* P.
3. CONN$(\mathcal{S})$ *is in* coNP.

Proof. The bound on diameter is an immediate consequence of Lemma 25.

The following algorithm solves ST-CONN$(\mathcal{S})$ given vertices $\mathbf{s}, \mathbf{t} \in G(\varphi)$. Start with $\mathbf{u} = \mathbf{s}$. At each step, find a variable x_i so that $u_i \neq t_i$ and flip it, until we reach $\mathbf{t}$. If at any stage no such variable exists, then declare that $\mathbf{s}$ and $\mathbf{t}$ are not connected. If the $\mathbf{s}$ and $\mathbf{t}$ are disconnected, the algorithm is bound to fail. So assume that they are connected. Correctness is proved by induction on $d = |\mathbf{s} - \mathbf{t}|$. It is clear that the algorithm works when $d = 1$. Assume that the algorithm works for $d - 1$. If s and t are connected and are distance d apart, Lemma 25 implies there is a path of length d between them in $G(\varphi)$. In particular, the algorithm will find a variable x_i to flip. The resulting assignment is at distance $d - 1$ from $\mathbf{t}$, so now we proceed by induction.

Next we prove that $\textsc{Conn}(\mathcal{S}) \in$ coNP. A short certificate of disconnectivity is a pair of assignments $\mathbf{s}$ and $\mathbf{t}$ which are solutions from different components. To verify that they are disconnected it suffices to run the algorithm for $\textsc{st-Conn}$. $\square$

5.2.2 OR-free and NAND-free sets of relations

We consider sets of OR-free relations. Sets of NAND-free relations are handled dually. Define the *coordinate-wise partial order* $\leq$ on Boolean vectors as follows: $\mathbf{a} \leq \mathbf{b}$ if $a_i \leq b_i$, for each i.

Lemma 27. *Let $\mathcal{S}$ be a set of* OR*-free relations and φ a* CNF$(\mathcal{S})$*-formula. Every component of $G(\varphi)$ contains a minimum solution with respect to the coordinate-wise order; moreover, every solution is connected to the minimum solution in the same component via a monotone path.*

Proof. Let's call a satisfying assignment locally minimal, if it has no neighboring satisfying assignments that are smaller than it. We will show that there is exactly one such assignment in each component of $G(\varphi)$.

Suppose there are two distinct locally minimal assignments $\mathbf{u}$ and $\mathbf{u}'$ in some component of $G(\varphi)$. Consider the path between them where the maximum Hamming weight of assignments on the path is minimized. If there are many such paths, pick one where the smallest number of assignments have the maximum Hamming weight. Denote this path by $\mathbf{u} = \mathbf{u^1} \to \mathbf{u^2} \to \cdots \to \mathbf{u^r} = \mathbf{u}'$. Let $\mathbf{u^i}$ be an assignment of largest Hamming weight in the path. Then $\mathbf{u^i} \neq \mathbf{u}$ and $\mathbf{u^i} \neq \mathbf{u}'$, since $\mathbf{u}$ and $\mathbf{u}'$ are locally minimal. The assignments $\mathbf{u^{i-1}}$ and $\mathbf{u^{i+1}}$ differ in exactly 2 variables, say, in x_1 and x_2. So $\{u_1^{i-1}u_2^{i-1},\ u_1^i u_2^i,\ u_1^{i+1}u_2^{i+1}\} = \{01, 11, 10\}$. Let $\hat{\mathbf{u}}$ be such that $\hat{u}_1 = \hat{u}_2 = 0$, and $\hat{u}_i = u_i$ for $i > 2$. If $\hat{\mathbf{u}}$ is a solution, then the path $\mathbf{u^1} \to \mathbf{u^2} \to \cdots \to \mathbf{u^i} \to \hat{\mathbf{u}} \to \mathbf{u^{i+1}} \to \cdots \to \mathbf{u^r}$ contradicts the way we chose the original path. Therefore, $\hat{\mathbf{u}}$ is not a solution. This means that there is a clause that is violated by it, but is satisfied by $\mathbf{u^{i-1}}$, $\mathbf{u^i}$, and $\mathbf{u^{i+1}}$. So the relation corresponding to that clause is not OR-free, which is a contradiction.

The unique locally minimal solution in a component is its minimum solution, because starting from any other assignment in the component, it is possible to keep moving to neighbors that are smaller, and the only time it becomes impossible to find such a neighbor is when the locally minimal solution is reached. Therefore, there is a monotone path from any satisfying assignment to the minimum in that component. $\square$

Corollary 28. *Let $\mathcal{S}$ be a set of* OR*-free relations. Then*

1. *For every $\varphi \in \mathrm{CNF}(\mathcal{S})$ with n variables, the diameter of each component of $G(\varphi)$ is bounded by $2n$.*

2. $\textsc{st-Conn}(\mathcal{S})$ *is in* P.

3. $\textsc{Conn}(\mathcal{S})$ *is in* coNP.

Proof. Given solutions $\mathbf{s}$ and $\mathbf{t}$ in the same component of $G(\varphi)$, there is a monotone path from each to the minimal solution $\mathbf{u}$ in the component. This gives a path from $\mathbf{s}$ to $\mathbf{t}$ of length at most $2n$. To check if $\mathbf{s}$ and $\mathbf{t}$ are connected, we just check that the minimal assignments reached from $\mathbf{s}$ and $\mathbf{t}$ are the same. ☐

5.2.3 The complexity of $\textsc{Conn}(\mathcal{S})$ for tight sets of relations

We can further specify the complexity of $\textsc{Conn}(\mathcal{S})$ for the tight cases which are not Schaefer, using a result of Juban [Jub99].

Lemma 29. *For $\mathcal{S}$ tight, but not Schaefer,* $\textsc{Conn}(\mathcal{S})$ *is* coNP-*complete.*

Proof. The problem $\textsc{Another-Sat}(\mathcal{S})$ is: given a formula φ in $\mathrm{CNF}(\mathcal{S})$ and a solution $\mathbf{s}$, does there exist a solution $\mathbf{t} \neq \mathbf{s}$? Juban ([Jub99], Theorem 2) shows that if $\mathcal{S}$ is not Schaefer, then $\textsc{Another-Sat}$ is NP-complete. He also shows (Corollary 1) that if $\mathcal{S}$ is not Schaefer, then the relation $x \neq y$ is expressible from $\mathcal{S}$ through substitutions.

Since $\mathcal{S}$ is not Schaefer, $\textsc{Another-Sat}(\mathcal{S})$ is NP-complete. Let $\varphi, \mathbf{s}$ be an instance of $\textsc{Another-Sat}$ on variables $x_1, \ldots, x_n$. We define a $\mathrm{CNF}(\mathcal{S})$ formula ψ on $x_1, \ldots, x_n, y_1, \ldots, y_n$ as

$$\psi(x_1, \ldots, x_n, y_1, \ldots, y_n) = \varphi(x_1, \ldots, x_n) \wedge_i (x_i \neq y_i)$$

It is easy to see that $G(\psi)$ is connected if and only if $\mathbf{s}$ is the unique solution to φ. ☐

Further we can show that $\textsc{Conn}(\mathcal{S})$ is in P if $\mathcal{S}$ is affine or bijunctive. Thus the only tight cases for which $\textsc{Conn}(\mathcal{S})$ is not known to be coNP-complete or in P are Horn and dual-Horn. We conjecture that these problems are in P.

5.3 The hard case of the dichotomy: non-tight sets of relations

We will show that all non-tight sets of relations lead to solution graphs that have identical properties in a natural sense that is captured in the notion of faithful expressibility. We define this notion in Section 5.3.1, and prove the Faithful Expressibility Theorem in Section 5.3.2. This theorem implies that the complexity of the connectivity questions for all such sets is the same, and the possible diameter of components of the solution graph is also related polynomially. In section 5.3.3 we will prove that for 3-CNF formulas the connectivity questions are PSPACE-complete, and the diameter can be exponential. This fact together with the Faithful Expressibility Theorem implies the hard side of all of our dichotomy results.

5.3.1 Faithful expressibility

In his dichotomy theorem, Schaefer [Sch78] used the following notion of expressibility: a relation R is *expressible from* a set $\mathcal{S}$ of relations if there is a CNF($\mathcal{S}$)-formula φ so that $R(\mathbf{x}) \equiv \exists \mathbf{y}\ \varphi(\mathbf{x}, \mathbf{y})$. This notion, is not sufficient for our purposes. Instead, we introduce a more delicate notion, which we call *faithful expressibility*. Intuitively, we view the relation R as a subgraph of the hypercube, rather than just a subset, and require that this graph structure be also captured by the formula φ.

Definition 16. *A relation R is* faithfully expressible *from a set of relations $\mathcal{S}$ if there is a* CNF($\mathcal{S}$)*-formula φ such that the following condition hold:*

1. $R = \{\mathbf{a} :\ \exists \mathbf{y}\ \varphi(\mathbf{a}, \mathbf{y})\}$;
2. *For every $\mathbf{a} \in R$, the graph $G(\varphi(\mathbf{a}, \mathbf{y}))$ is connected;*
3. *For $\mathbf{a}, \mathbf{b} \in R$ with $|\mathbf{a} - \mathbf{b}| = 1$, there exists $\mathbf{w}$ such that $(\mathbf{a}, \mathbf{w})$ and $(\mathbf{b}, \mathbf{w})$ are solutions of φ.*

For $\mathbf{a} \in R$, the *witnesses* of $\mathbf{a}$ are the $\mathbf{y}$'s such that $\varphi(\mathbf{a}, \mathbf{y})$ is true. The last two conditions say that the witnesses of $\mathbf{a} \in R$ are connected, and that neighboring $\mathbf{a}, \mathbf{b} \in R$ have a common witness. This allows us to simulate an edge $(\mathbf{a}, \mathbf{b})$ in $G(R)$ by a path in $G(\varphi)$, and thus relate the connectivity properties of the solution spaces. There is however, a price to pay: it is much harder to come up with formulas that faithfully express a relation R. An example is when $\mathcal{S}$ is the set of all paths of length 4 in $\{0, 1\}^3$, a set that plays a crucial role in our proof. While 3-SAT relations are easily expressible from $\mathcal{S}$ in Schaefer's sense, the CNF($\mathcal{S}$)-formulas that faithfully express 3-SAT relations are fairly complicated and have a large witness space.

Lemma 30. *Let $\mathcal{S}$ and $\mathcal{S}'$ be sets of relations such that every $R \in \mathcal{S}'$ is faithfully expressible from $\mathcal{S}$. Given a* CNF($\mathcal{S}'$)*-formula $\psi(\mathbf{x})$, one can efficiently construct a* CNF($\mathcal{S}$)*-formula $\varphi(\mathbf{x}, \mathbf{y})$ such that:*

1. *$\psi(\mathbf{x}) \equiv \exists \mathbf{y}\ \varphi(\mathbf{x}, \mathbf{y})$;*

2. *if $(\mathbf{s}, \mathbf{w^s}), (\mathbf{t}, \mathbf{w^t}) \in \varphi$ are connected in $G(\varphi)$ by a path of length d, then there is a path from $\mathbf{s}$ to $\mathbf{t}$ in $G(\psi)$ of length at most d;*

3. *If $\mathbf{s}, \mathbf{t} \in \psi$ are connected in $G(\psi)$, then for every witness $\mathbf{w^s}$ of $\mathbf{s}$, and every witness $\mathbf{w^t}$ of $\mathbf{t}$, there is a path from $(\mathbf{s}, \mathbf{w^s})$ to $(\mathbf{t}, \mathbf{w^t})$ in $G(\varphi)$.*

Proof. Suppose ψ is a formula on n variables that consists of m clauses $C_1, \ldots, C_m$. For clause C_j, assume that the set of variables is $V_j \subseteq [n]$, and that it involves relation $R_j \in \mathcal{S}$. Thus, $\psi(\mathbf{x})$ is $\wedge_{j=1}^{m} R_j(\mathbf{x}_{V_j})$. Let φ_j be the faithful expression for R_j from $\mathcal{S}'$, so that $R_j(\mathbf{x}_{V_j}) \equiv \exists \mathbf{y}_j\ \varphi_j(\mathbf{x}_{V_j}, \mathbf{y}_j)$. Let $\mathbf{y}$ be the vector $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$ and let $\varphi(\mathbf{x}, \mathbf{y})$ be the formula $\wedge_{j=1}^{m} \varphi_j(\mathbf{x}_{V_j}, \mathbf{y}_j)$. Then $\psi(\mathbf{x}) \equiv \exists \mathbf{y}\ \varphi(\mathbf{x}, \mathbf{y})$.

Statement (2) follows from (1) by projection of the path on the coordinates of $\mathbf{x}$. For statement (3), consider $\mathbf{s}, \mathbf{t} \in \psi$ that are connected in $G(\psi)$ via a path $\mathbf{s} = \mathbf{u^0} \to \mathbf{u^1} \to \cdots \to \mathbf{u^r} = \mathbf{t}$. For every $\mathbf{u^i}, \mathbf{u^{i+1}}$, and clause C_j, there exists an assignment $\mathbf{w^i}_j$ to $\mathbf{y}_j$ such that both $(\mathbf{u^i}_{V_j}, \mathbf{w^i}_j)$ and $(\mathbf{u^{i+1}}_{V_j}, \mathbf{w^i}_j)$ are solutions of φ_j, by condition (2) of faithful expressibility. Thus $(\mathbf{u^i}, \mathbf{w^i})$ and $(\mathbf{u^{i+1}}, \mathbf{w^i})$ are both solutions of φ, where $\mathbf{w^i} = (\mathbf{w^i}_1, \ldots \mathbf{w^i}_m)$. Further, for every $\mathbf{u^i}$, the space of solutions of $\varphi(\mathbf{u^i}, \mathbf{y})$ is the product space of the solutions of $\varphi_j(\mathbf{u^i}_{V_j}, \mathbf{y}_j)$ over $j = 1, \ldots, m$. Since these are all connected by condition (3) of faithful expressibility, $G(\varphi(\mathbf{u^i}, \mathbf{y}))$ is connected. The following describes a path from $(\mathbf{s}, \mathbf{w^s})$ to $(\mathbf{t}, \mathbf{w^t})$ in $G(\varphi)$: $(\mathbf{s}, \mathbf{w^s}) \rightsquigarrow (\mathbf{s}, \mathbf{w^0}) \to (\mathbf{u^1}, \mathbf{w^0}) \rightsquigarrow (\mathbf{u^1}, \mathbf{w^1}) \to \cdots \rightsquigarrow (\mathbf{u^{r-1}}, \mathbf{w^{r-1}}) \to (\mathbf{t}, \mathbf{w^{r-1}}) \rightsquigarrow (\mathbf{t}, \mathbf{w^t})$. Here $\rightsquigarrow$ indicates a path in $G(\varphi(\mathbf{u^i}, \mathbf{y}))$. $\square$

Corollary 31. *Suppose $\mathcal{S}$ and $\mathcal{S}'$ are sets of relations such that every $R \in \mathcal{S}'$ is faithfully expressible from $\mathcal{S}$.*

1. *There are polynomial time reductions from* CONN$(\mathcal{S}')$ *to* CONN$(\mathcal{S})$, *and from* ST-CONN$(\mathcal{S}')$ *to* ST-CONN$(\mathcal{S})$.

2. *Given a* CNF$(\mathcal{S}')$*-formula $\psi(\mathbf{x})$ with m clauses, one can efficiently construct a* CNF$(\mathcal{S})$*-formula $\varphi(\mathbf{x}, \mathbf{y})$ such that the length of $\mathbf{y}$ is $O(m)$ and the diameter of the solution space does not decrease.*

5.3.2 Faithfully expressing a relation from a non-tight set of relations

Observe that faithful expressibility is easily shown to be a transitive property. To prove Theorem 21, first we will show that we can faithfully express the 3-clause relations from the relations in $\mathcal{S}$. Secondly, it is easy to show that any other relation can be expressed faithfully from 3-clauses.

Expressing 3-clauses from non-tight sets of relations

First we define what we mean by 3-clauses. If $k \geq 2$, then a *k-clause* is a disjunction of k variables or negated variables. For $0 \leq i \leq k$, let D_i be the set of all satisfying truth assignments of the k-clause whose first i literals are negated, and let $\mathcal{S}_k = \{D_0, D_1, \ldots, D_k\}$. Thus, CNF($\mathcal{S}_k$) is the collection of k-CNF formulas.

Lemma 32. *If set $\mathcal{S}$ of relations is not tight, $\mathcal{S}_3$ is faithfully expressible from $\mathcal{S}$.*

Proof. First, observe that all 2-clauses are faithfully expressible from $\mathcal{S}$. There exists $R \in \mathcal{S}$ which is not OR-free, so we can express $(x_1 \vee x_2)$ by substituting constants in R. Similarly, we can express $(\bar{x}_1 \vee \bar{x}_2)$ using a relation that is not NAND-free. The last 2-clause $(x_1 \vee \bar{x}_2)$ can be obtained from OR and NAND by a technique that corresponds to reverse resolution. $(x_1 \vee \bar{x}_2) = \exists y\ (x_1 \vee y) \wedge (\bar{y} \vee \bar{x}_2)$. It is easy to see that this gives a faithful expression. From here onwards we assume that $\mathcal{S}$ contains all 2-clauses. The proof now proceeds in four steps. First, we will express a relation in which there exist two elements that are at graph distance larger than their Hamming distance. Second, we will express a relation that is just a single path between such elements. Third, we will express a relation which is a path of length 4 between elements at Hamming distance 2. Finally, we will express the 3-clauses.

Step 1. *Faithfully expressing a relation in which some distance expands.*

For a relation R, we say that the distance between $\mathbf{a}$ and $\mathbf{b}$ *expands* if $\mathbf{a}$ and $\mathbf{b}$ are connected in $G(R)$, but $d_R(\mathbf{a}, \mathbf{b}) > |\mathbf{a} - \mathbf{b}|$. By Lemma 25 no distance expands in componentwise bijunctive relations. This property also holds for the relation $R_{\text{NAE}} = \{0,1\}^3 \setminus \{000, 111\}$, which is not componentwise bijunctive. However, we show that if R is not componentwise bijunctive, then, by adding 2-clauses, we can faithfully express a relation Q in which some distance expands. For instance, when $R = R_{\text{NAE}}$, then we can take $Q(x_1, x_2, x_3) = R_{\text{NAE}}(x_1, x_2, x_3) \wedge (\bar{x_1} \vee \bar{x}_3)$. The distance between $\mathbf{a} = 100$ and $\mathbf{b} = 001$ in Q expands. Similarly, in the general construction,

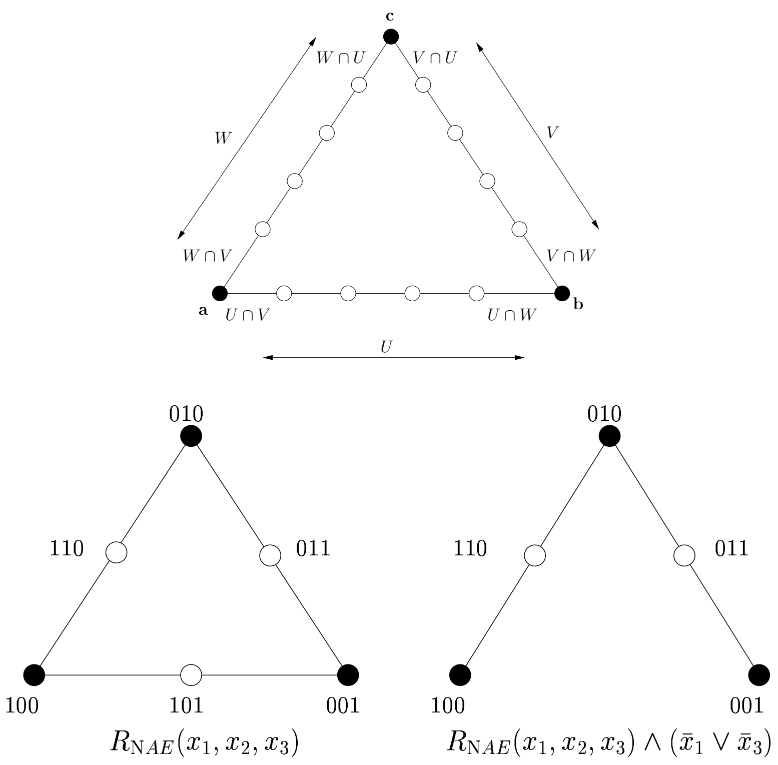


Figure 5.2: Proof of Step 1 of Lemma 32, and an example.

we identify $\mathbf{a}$ and $\mathbf{b}$ on a cycle, and add 2-clauses that eliminate all the vertices along the shorter arc between $\mathbf{a}$ and $\mathbf{b}$.

Since $\mathcal{S}$ is not tight, it contains a relation R which is not componentwise bijunctive. If R contains $\mathbf{a}, \mathbf{b}$ where the distance between them expands, we are done. So assume that for all $\mathbf{a}, \mathbf{b} \in G(R)$, $d_R(\mathbf{a}, \mathbf{b}) = |\mathbf{a} - \mathbf{b}|$. Since R is not componentwise bijunctive, there exists a triple of assignments $\mathbf{a}, \mathbf{b}, \mathbf{c}$ lying in the same component such that $\text{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ is not in that component (which also easily implies it is not in R). Choose the triple such that the sum of pairwise distances $d_R(\mathbf{a}, \mathbf{b}) + d_R(\mathbf{b}, \mathbf{c}) + d_R(\mathbf{c}, \mathbf{a})$ is minimized. Let $U = \{i | a_i \neq b_i\}$, $V = \{i | b_i \neq c_i\}$, and $W = \{i | c_i \neq a_i\}$. Since $d_R(\mathbf{a}, \mathbf{b}) = |\mathbf{a} - \mathbf{b}|$, a shortest path does not flip variables outside of U, and each variable in U is flipped exactly once. The same holds for V and W. We note some useful properties of the sets U, V, W.

1. *Every index $i \in U \cup V \cup W$ occurs in exactly two of U, V, W.*

   Consider going by a shortest path from $\mathbf{a}$ to $\mathbf{b}$ to $\mathbf{c}$ and back to $\mathbf{a}$. Every $i \in U \cup V \cup W$ is seen an even number of times along this path since we return to $\mathbf{a}$. It is seen at least once, and at most thrice, so in fact it occurs twice.

2. *Every pairwise intersection $U \cap V, V \cap W$ and $W \cap U$ is non-empty.*

   Suppose the sets U and V are disjoint. From Property 1, we must have $W = U \cup V$. But then it is easy to see that $\mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c}) = \mathbf{b}$ which is in R. This contradicts the choice of $\mathbf{a}, \mathbf{b}, \mathbf{c}$.

3. *The sets $U \cap V$ and $U \cap W$ partition the set U.*

   By Property 1, each index of U occurs in one of V and W as well. Also since no index occurs in all three sets U, V, W this is in fact a disjoint partition.

4. *For each index $i \in U \cap W$, it holds that $\mathbf{a} \oplus \mathbf{e}_i \notin R$.*

   Assume for the sake of contradiction that $\mathbf{a}' = \mathbf{a} \oplus \mathbf{e}_i \in R$. Since $i \in U \cap W$ we have simultaneously moved closer to both $\mathbf{b}$ and $\mathbf{c}$. Hence we have $d_R(\mathbf{a}', \mathbf{b}) + d_R(\mathbf{b}, \mathbf{c}) + d_R(\mathbf{c}, \mathbf{a}') < d_R(\mathbf{a}, \mathbf{b}) + d_R(\mathbf{b}, \mathbf{c}) + d_R(\mathbf{c}, \mathbf{a})$. Also $\mathrm{maj}(\mathbf{a}', \mathbf{b}, \mathbf{c}) = \mathrm{maj}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \notin R$. But this contradicts our choice of $\mathbf{a}, \mathbf{b}, \mathbf{c}$.

Property 4 implies that the shortest paths to $\mathbf{b}$ and $\mathbf{c}$ diverge at $\mathbf{a}$, since for any shortest path to $\mathbf{b}$ the first variable flipped is from $U \cap V$ whereas for a shortest path to $\mathbf{c}$ it is from $W \cap V$. Similar statements hold for the vertices $\mathbf{b}$ and $\mathbf{c}$. Thus along the shortest path from $\mathbf{a}$ to $\mathbf{b}$ the first bit flipped is from $U \cap V$ and the last bit flipped is from $U \cap W$. On the other hand, if we go from $\mathbf{a}$ to $\mathbf{c}$ and then to $\mathbf{b}$, all the bits from $U \cap W$ are flipped before the bits from $U \cap V$. We use this crucially to define Q. We will add a set of 2-clauses that enforce the following rule on paths starting at $\mathbf{a}$: *Flip variables from $U \cap W$ before variables from $U \cap V$.* This will eliminate all shortest paths from $\mathbf{a}$ to $\mathbf{b}$ since they begin by flipping a variable in $U \cap V$ and end with $U \cap W$. The paths from $\mathbf{a}$ to $\mathbf{b}$ via $\mathbf{c}$ survive since they flip $U \cap W$ while going from $\mathbf{a}$ to $\mathbf{c}$ and $U \cap V$ while going from $\mathbf{c}$ to $\mathbf{b}$. However all remaining paths have length at least $|\mathbf{a} - \mathbf{b}| + 2$ since they flip twice some variables not in U.

Take all pairs of indices $\{(i, j) | i \in U \cap W, j \in U \cap V\}$. The following conditions hold from the definition of U, V, W: $a_i = \bar{c}_i = \bar{b}_i$ and $a_j = c_j = \bar{b}_j$. Add the 2-clause C_{ij} asserting that the pair of variables $x_i x_j$ must take values in $\{a_i a_j, c_i c_j, b_i b_j\} = \{a_i a_j, \bar{a}_i a_j, \bar{a}_i \bar{a}_j\}$. The new relation is $Q = R \wedge_{i,j} C_{ij}$. Note that $Q \subset R$. We verify that the distance between $\mathbf{a}$ and $\mathbf{b}$ in Q expands. It is easy to see that for any $j \in U$, the assignment $\mathbf{a} \oplus \mathbf{e}_j \notin Q$. Hence there are no shortest paths left from $\mathbf{a}$ to $\mathbf{b}$. On the other hand, it is easy to see that $\mathbf{a}$ and $\mathbf{b}$ are still connected, since the vertex $\mathbf{c}$ is still reachable from both.

Step 2. *Isolating a pair of assignments whose distance expands.*

The relation Q obtained in Step 1 may have several disconnected components. This *cleanup* step isolates a single pair of assignments whose distance expands. By adding 2-clauses, we show that one can express a path of length $r+2$ between assignments at distance r.

Take $\mathbf{a}, \mathbf{b} \in Q$ whose distance expands in Q and $d_Q(\mathbf{a}, \mathbf{b})$ is minimized. Let $U = \{i : a_i \neq b_i\}$, and $|U| = r$. Shortest paths between $\mathbf{a}$ and $\mathbf{b}$ have certain useful properties:

1. *Each shortest path flips every variable from U exactly once.*
   Observe that each index $j \in U$ is flipped an odd number of times along any path from $\mathbf{a}$ to $\mathbf{b}$. Suppose it is flipped thrice along a shortest path. Starting at $\mathbf{a}$ and going along this path, let $\mathbf{b}'$ be the assignment reached after flipping j twice. Then the distance between $\mathbf{a}$ and $\mathbf{b}'$ expands, since j is flipped twice along a shortest path between them in Q. Also $d_Q(\mathbf{a}, \mathbf{b}') < d_Q(\mathbf{a}, \mathbf{b})$, contradicting the choice of $\mathbf{a}$ and $\mathbf{b}$.

2. *Every shortest path flips exactly one variable $i \notin U$.*
   Since the distance between $\mathbf{a}$ and $\mathbf{b}$ expands, every shortest path must flip some variable $i \notin U$. Suppose it flips more than one such variable. Since $\mathbf{a}$ and $\mathbf{b}$ agree on these variables, each of them is flipped an even number of times. Let i be the first variable to be flipped twice. Let $\mathbf{b}'$ be the assignment reached after flipping i the second time. It is easy to verify that the distance between $\mathbf{a}$ and $\mathbf{b}'$ also expands, but $d_Q(\mathbf{a}, \mathbf{b}') < d_Q(\mathbf{a}, \mathbf{b})$.

3. *The variable $i \notin U$ is the first and last variable to be flipped along the path.* Assume the first variable flipped is not i. Let $\mathbf{a}'$ be the assignment reached along the path before we flip i the first time. Then $d_Q(\mathbf{a}', \mathbf{b}) < d_Q(\mathbf{a}, \mathbf{b})$. The distance between $\mathbf{a}'$ and $\mathbf{b}$ expands since the shortest path between them flips the variables i twice. This contradicts the choice of $\mathbf{a}$ and $\mathbf{b}$. Assume $j \in U$ is flipped twice. Then as before we get a pair $\mathbf{a}', \mathbf{b}'$ that contradict the choice of $\mathbf{a}, \mathbf{b}$.

Every shortest path between $\mathbf{a}$ and $\mathbf{b}$ has the following structure: first a variable $i \notin U$ is flipped to $\bar{a}_i$, then the variables from U are flipped in some order, finally the variable i is flipped back to a_i.

Different shortest paths may vary in the choice of $i \notin U$ in the first step and in the order in which the variables from U are flipped. Fix one such path $T \subseteq Q$. Assume that $U = \{1, \ldots, r\}$ and the variables are flipped in this order, and the additional variable flipped twice is $r+1$. Denote the path by $\mathbf{a} \rightarrow \mathbf{u^0} \rightarrow \mathbf{u^1} \rightarrow \cdots \rightarrow \mathbf{u^r} \rightarrow \mathbf{b}$. Next we prove that we cannot flip the $r+1^{th}$ variable at an intermediate vertex along the path.

4 *For* $1 \leq j \leq r-1$ *the assignment* $\mathbf{u^j} \oplus \mathbf{e_{r+1}} \notin Q$.
Suppose that for some j, we have $\mathbf{c} = \mathbf{u^j} \oplus \mathbf{e_{r+1}} \in Q$. Then $\mathbf{c}$ differs from $\mathbf{a}$ on $\{1, \ldots, i\}$ and from $\mathbf{b}$ on $\{i+1, \ldots, r\}$. The distance from $\mathbf{c}$ to at least one of $\mathbf{a}$ or $\mathbf{b}$ must expand, else we get a path from $\mathbf{a}$ to $\mathbf{b}$ through $\mathbf{c}$ of length $|\mathbf{a} - \mathbf{b}|$ which contradicts the fact that this distance expands. However $d_Q(\mathbf{a}, \mathbf{c})$ and $d_Q(\mathbf{b}, \mathbf{c})$ are strictly less than $d_Q(\mathbf{a}, \mathbf{b})$ so we get a contradiction to the choice of $\mathbf{a}, \mathbf{b}$.

We now construct the path of length $r+2$. For all $i \geq r+2$ we set $x_i = a_i$ to get a relation on $r+1$ variables. Note that $\mathbf{b} = \bar{a}_1 \ldots \bar{a}_r a_{r+1}$. Take $i < j \in U$. Along the path T the variable i is flipped before j so the variables $x_i x_j$ take one of three values $\{a_i a_j, \bar{a}_i a_j, \bar{a}_i \bar{a}_j\}$. So we add a 2-clause C_{ij} that requires $x_i x_j$ to take one of these values and take $T = Q \wedge_{i,j} C_{ij}$. Clearly, every assignment along the path lies in T. We claim that these are the only solutions. To show this, take an arbitrary assignment $\mathbf{c}$ satisfying the added constraints. If for some $i < j \leq r$ we have $c_i = a_i$ but $c_j = \bar{a}_j$, this would violate C_{ij}. Hence the first r variables of $\mathbf{c}$ are of the form $\bar{a}_1 \ldots \bar{a}_i a_{i+1} \ldots a_r$ for $0 \leq i \leq r$. If $c_{r+1} = \bar{a}_{r+1}$ then $\mathbf{c} = \mathbf{u}^i$. If $c_{r+1} = a_{r+1}$ then $\mathbf{c} = \mathbf{u}^i \oplus \mathbf{e}_{r+1}$. By property 4 above, such a vector satisfies Q if and only if $i = 0$ or $i = r$, which correspond to $\mathbf{c} = \mathbf{a}$ and $\mathbf{c} = \mathbf{b}$ respectively.

Step 3. *Faithfully expressing paths of length* 4.

Let $\mathcal{P}$ denote the set of all ternary relations whose graph is a path of length 4 between two assignments at Hamming distance 2. Up to permutations of coordinates, there are 6 such relations. Each of them is the conjunction of a 3-clause and a 2-clause. For instance, the relation $M = \{100, 110, 010, 011, 001\}$ can be written as of $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3)$. (It is named so, because its graph looks like the letter 'M' on the cube.) These relations are "minimal" examples of relations that are not componentwise bijunctive. By projecting out intermediate variables from the path T obtained in Step 2, we faithfully express one of the relations in $\mathcal{P}$. We faithfully express other relations in $\mathcal{P}$ using this relation.

We write all relations in $\mathcal{P}$ in terms of $M(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3)$, by negating variables. For example, $M(\bar{x}_1, x_2, x_3) = (\bar{x}_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_3) = \{000, 010, 110, 111, 101\}$.

Define the relation $P(x_1, x_{r+1}, x_2) = \exists x_3 \ldots x_r \, T(x_1, \ldots, x_{r+1})$. Table 5.3.2 listing all tuples in P and their witnesses, shows that the conditions for faithful expressibility are satisfied, and $P \in \mathcal{P}$.

x_1, x_2, x_{r+1}	$x_3, \ldots, x_r$
$a_1 a_2 a_{r+1}$	$a_3 \ldots a_r$
$a_1 a_2 \bar{a}_{r+1}$	$a_3 \ldots a_r$
$\bar{a}_1 a_2 \bar{a}_{r+1}$	$a_3 \ldots a_r$
$\bar{a}_1 \bar{a}_2 \bar{a}_{r+1}$	$a_3 \ldots a_k,\ \bar{a}_3 a_4 \ldots a_r,\ \bar{a}_3 \bar{a}_4 a_5 \ldots a_r\ \ldots \bar{a}_3 \bar{a}_4 \ldots \bar{a}_r$
$\bar{a}_1 \bar{a}_2 a_{r+1}$	$\bar{a}_3 \bar{a}_4 \ldots \bar{a}_r$

Table 5.2: Proof of Step 3 of Lemma 32

Let $P(x_1, x_2, x_3) = M(l_1, l_2, l_3)$, where l_i is one of $\{x_i, \bar{x}_i\}$. We can now use P and 2-clauses to express every other relation in $\mathcal{P}$. Given $M(l_1, l_2, l_3)$ every relation in $\mathcal{P}$ can be obtained by negating some subset of the variables. Hence it suffices to show that we can express faithfully $M(\bar{l}_1, l_2, l_3)$ and $M(l_1, \bar{l}_2, l_3)$ (M is symmetric in x_1 and x_3). In the following let λ denote one of the literals $\{y, \bar{y}\}$, such that it is $\bar{y}$ if and only if l_1 is $\bar{x}_1$.

$$\begin{aligned}
M(\bar{l}_1, l_2, l_3) &= (\bar{l}_1 \vee l_2 \vee l_3) \wedge (l_1 \vee \bar{l}_3) \\
&= \exists y\ (\bar{l}_1 \vee \bar{\lambda}) \wedge (\lambda \vee l_2 \vee l_3) \wedge (l_1 \vee \bar{l}_3) \\
&= \exists y\ (\bar{l}_1 \vee \bar{\lambda}) \wedge (\lambda \vee l_2 \vee l_3) \wedge (l_1 \vee \bar{l}_3) \wedge (\bar{\lambda} \vee \bar{l}_3) \\
&= \exists y\ (\bar{l}_1 \vee \bar{\lambda}) \wedge (l_1 \vee \bar{l}_3) \wedge M(\lambda, l_2, l_3) \\
&= \exists y\ (\bar{l}_1 \vee \bar{\lambda}) \wedge (l_1 \vee \bar{l}_3) \wedge P(y, x_2, x_3)
\end{aligned}$$

For the next expression let λ denote one of the literals $\{y, \bar{y}\}$, such that it is negated if and only if l_2 is $\bar{x}_2$.

$$\begin{aligned}
M(l_1, \bar{l}_2, l_3) &= (l_1 \vee \bar{l}_2 \vee l_3) \wedge (\bar{l}_1 \vee \bar{l}_3) \\
&= \exists y\ (l_1 \vee l_3 \vee \lambda) \wedge (\bar{\lambda} \vee \bar{l}_2) \wedge (\bar{l}_1 \vee \bar{l}_3) \\
&= \exists y\ (\bar{\lambda} \vee \bar{l}_2) \wedge M(l_1, \lambda, l_3) \\
&= \exists y\ (\bar{\lambda} \vee \bar{l}_2) \wedge P(x_1, y, x_3)
\end{aligned}$$

The above expressions are both based on resolution and it is easy to check that they satisfy the properties of faithful expressibility.

Step 4. *Faithfully expressing $\mathcal{S}_3$.*

We faithfully express $(x_1 \vee x_2 \vee x_3)$ from M using a formula derived from a gadget in [HD02]. This gadget expresses $(x_1 \vee x_2 \vee x_3)$ in terms of "Protected OR", which corresponds to our relation M.

$x_1x_2x_3$	$y_1 \dots y_5$			
111	00011, 00111, 00110, 00100, 01100, 01101,	01001, 11001, 11000,	10000,	10010, 10011
110		01001, 11001, 11000,	10000	
100			10000	
101	00011, 00111, 00110, 00100,		10000,	10010, 10011
001	00011, 00111, 00110, 00100			
011	00011, 00111, 00110, 00100, 01100, 01101,	01001		
010		01001		

Table 5.3: Proof of Step 4 of Lemma 32

$$\begin{aligned}(x_1 \vee x_2 \vee x_3) \quad &= \quad \exists y_1 \dots y_5 \, (x_1 \vee \bar{y}_1) \wedge (x_2 \vee \bar{y}_2) \wedge (x_3 \vee \bar{y}_3) \wedge (x_3 \vee \bar{y}_4) \\ &\wedge M(y_1, y_5, y_3) \wedge M(y_2, \bar{y}_5, y_4) \end{aligned} \tag{5.1}$$

Table 5.3.2 shows that the conditions for faithful expressibility are satisfied.

From the relation $(x_1 \vee x_2 \vee x_3)$ we derive the other 3-clauses by reverse resolution, for instance

$$(\bar{x}_1 \vee x_2 \vee x_3) = \exists y \, (\bar{x}_1 \vee \bar{y}) \wedge (y \vee x_2 \vee x_3)$$

□

Expressing a relation faithfully from $\mathcal{S}_3$

Lemma 33. *Let $R \subseteq \{0,1\}^k$ be any relation of arity $k \geq 1$. R is faithfully expressible from $\mathcal{S}_3$.*

Proof. If $k \leq 3$ then R can be expressed as a formula in CNF$(\mathcal{S}_3)$ with constants, without introducing witness variables. This kind of expression is always faithful.

If $k \geq 4$ then R can be expressed as a formula in CNF$(\mathcal{S}_k)$, without witnesses (i.e. faithfully). We will show that every k-clause can be expressed faithfully from $\mathcal{S}_{k-1}$. Then, by induction, it can be expressed faithfully from $\mathcal{S}_3$. For simplicity we express a k-clause corresponding to the relation D_0. The remaining relations are expressed equivalently. We express D_0 in a way that is standard in other complexity reductions, and turns out to be faithful:

$$(x_1 \vee x_2 \vee \cdots \vee x_k) = \exists y \, (x_1 \vee x_2 \vee y) \wedge (\bar{y} \vee x_3 \vee \cdots \vee x_k).$$

This is the reverse operation of resolution. For any satisfying assignment for $\mathbf{x}$, its witness space is either $\{0\}$, $\{1\}$ or $\{0,1\}$, so in all cases it is connected. Furthermore, the only way two neighboring satisfying assignments for x can have no common witness is if one of them has witness set $\{0\}$, and the other one has witness set $\{1\}$. This implies that the first one has $(x_3, \ldots, x_k) = (0, \ldots, 0)$, and the other one has $(x_1, x_2) = (0, 0)$, thus they differ in the assignments of at least two variables: one from $\{x_1, x_2\}$ and one from $\{x_3, \ldots, x_k\}$. In that case they cannot be neighboring assignments. Therefore all requirements of faithful expressibility are satisfied. $\square$

5.3.3 Hardness results for 3-CNF formulas

We show that 3-CNF formulas can have exponential diameter, by inductively constructing a path of length at least $2^{\frac{n}{2}}$ on n variables and then identifying it with the solution space of a 3-CNF formula with $O(n^2)$ clauses.

Lemma 34. *For n even, there is a 3-*CNF *formula φ_n with n variables and $O(n^2)$ clauses, such that $G(\varphi_n)$ is a path of length greater than $2^{\frac{n}{2}}$.*

Proof. The construction is in two steps: we first exhibit an induced subgraph G_n of the n dimensional hypercube with large diameter. We then construct a 3-CNF formula φ_n so that $G_n = G(\varphi_n)$.

The graph G_n is a path of length $2^{\frac{n}{2}}$. We construct it using induction. For $n = 2$, we take $V(G_2) = \{(0,0), (0,1), (1,1)\}$ which has diameter 2. Assume that we have constructed G_{n-2} with $2^{\frac{n-2}{2}}$ vertices, and with distinguished vertices $\mathbf{s_{n-2}}, \mathbf{t_{n-2}}$ such that the shortest path from $\mathbf{s}$ to $\mathbf{t}$ in G_{n-2} has length $2^{\frac{n-2}{2}}$. We now describe the set $V(G_n)$. For each vertex $\mathbf{v} \in V(G_{n-2})$, $V(G_n)$ contains two vertices $(\mathbf{v}, 0, 0)$ and $(\mathbf{v}, 1, 1)$. Note that the subgraph induced by these vertices alone consists of two disconnected copies of G_{n-2}. To connect these two components, we add the vertex $\mathbf{m} = (\mathbf{t}, 0, 1)$ (which is connected to $(\mathbf{t}, 0, 0)$ and $(\mathbf{t}, 1, 1)$ in the induced subgraph). Note that the resulting graph G_n is connected, but any path from $(\mathbf{u}, 0, 0)$ to $(\mathbf{v}, 1, 1)$ must pass through $\mathbf{m}$. Further note that by induction, the graph G_n is also a path. The vertices $\mathbf{s_n} = (\mathbf{s_{n-2}}, 0, 0)$ and $\mathbf{t_n} = (\mathbf{s_{n-2}}, 1, 1)$ are diametrically opposite ends of this path. The path length is at least $2 \cdot 2^{\frac{n-2}{2}} + 2 > 2^{\frac{n}{2}}$. Also $\mathbf{s_2} = (0, 0)$, $\mathbf{s_n} = (\mathbf{s_{n-2}}, 0, 0)$, $\mathbf{t_n} = (\mathbf{s_{n-2}}, 1, 1)$ and hence $\mathbf{s_n} = (0, \ldots, 0), \mathbf{t_n} = (0, \ldots, 0, 1, 1)$.

We construct a sequence of 3-CNF formulas $\varphi_n(x_1, \ldots, x_n)$ so that $G_n = G(\varphi_n)$. Let $\varphi_2(x_1, x_2) = \bar{x}_1 \vee x_2$. Assume we have $\varphi_{n-2}(x_1, \ldots, x_{n-2})$. We add two variables x_{n-1} and x_n and the clauses

$$\varphi_{n-2}(x_1, \ldots, x_{n-2}), \;\; \bar{x}_{n-1} \wedge x_n$$

$$x_{n-1} \vee \bar{x}_n \vee \bar{x}_i \qquad \text{for } \; i \leq n-4 \tag{5.2}$$

$$x_{n-1} \vee \bar{x}_n \vee x_i \qquad \text{for } \; i = n-3, n-2 \tag{5.3}$$

Note that a clause in 5.2 is just the implication $(\bar{x}_{n-1} \wedge x_n) \rightarrow \bar{x}_i$. Thus clauses 5.2, 5.3 enforce the condition that $x_{n-1} = 0, x_n = 1$ implies that $(x_1, \ldots, x_{n-2}) = \mathbf{t_{n-2}} = (0, \ldots, 0, 1, 1)$. □

The proof that $\text{Conn}(\mathcal{S}_3)$ and $\text{st-Conn}(\mathcal{S}_3)$ are PSPACE-complete is fairly intricate, and is via a direct reduction from the computation of a polynomial-space Turing machine. The result for ST-CONN can also be proved using results of Hearne and Demaine on Non-deterministic Constraint Logic [HD02]. It does not appear that completeness for CONN follows from their results.

Lemma 35. $\text{st-Conn}(\mathcal{S}_3)$ *and* $\text{Conn}(\mathcal{S}_3)$ *are* PSPACE-*complete.*

Proof. Given a CNF($\mathcal{S}_3$) formula φ and solutions s, t we can check if they are connected in $G(\varphi)$ with polynomial amount of space. Similarly for CONN($\mathcal{S}_3$), we can check for all pairs of assignments whether they are satisfying and connected in $G(\varphi)$ with polynomial amount of space, so both problems are in PSPACE.

Next we show that $\text{Conn}(\mathcal{S}_3)$ and $\text{st-Conn}(\mathcal{S}_3)$ are PSPACE-hard. Let A be a language decided by a deterministic Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ in space n^k for some constant k. We give a polynomial time reduction from A to $\text{st-Conn}(\mathcal{S}_3)$ and $\text{Conn}(\mathcal{S}_3)$.

The reduction maps a string w (with $|w| = n$) to a 3-CNF formula φ and two satisfying assignments for the formula, which are connected in $G(\varphi)$ if and only if M accepts w. Furthermore, all satisfying assignments of φ are connected to one of these two assignments, so that $G(\varphi)$ is connected if and only if M accepts w.

Before we show how to construct φ, we modify M in several ways:

1. We add a clock that counts from 0 to $n^k \times |Q| \times |\Gamma|^{n^k} = 2^{O(n^{k+1})}$, which is the total number of possible distinct configurations of M. It uses a separate tape of length $O(n^{k+1})$ with the alphabet $\{0, 1\}$. Before a transition happens, control is passed on to the clock, its counter is incremented, and finally the transition is completed.

2. We define a token accepting configuration. Whenever q_{accept} is reached, the clock is stopped and set to zero, the original tape is erased and the head is placed in the initial position.

3. Whenever q_{reject} is reached the machine goes into its initial configuration. First w is written back on the input tape. This step requires adding n states to the machine in order to write the n letters of w. This increases the number of states of M' to $O(n)$. Next, the rest of the tape is erased, the clock is set to zero, the head is placed in the initial position, and the state is set to q_0.

4. Whenever the clock overflows, the machine goes into q_{reject}.

The new machine M' runs forever if w is not in A and accepts if w is in A. It also has the property that every configuration leads either to the accepting configuration or to the initial configuration with input w. Therefore the space of configurations is connected if and only if $w \in A$. Let's denote by Q' the states of M' and by δ' its transitions. As mentioned earlier, $|Q'| = O(n)$, and M' runs on two tapes, one of size $N = n^k$, and the other (for the clock) of size $N_c = O(n^{k+1})$. The alphabet of M' on one tape is Γ, and on the other $\{0, 1\}$. For simplicity we can also assume that at each transition the machine uses only one of the two tapes.

Next, we construct a CNF-formula ψ whose solutions are the configurations of M'. However, the space of solutions of ψ is disconnected.

For each $i \in [N]$ and $a \in \Gamma$, we have a variable $x(i, a)$. If $x(i, a) = 1$, this means that the i^{th} tape cell contains symbol a. For every $i \in [N]$ there is a variable $y(i)$ which is 1 if the head is at position i. For every $q \in Q'$, there is a variable $z(q)$ which is 1 if the current state is q. Similarly for every $j \in [N_c]$ and $a \in \{0, 1\}$ we have variables $x_c(j, a)$ and a variable $y_c(j)$ which is 1 if the head of the clock tape is at position j.

We enforce the following conditions:

1. Every cell contains some symbol:

$$\psi_1 = \bigwedge_{i \in [N]} \left(\vee_{a \in \Gamma} \; x(i, a) \right) \bigwedge_{j \in [N_c]} \left(\vee_{a \in \{0,1\}} \; x_c(j, a) \right).$$

2. No cell contains two symbols:

$$\psi_2 = \bigwedge_{i \in [N]} \bigwedge_{a \neq a' \in \Gamma} \left(\overline{x(i, a)} \vee \overline{x(i, a')} \right) \bigwedge_{j \in [N_c]} \left(\overline{x_c(j, 0)} \vee \overline{x_c(j, 1)} \right).$$

3. The head is in some position, and in some state:

$$\psi_3 = \left(\vee_{i \in [N]} \; y(i) \right) \bigwedge \left(\vee_{j \in [N]} \; y_c(j) \right) \bigwedge \left(\vee_{q \in Q_1} \; z(q) \right).$$

4. The head is in a unique position, and in a unique state:

$$\psi_4 = \bigwedge_{i \neq i' \in [N]} \left(\overline{y(i)} \vee \overline{y(i')} \right) \bigwedge_{j \neq j' \in [N_c]} \left(\overline{y_c(j)} \vee \overline{y_c(j')} \right) \bigwedge_{q \neq q' \in Q'} \left(\overline{z(q)} \vee \overline{z(q')} \right).$$

Solutions of $\psi = \psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \psi_4$ are in 1-1 correspondence with configurations of M'. Furthermore, the assignments corresponding to any two distinct configurations differ in at least two variables.

Next, to connect the solution space along valid transitions of M', we relax conditions 2 and 4 by introducing new transition variables, which allow the head to have two states or a cell to have two symbols at the same time. This allows us to go from one configuration to the next.

Consider a transition $\delta(q, a) = (q', b, R)$, which operates on the first tape, for example. Fix the position of the head of the first tape to be i, and the symbol in position $i + 1$ to be c. The variables that are changed by the transition are: $x(i, a)$, $y(i)$, $z(q)$, $x(i, b)$, $y(i + 1)$, $z(q')$. Before the transition the first three are set to 1, the second three are set to 0, and after the transition they are all flipped. Corresponding to this transition (which is specified by i, q, a, and c) we introduce a transition variable $t(i, q, a, c)$. We now relax conditions 2 and 4 as follows:

- Replace $\left(\overline{x(i,a)} \vee \overline{x(i,b)}\right)$ by $\left(\overline{x(i,a)} \vee \overline{x(i,b)} \vee t(i,q,a,c)\right)$.
- Replace $\left(\overline{y(i)} \vee \overline{y(i+1)}\right)$ by $\left(\overline{y(i)} \vee \overline{y(i+1)} \vee t(i,q,a,c)\right)$.
- Replace $\left(\overline{z(q)} \vee \overline{z(q')}\right)$ by $\left(\overline{z(q)} \vee \overline{z(q')} \vee t(i,q,a,c)\right)$.

This is done for every value of q, a, i and c (and also for transitions acting on the clock tape). We add the transition variables to the corresponding clauses so that for example the clause $\left(\overline{x(i,a)} \vee \overline{x(i,b)}\right)$ could potentially become very long, such as:

$$\left(\overline{x(i,a)} \vee \overline{x(i,b)} \vee t(i,q_1,a,c_1) \vee t(i,q_2,a,c_2) \vee \ldots\right).$$

However, the total number of transition variables is only polynomial in n. We also add a constraint for every pair of transition variables $t(i, q, a, c)$, $t(i', q', a', c')$, saying they cannot be 1 simultaneously: $(\overline{t(i,q,a,c)} \vee \overline{t(i',q',a',c')})$. This ensures that only one transition can be happening at any time. The effect of adding the transition variables to the clauses of ψ_2 and ψ_4 is that by setting $t(i, q, a, c)$ to 1, we can simultaneously set $x(i, a)$ and $x(i, b)$ to 1, and so on. This gives a path from the initial configuration to the final configuration as follows: Set $t(i, q, a, c) = 1$, set $x(i, b) = 1$, $y(i + 1) = 1$, $z(q') = 1$, $x(i, a) = 0$, $y(i) = 0$, $z(q) = 0$, then set $t(i, q, a, c) = 0$. Thus consecutive configurations are now connected. To avoid connecting to other configurations, we also add an expression to ensure that these are the only assignments the 6 variables can take when $t(i,q,a,c) = 1$:

$$\psi_{i,q,a,c} = \overline{t(i,q,a,c)} \vee ((x(i,a), y(i), z(q), x(i,b)), y(i+1), z(q')) \in \{111000, 111100, 111110, 111111, 011111, 001111, 000111\}).$$

This expression can of course be written in conjunctive normal form.

Call the resulting CNF formula $\varphi(\mathbf{x}, \mathbf{x_c}, \mathbf{y}, \mathbf{y_c}, \mathbf{z}, \mathbf{t})$. Note that $\varphi(\mathbf{x}, \mathbf{x_c}, \mathbf{y}, \mathbf{y_c}, \mathbf{z}, \mathbf{0}) = \psi(\mathbf{x}, \mathbf{x_c}, \mathbf{y}, \mathbf{y_c}, \mathbf{z})$, so a solution where all transition variables are 0 corresponds to a configuration of M'. To see that we have not introduced any shortcut between configurations that are not valid machine transitions, notice that in any solution of φ, at most a single transition variable can be 1. Therefore none of the transitional solutions belonging to different transitions can be adjacent. Furthermore, out of the solutions that have a transition variable set to 1, only the first and the last correspond to a valid configuration. Therefore none of the intermediate solutions can be adjacent to a solution with all transition variables set to 0.

The formula φ is a CNF formula where clause size is unbounded. We use the same reduction as in the proof of Lemma 33 to get a 3-CNF formula. By Lemma 30 and Corollary 31, ST-CONN and CONN for $\mathcal{S}_3$ are PSPACE-complete. □

5.4 Future directions

In Section 2, we conjectured a trichotomy for CONN$(\mathcal{S})$. We have made progress towards this conjecture; what remains is to pinpoint the complexity of CONN$(\mathcal{S})$ when $\mathcal{S}$ is Horn or dual-Horn. We can extend our dichotomy theorem for st-connectivity to formulas without constants; the complexity of connectivity for formulas without constants is open. We conjecture that when $\mathcal{S}$ is not tight, one can improve the diameter bound from $2^{\Omega(\sqrt{n})}$ to $2^{\Omega(n)}$. Finally, we believe that our techniques can shed light on other connectivity-related problems, such as approximating the diameter and counting the number of components. For counting the number of components, using results of Creignou and Hermann [CH96], we can show that the problem is in P for affine, monotone and dual-monotone relations, and #P-complete otherwise.

Chapter 6

Application of belief propagation for extended MRFs to source coding

In Chapter 3 we described the survey propagation algorithm for 3-SAT as a novel application of belief propagation to an extended MRF. Here we look at another problem of practical interest - the source coding problem. Motivated by the success of belief propagation algorithms applied to graphical error-correcting codes, we look for a way to apply belief propagation to the dual problem of source coding. While the naive application of belief propagation does not yield a good algorithm, we show that applying belief propagation to extended MRFs, such as the ones described in Chapter 3 results in very good performance.

6.1 Motivation

Graphical codes such as turbo and low-density parity check (LDPC) codes, when decoded with the belief propagation or sum-product algorithm, perform close to capacity [RSU01, e.g.,]. Similarly, LDPC codes have been successfully used for various types of lossless compression schemes [CSV03, e.g.,]. One standard approach to lossy source coding is based on trellis codes and the Viterbi algorithm. Here, we explore the use of codes based on graphs with cycles, whose potential has not yet been fully realized for lossy compression. A major challenge in applying such graphical codes to lossy compression is the lack of practical (i.e., computationally efficient) algorithms for encoding and decoding. For concreteness, we focus on the problem of quantizing a Bernoulli source with $p = \frac{1}{2}$. Developing and analyzing effective algorithms for this problem is a natural first step towards solving more general lossy compression problems (e.g., involving continuous sources or memory).

Our approach to lossy source coding is based on the dual codes of LDPC codes, known as low-density generator matrix (LDGM) codes. Other research groups have applied forms of survey propagation for source encoding based on codes composed of local non-linear "check" functions [M05́] and k-SATproblems with doping [BBCZ04]. Most recently, Martinian and Wainwright [MW06] have proposed more sophisticated low-density source codes that generalize the LDGM codes considered here, and proved that an optimal encoding algorithm can achieve the rate-distortion bound for a binary symmetric source.

6.2 Background and set-up

Given a $\mathrm{Ber}(\frac{1}{2})$ source, any particular i.i.d. realization $y \in \{0,1\}^n$ is referred to as a *source sequence*. The goal is to compress source sequences y by mapping them to shorter binary vectors $x \in \{0,1\}^m$ with $m < n$, where the quantity $R := \frac{m}{n}$ is the compression ratio. The source decoder then maps the compressed sequence x to a reconstructed source sequence $\widehat{y}$. For a given pair $(y, \widehat{y})$, the reconstruction fidelity is measured by the Hamming distortion $d_H(y, \widehat{y}) := \frac{1}{n}\sum_{i=1}^{n} |y_i - \widehat{y}_i|$. The overall quality of our encoder-decoder pair is measured by the average Hamming distortion $D := \mathbb{E}[d_H(Y, \widehat{Y})]$. For the $\mathrm{Ber}(\frac{1}{2})$ source, the rate distortion function is well-known to take the form $R(D) = 1 - H(D)$ for $D \in [0, 0.5]$, and 0 otherwise.

Our approach to lossy source coding is based on low-density generator matrix codes, hereafter referred to as LDGM codes, which arise naturally as the duals of LDPC codes. For a given rate $R = \frac{m}{n} < 1$, let A be an $n \times m$ matrix with $\{0,1\}$ entries, where we assume $\operatorname{rank} A = m$ without loss of generality. The low-density condition requires that the number of 1s in each row and column is bounded. The matrix A is the generator matrix of the LDGM, thereby defining the code $\mathbb{C}(A) := \{z \in \{0,1\}^n \mid z = Ax \text{ for some } x \in \{0,1\}^m\}$, where arithmetic is performed over $\mathrm{GF}(2)$. It will also be useful to consider the code over (x, z) given by $\bar{\mathbb{C}}(A) := \{(x, z) \in \{0,1\}^{n+m} \mid z = Ax\}$. We refer to elements of x as *information bits*, and elements of z as *source bits*. In the LDGM approach to source coding, the encoding phase of the source coding problem amounts to mapping a given source sequence $y \in \{0,1\}^n$ to an information vector $x(y) \in \{0,1\}^m$. Decoding is straightforward: we simply form $\widehat{y}(x) = Ax$. The challenge lies in the encoding phase: in particular, we must determine the information bit vector x such that the Hamming distortion $\frac{1}{n}\|y - Ax\|_1$ is minimized. This combinatorial optimization problem is equivalent to an MAX-XORSAT problem, and hence known to be NP-hard in general.

It is convenient to represent a given LDGM code, specified by generator matrix A, as a factor graph $G = (V, C, E)$, where $V = \{1, \ldots, m\}$ denotes the set of information bits and $C := \{1, \ldots, n\}$ denotes the set of checks (or equivalently, source bits), and E denotes the set of edges between checks and information bits. As illustrated in Figure 6.1, the n source bits are lined up at the bottom of the graph, and each is connected to a unique check neighbor. Each check, in turn, is connected to (some subset of) the m information bits at the top of the graph. Note that there is a one-to-one correspondence between source bits and checks. As in the previous chapters, we use letters a, b, c to refer to elements of C, corresponding either to a source bit or the associated check. Conversely, we use letters i, j, k to refer to information bits in the set V. For each information bit $i \in V$, let $C(i) \subseteq C$ denote its check neighbors: $C(i) := \{a \in C \mid (a, i) \in E\}$. Similarly, for each check $a \in C$, we define the set $V(a) := \{i \in V \mid (a, i) \in E\}$. We use the notation $\bar{V}(a) := V(a) \cup \{a\}$ to denote the set of *all* bits—both information and source—that are adjacent to check a.

6.3 Markov random fields and decimation with generalized codewords

A natural first idea to solving the source encoding problem would be to follow the channel coding approach: run the sum-product algorithm on the ordinary factor graph, and then threshold the resulting log-likelihood ratios (LLRs) at each bit to determine a source encoding $x(\widehat{y})$. Unfortunately, this approach fails: either the algorithm fails to converge or the LLRs fail to yield reliable information, resulting in a poor source encoding. Inspired by survey propagation for satisfiability problems [MPZ02], we consider an approach with two components: (a) extending the factor distribution so as to include not just ordinary codewords but also a set of partially assigned codewords, and (b) performing a sequence of message-passing and decimation steps, each of which entails setting fraction of bits to their preferred values.

More specifically, we consider Markov random fields over a larger space of so-called generalized codewords, which are members of the space $\{0, 1, *\}^{n+m}$ where $*$ is a new symbol. As we will see, the interpretation of $x_i = *$ is that the associated bit i is *free*. Conversely, any bit for which $x_i \in \{0, 1\}$ is *forced*. One possible view of a generalized codeword, as with the survey propagation and k-SATproblems, is as an index for a cluster of ordinary codewords. We define a family of Markov random fields, parameterized by a weight for $*$-variables, and a weight that measures fidelity to the source sequence. As a particular case, our family of MRFs includes a weighted distribution over the set of ordinary codewords. Although the specific extension considered here is natural to us (and yields good source coding results), it could be worthwhile to consider alternative ways in which to extend the original distribution to generalized codewords.

6.3.1 Generalized codewords

Definition 17 (Check states)**.** *In any generalized codeword, each check is in one of two possible exclusive states:*

(i) we say that check $a \in C$ is forcing whenever none of its bit neighbors are free, and the local $\{0,1\}$-codeword $(z_a; x_{V(a)}) \in \{0,1\}^{1+|V(a)|}$ satisfies parity check a.

(ii) on the other hand, check a is free whenever $z_a = *$, and moreover $x_i = *$ for at least one $i \in V(a)$.

Note that the source bit z_a is free (or forced) if and only if the associated check a is free (or forcing). With this set-up, our space of generalized codewords is defined as follows:

Definition 18 (Generalized codeword)**.** *A vector $(z,x) \in \{0,1,*\}^{n+m}$ is a valid generalized codeword when the following conditions hold:*

(i) all checks a are either forcing or free.

(ii) if some information bit x_i is forced (i.e., $x_i \in \{0,1\}$), then at at least *two check neighbors $a \in C(i)$ must be forcing it.*

For a generator matrix in which every information bit has degree two or greater, it can be seen that any ordinary codeword $(z,x) \in \bar{\mathbb{C}}(A)$ is also a generalized codeword. In addition, there are generalized codewords that include $*$'s in some positions, and hence do not correspond to ordinary codewords. One such (non-trivial) generalized codeword is illustrated in Figure 6.1. A natural way in which to generate generalized codewords is via an iterative "peeling" or "leaf-stripping" procedure. This is similar to the coarsening procedure from Chapter 3.

Peeling procedure: Given some initial source sequence $z \in \{0,1,*\}^n$, initialize all information bits x_i to be forced.

1. While there exists a forced information bit x_i with exactly one forcing check neighbor a, set $x_i = z_a = *$.

2. When all remaining forced information bits have at least two forcing checks, go to Step 3.

3. For any free check $z_a = *$ with *no* free information bit neighbors, set $z_a = \oplus_{i \in V(a)} x_i$.

When initialized with at least one free check, Step 1 of this peeling procedure can terminate in one of two possible ways: either the initial configuration is stripped down to the all-$*$ configuration, or Step 1 terminates at a configuration such that every forced information bit has two or more forcing check neighbors, thus ensuring that condition (ii) of Definition 18 is satisfied. As noted previously [MY03], these cores can be viewed as "duals" to stopping sets in the dual LDPC. Finally, Step 3 ensures that every free check has at least one free information bit, thereby satisfying condition (i) of Definition 18.

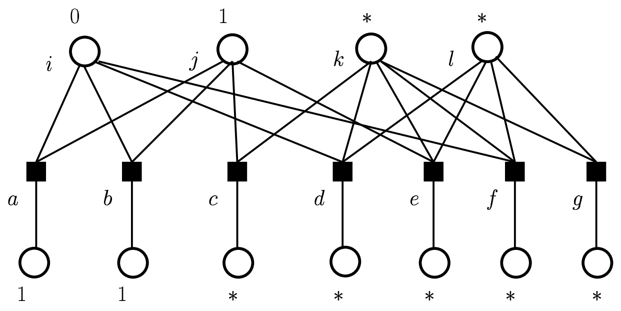


Figure 6.1. Illustration of a generalized codeword for a small LDGM. Information bits i and j are both forced; for each, the two forcing checks are a and b. The remaining checks and bits are all free.

6.3.2 Weighted version

Given a particular source sequence $y \in \{0,1\}^n$, we form a probability distribution over the set of generalized codewords as follows. For any generalized codeword $(z,x) \in \{0,1,*\}^{n+m}$, we define the sets

$$n_*^{\text{sou}}(z) := \big|\{i \in \{1,\ldots,n\} \,|\, z_i = *\}\big|,$$

$$n_*^{\text{info}}(x) := \big|\{i \in \{1,\ldots,m\} \,|\, x_i = *\}\big|,$$

corresponding to the number of $*$-variables in the source and information bits respectively. We associate non-negative weights w_{sou} and w_{info} with the $*$-variables in the source and information bits respectively. Finally, we introduce a non-negative parameter γ, which will be used to penalize disagreements between the source bits z and the given (fixed) source sequence y. Of interest to us in the sequel is the weighted probability distribution

$$p(z,x; w_{\text{sou}}, w_{\text{info}}, \lambda) \propto w_{\text{sou}}^{n_*^{\text{sou}}(z)} \times w_{\text{info}}^{n_*^{\text{info}}(x)} \times \exp^{-2\gamma d_H(y,z)}. \tag{6.1}$$

Note that for $w_{\text{sou}} = w_{\text{info}} = 0$, this distribution reduces to the standard weighted distribution over ordinary codewords.

6.3.3 Representation as Markov random field

We now seek to represent the set of generalized codewords as a Markov random field (MRF). A first important observation is that state augmentation is necessary to achieve such a Markov representation with respect to the original factor graph.

Lemma 36. *For positive $w_{\text{sou}}, w_{\text{info}}$, the set of generalized codewords* cannot *be represented as a Markov random field based on the original factor graph G where the state space at each bit is simply $\{0, 1, *\}$.*

Proof. It suffices to demonstrate that it is impossible to construct an indicator function for membership in the set of generalized codewords as a product of local compatibility functions on $\{0, 1, *\}$, one for each check. The key is that the set of all local generalized codewords cannot be defined only in terms of the variables $x_{\bar{V}(a)}$; rather, the validity depends also on all bit neighbors of checks that are incident to bits in $\bar{V}(a)$. or more formally on bits with indices in the set

$$\cup_{i \in \bar{V}(a)} \big\{ j \in V \;\big|\; j \in V(b) \text{ for some } b \in C(i) \big\}. \tag{6.2}$$

As a particular illustration, consider the trivial LDGM code consisting of a single source bit (and check) connected to three information bits. From Definition 17 and Definition 18, it can be seen that the only generalized codeword is the all-$*$ configuration. Thus, any check function used to define membership in the set of generalized codewords would have to assign zero mass to any other $\{0, 1, *\}$ configuration. Now suppose that this simple LDGM is embedded within a larger LDGM code. For instance, consider the check labeled e (with source bit z_e) and corresponding information bits $\{j, k, l\}$ in Figure 6.1. With respect to the generalized codeword in this figure, we see that the local configuration $(x_j, x_k, x_l z_e) = (1, *, *, *)$ is locally valid, which contradicts our conclusion from considering the trivial LDGM code in isolation. Hence, the constraints enforced by a given check change depending on the larger context in which it is embedded. $\square$

Consequently, obtaining a factorization of the distribution requires keeping track of variables in the extended set (6.2). Accordingly, as in the reformulation of survey propagation for SATproblems, we introduce a new variable P_i, so that there is a vector (x_i, P_i) associated with each

Bits to checks

$$\begin{aligned}
M^{0f}_{i\to a} &\leftarrow \lambda^0_i \Big\{ \prod_{b\in C(i)\backslash\{a\}} \big[M^{0f}_{b\to i} + M^{0w}_{b\to i}\big] - \prod_{b\in C(i)\backslash\{a\}} M^{0w}_{b\to i} \Big\} \\
M^{1f}_{i\to a} &\leftarrow \lambda^1_i \Big\{ \prod_{b\in C(i)\backslash\{a\}} \big[M^{1f}_{b\to i} + M^{1w}_{b\to i}\big] - \prod_{b\in C(i)\backslash\{a\}} M^{1w}_{b\to i} \Big\} \\
M^{0w}_{i\to a} &\leftarrow \lambda^0_i \Big\{ \prod_{b\in C(i)\backslash\{a\}} \big[M^{0f}_{b\to i} + M^{0w}_{b\to i}\big] - \prod_{b\in C(i)\backslash\{a\}} M^{0w}_{b\to i} - \sum_{c\in C(i)\backslash\{a\}} M^{0f}_{c\to i} \prod_{b\in C(i)\backslash\{a,c\}} M^{0w}_{b\to i} \Big\}. \\
M^{1w}_{i\to a} &\leftarrow \lambda^1_i \Big\{ \prod_{b\in C(i)\backslash\{a\}} \big[M^{1f}_{b\to i} + M^{1w}_{b\to i}\big] - \prod_{b\in C(i)\backslash\{a\}} M^{1w}_{b\to i} - \sum_{c\in C(i)\backslash\{a\}} M^{1f}_{c\to i} \prod_{b\in C(i)\backslash\{a,c\}} M^{1w}_{b\to i} \Big\}. \\
M^{*}_{i\to a} &\leftarrow w_{\text{info}} \prod_{b\in C(i)\backslash\{a\}} M^{*}_{b\to i}
\end{aligned}$$

Checks to bits

$$\begin{aligned}
M^{0f}_{a\to i} &\leftarrow \frac{1}{2}\Big[\prod_{j\in\bar{V}(a)\backslash\{i\}} \big(M^{0f}_{j\to a} + M^{1f}_{j\to a}\big) + \prod_{j\in\bar{V}(a)\backslash\{i\}} \big(M^{0f}_{j\to a} - M^{1f}_{j\to a}\big)\Big] \\
M^{1f}_{a\to i} &\leftarrow \frac{1}{2}\Big[\prod_{j\in\bar{V}(a)\backslash\{i\}} \big(M^{0f}_{j\to a} + M^{1f}_{j\to a}\big) - \prod_{j\in\bar{V}(a)\backslash\{i\}} \big(M^{0f}_{j\to a} - M^{1f}_{j\to a}\big)\Big] \\
M^{0w}_{a\to i} &\leftarrow \prod_{j\in\bar{V}(a)\backslash\{i\}} \big[M^{*}_{j\to a} + M^{1w}_{j\to a} + M^{0w}_{j\to a}\big] - \prod_{j\in\bar{V}(a)\backslash\{i\}} \big[M^{1w}_{j\to a} + M^{0w}_{j\to a}\big] \\
&\qquad - \sum_{k\in\bar{V}(a)\backslash\{i\}} M^{*}_{k\to a} \prod_{j\in\bar{V}(a)\backslash\{i,k\}} \big[M^{1w}_{j\to a} + M^{0w}_{j\to a}\big] \\
M^{1w}_{a\to i} &= M^{0w}_{a\to i}. \\
M^{*}_{a\to i} &\leftarrow \prod_{j\in\bar{V}(a)\backslash\{i\}} \big[M^{*}_{j\to a} + M^{1w}_{j\to a} + M^{0w}_{j\to a}\big] - \prod_{j\in\bar{V}(a)\backslash\{i\}} \big[M^{1w}_{j\to a} + M^{0w}_{j\to a}\big]
\end{aligned}$$

Figure 6.2. Message-passing updates involve five types of messages from bit to check, and five types of messages from check to bit. Any source bit z_a always sends to its only check a the message 5-vector $(\psi_a(0),\ \psi_a(1),\ 0,\ 0,\ w_{\text{sou}})$. The message vector in any given direction on any edge is normalized to sum to one.

bit. To define P_i, first let $\mathcal{P}(i) = \mathcal{P}(C(i))$ denote the power set of all of the clause neighbors $C(i)$ of bit i. (I.e., $\mathcal{P}(i)$ is a set with $2^{|C(i)|}$ elements). The variable P_i takes on subsets of $C(i)$, and we decompose it as $P_i = P^0_i \cup P^1_i$, where at any time *at most one* of P^1_i and P^0_i are non-empty. The variable P_i has the following decomposition and interpretation: (a) if $P^0_i = P^1_i = \emptyset$, then no checks are forcing bit x_i; (b) if $P_i = P^1_i \neq \emptyset$, then certain checks are forcing x_i to be one (so that necessarily $x_i = 1$); and (c) similarly, if $P_i = P^0_i \neq \emptyset$, then certain checks are forcing x_i to be zero (so that necessarily $x_i = 0$). By construction, this definition excludes the case that both P^0_i and P^1_i non-empty at the same time, so that the state space of P_i has cardinality $2^{|C(i)|} + 2^{|C(i)|} - 1 \ = \ 2^{|C(i)|+1} - 1$.

We now specify a set of compatibility functions to capture the Markov random field over generalized codewords.

Variable compatibilities

For each bit index i (or a), let λ_i^1 and λ_i^0 denote the weights assigned to the events $x_i = 1$ and $x_i = 0$ respectively. For source encoding, these weights are specified as $\lambda_i^1 = \lambda_i^0 = 1$ for all information bits i (i.e., no a priori bias on the information bits), so that the compatibility function takes the form:

$$\psi_i(x_i, P_i) := \begin{cases} 1 & \text{if } x_i = 1 \text{ and } |P_i| = |P_i^1| \geq 2 \\ 1 & \text{if } x_i = 0 \text{ and } |P_i| = |P_i^0| \geq 2 \\ w_{\text{info}} & \text{if } x_i = * \text{ and } P_i = \emptyset \end{cases} \tag{6.5}$$

The source bits have compatibility functions of the form $\psi_a(z_a, P_a) = \lambda_a^0$ if $z_a = 0$ and $P_a^1 = \{a\}$; $\psi_a(z_a, P_a) = \lambda_a^1$ if $z_a = 1$ and $P_a^1 = \{a\}$; and $\psi_a(z_a, P_a) = w_{\text{sou}}$ if $z_a = *$ and $P_a = \emptyset$. Here $\lambda_a^1 := y_a \exp(\gamma) + (1 - y_a) \exp(-\gamma)$, $\lambda_a^0 := 1/\lambda_a^1$, and the parameter $\gamma > 0$ reflects how strongly the source observations are weighted.

Check compatibilities

For a given check a, the associated compatibility function $\phi_a(x_{V(a)}, z_a, P_{V(a)})$ is constructed to ensure that the following two properties hold: (1) The configuration $\{z_a\} \cup x_{V(a)}$ is *valid* for check a, meaning that (a) either it includes no $*$'s, in which case the pure $\{0, 1\}$ configuration must be a local codeword; or (b) the associated source bit is free (i.e., $z_a = *$), and $x_i = *$ for at least one $i \in V(a)$. (2) For each index $i \in V(a)$, the following condition holds: (a) either $a \in P_i$ and a forces x_i, or (b) there holds $a \notin P_i$ and a does not force x_i.

Proposition 37. *With the singleton and factor compatibilities as above, consider the distribution* $p^{wei}((x, P(x)), (z, P(z)))$*, defined as a Markov random field (MRF) over the factor graph in the following way:*

$$\prod_{i \in V} \psi_i(x_i, P_i) \prod_{a \in C} \psi_a(z_a, P_a) \phi_a(x_{V(a)}, z_a, P_{V(a)}). \tag{6.6}$$

Its marginal distribution over (x, z) *agrees with the weighted distribution* (6.1).

6.3.4 Applying belief propagation

In our extended Markov random field, the random variable at each bit node i is of the form (x_i, P_i), and belongs to the Cartesian product $\{0, 1, *\} \times [\mathcal{P}(i) \times \{0, 1\}]$. (To clarify the additional $\{0, 1\}$, the variable $P_i = P_i^0 \cup P_i^1$ corresponds to a particular subset of $\mathcal{P}(i)$, but we also need to specify whether $P_i = P_i^0$ or $P_i = P_i^1$.) Although the cardinality of $\mathcal{P}(i)$ can is exponential in the bit degree, it turns out that message-passing can be implemented by keeping track of only five numbers for each message (in either direction). These five cases are the following:

(i) $(x_i = 0, a \in P_i^0)$**:** check a is forcing x_i to be equal to zero. We say x_i is a *forced zero with respect to* a, and use $M_{i\to a}^{0f}$ and $M_{a\to i}^{0f}$ for the corresponding bit-to-check and check-to-bit messages.

(ii) $(x_i = 1, a \in P_i^1)$**:** check a is forcing x_i to be equal to one. We say that x_i is a *forced one with respect to* a, and denote the corresponding messages $M_{i\to a}^{1f}$ and $M_{a\to i}^{1f}$.

(iii) $(x_i = 0, \emptyset \neq P_i^0 \subseteq C(i)\backslash\{a\})$**:** A check subset *not* including a is forcing $x_i = 0$. We say x_i is a *weak zero with respect to check* a, and denote the messages $M_{i\to a}^{0w}$ and $M_{a\to i}^{0w}$.

(iv) $(x_i = 1, \emptyset \neq P_i^1 \subseteq C(i)\backslash\{a\})$**:** A check subset *not* including a forces $x_i = 1$. We say that that x_i is a *weak one with respect to check* a, and use corresponding messages $M_{i\to a}^{1w}$ and $M_{a\to i}^{1w}$.

(v) $(x_i = *, P_i^1 = P_i^0 = \emptyset)$**:** No checks force bit x_i; associated messages are denoted by $M_{i\to a}^{*}$ and $M_{i\to a}^{*}$.

The differences between these cases is illustrated in Figure 6.1. The information bit $x_i = 0$ is a forced zero with respect to checks a and b (case (i)), and a weak zero with respect to checks d and f (case (iii)). Similarly, the setting $x_j = 1$ is a forced one for checks a and b, and a weak one for checks c and e. Finally, there are a number of $*$ variables to illustrate case (v). With these definitions, it is straightforward (but requiring some calculation) to derive the BP message-passing updates as applied to the generalized MRF, as shown in Figure 6.2. It can be seen that this family of algorithms includes ordinary BP as a special case: in particular, if $w_{\text{sou}} = w_{\text{info}} = 0$, then the updates reduce to the usual BP updates on a weighted MRF over ordinary codewords.

When the message updates have converged, the sum-product pseudomarginals (i.e., approximations to the true marginal distributions) are calculated as follows:

$$\begin{aligned}
\mu_i(0) &\propto \lambda_i^0 \Big\{ \prod_{a\in C(i)} \left[M_{a\to i}^{0f} + M_{a\to i}^{0w}\right] - \prod_{a\in C(i)} M_{a\to i}^{0w} \\
&\qquad - \sum_{b\in C(i)} M_{b\to i}^{0f} \prod_{a\in C(i)\backslash\{b\}} M_{a\to i}^{0w} \Big\} \\
\mu_i(*) &\propto w_{\text{info}} \prod_{a\in C(i)} M_{a\to i}^{*}.
\end{aligned}$$

with a similar expression for $\mu_i(1)$. The overall triplet is normalized to sum to one. As with survey propagation and SATproblems, the practical use of these message-passing updates for source encoding entail: (1) Running the message-passing algorithm until convergence; (2) Setting a fraction of information bits, and simplifying the resulting code; and (3) Running the message-passing algorithm on the simplified code, and repeating. We choose information bits to set based on bias magnitude $B_i := |\mu_i(1) - \mu_i(0)|$.

6.4 Experimental results

We have applied a C-based implementation of our algorithm to LDGM codes with various degree distributions and source sequences of length n ranging from 200 to $100,000$. Although message-passing can be slow to build up appreciable biases for regular degree distributions, we find that biases accumulate quite rapidly for suitably irregular degree distributions. We chose codes randomly from irregular distributions optimized[1] for ordinary message-passing on the BEC or BSC using density-evolution [RSU01]. Figure 6.3 compares experimental results to the rate-distortion bound $R(D)$. We applied message-passing using a damping parameter $\alpha = 0.50$, and with $w_{\text{sou}} = 1.10$, $w_{\text{info}} = 1.0$ and γ varying from 1.45 (for rate 0.90) to 0.70 (for rate 0.30). Each round of decimation entailed setting all information bits with biases above a given threshold, up to a maximum of 2% of the total number of bits. As seen in Figure 6.3, the performance is already very good even for intermediate block length $n = 10,000$, and it improves for larger block lengths. After having refined our decimation procedure, we have also managed to obtain good source encodings (though currently not quite as good as Figure 6.3) using ordinary BP message-passing (i.e., $w_{\text{sou}} = w_{\text{info}} = 0$) and decimation; however, in experiments to date, in which we do not adjust parameters adaptively during decimation, we have found it difficult to obtain consistent convergence of ordinary BP (and more generally, message-passing with $w_{\text{sou}}, w_{\text{info}} \approx 0$) over all decimation rounds.

6.5 Future directions

Whereas in the case of the 3-SAT problem we were able to draw a connection between generalized assignments and the clustering of solutions, in the source coding problem we did not draw such an explicit connection. In fact, the design of the generalized MRF for source coding is

[1]This choice, though not optimized for source encoding, is reasonable in light of the connection between LDPC channel coding and LDGM source coding in the erasure case [MY03].

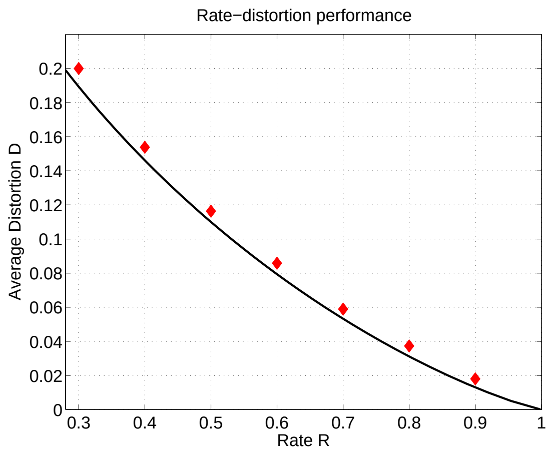


Figure 6.3. Plot of rate versus distortion, comparing the Shannon limit (solid line) and empirical performance using LDGM codes with blocklength $n = 10,000$. Each diamond is the average distortion over 15 trials.

somewhat arbitrary. Its success is a "proof of concept", confirming that it is possible to apply belief propagation to this problem, but further research is needed to understand what are the theoretical reasons for the success, and tune it further. It still remains to perform a systematic comparison of the performance of message-passing/decimation procedures over a range of parameters $(w_{\text{sou}}, w_{\text{info}}, \gamma)$ for a meaningful quantitative comparison.

Another important direction is developing methods for optimizing LDGM codes, and the choice of parameters in our extended MRFs with respect to the particular code. An important practical issue is to investigate the tradeoff between the conservativeness of the decimation procedure (i.e., computation time) versus quality of source encoding. Finally, the limiting rate-distortion performance of LDGM codes is a theoretical question that (to the best of our knowledge) remains open.

Bibliography

[ABS03] M. Alekhnovich and E. Ben-Sasson. Linear upper bounds for random walk on small density random 3cnf. In *Proc.* 44^{th} *IEEE Symp. Foundations of Computer Science*, pages 352–361, 2003.

[ACIM01] D. Achlioptas, A. Chtcherba, G. Istrate, and M. Molloy. The phase transition in NAE-SAT and 1-in-k SAT. In *Proc.* 12^{th} *ACM-SIAM Symp. Discrete Algorithms*, pages 721–722, 2001.

[AGK05] E. Aurell, U. Gordon, and S. Kirkpatrick. Comparing beliefs, surveys, and random walks. In *Neural Information Processing Systems*, January 2005.

[AP03] D. Achiloptas and Y. Peres. The threshold for random k-SAT is $2^k 2 \log 2 - o(k)$. In *Proc.* 35^{th} *ACM Symp. Theory of Computing*, pages 223–231, 2003.

[ART06] D. Achlioptas and F. Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In 38^{th} *ACM Symp. Theory of Computing*, 2006.

[BBCZ04] D. Battaglia, A. Braunstein, J. Chavas, and R. Zecchina. Source coding by efficient selection of ground states. Technical report, 2004. Preprint at URL:http://lanl.arXiv.org/abs/cond-mat/0412652.

[BFU93] A. Z. Broder, A. M. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-cnf formulas. In *Proc.* 4^{th} *ACM-SIAM Symp. Discrete Algorithms*, pages 322–330, 1993.

[BMZ03] A. Braunstein, M. Mézard, and R. Zecchina. Survey propagation: an algorithm for satisfiability. Technical report, 2003. Preprint at URL:http://lanl.arXiv.org/cs.CC/0212002.

[BZ04] A. Braunstein and R. Zecchina. Survey propagation as local equilibrium equations. Technical report, 2004. Preprint at URL:http://lanl.arXiv.org/cond-mat/0312483.

[CD04] Nadia Creignou and Hervé Daudé. Combinatorial sharpness criterion and phase transition classification for random csps. *Inf. Comput.*, 190(2):220–238, 2004.

[CF02] J. Coughlan and S. Ferreira. Finding deformable shapes using loopy belief propagation. In *European Conference on Computer Vision*, 2002.

[CH96] N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125(1):1–12, 1996.

[Chv91] V. Chvatal. Almost all graphs with 1.44 edges are 3-colourable. *Random Struct. Algorithms*, 2:11–28, 1991.

[CM04] S. Cocco and R. Monasson. Heuristic average-case analysis of the backtrack resolution of random 3-satisfiability instances. *Theor. Comput. Sci.*, 320(2-3):345–372, 2004.

[Coo71] S. Cook. The complexity of theorem-proving procedures. In *Proceeding of* 3^{rd} *ACM Symp. Theory of Computing*, page 151, 1971.

[CR92] V. Chvatal and B. Reed. Mick gets some (the odds are on his side). In *Proc.* 33^{rd} *IEEE Symp. Foundations of Computer Science*, pages 620–627, Pittsburgh Pennsylvania, 1992.

[CSV03] G. Caire, S. Shamai, and S. Verdu. A new data compression algorithm for sources with memory based on error-correcting codes. In *Proceedings of the 2003 IEEE Information Theory Workshop*, pages 291–295, 2003.

[DB97] O. Dubois and Y. Boufkhad. A general upper bound for the satisfiability threshold of random r-sat formulae. *Journal of Algorithms*, 24:395–420, 1997.

[DBM00] O. Dubois, Y. Boufkhad, and J. Mandler. Typical random 3-sat formulae and the satisfiability threshold. In *Proceedings of 11'th SODA*, pages 126–127, 2000.

[DBM03] O. Dubois, Y. Boufkhad, and J. Mandler. Typical random 3-sat formulae and the satisfiability threshold. *Electronic Colloquium on Computational Complexity (ECCC)*, 10(007), 2003.

[DLL62] M. Davis, G. Logemann, and D. W. Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.

[dlV92] W. Fernandez de la Vega. On random 2-sat. Unpublished manuscript., 1992.

[FPC00] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Intl. J. Computer Vision*, 40(1):25–47, 2000.

[Fri99] E. Friedgut. Neccesary and sufficient conditions for sharp threhsolds of graph properties and the k-problem. *J. Amer. Math. Soc.*, 12:1017–1054, 1999.

[GKMP06] P. Gopalan, Ph. G. Kolaitis, E. Maneva, and C. H. Papadimitriou. The connectivity of boolean satisfiability: Computational and structural dichotomies. In *Proceedings of* 33^{rd} *International Colloquium on Automata, Languages and Programming*, 2006.

[Goe96] A. Goerdt. A remark on random 2-sat. *J. Computer System and Sciences*, 53:469–486, 1996.

[HD02] R.A. Hearne and E.D. Demaine. The Nondeterministic Constraint Logic model of computation: Reductions and applications. In 29^{th} *Intl. Colloquium on Automata, Languages and Programming*, pages 401–413, 2002.

[JSV00] S. Janson, Y. C. Stamatiou, and M. Vamvakari. Bounding the unsatisfiability threshold of random 3-sat. *Random Struct. Algorithms*, 17(2):103–116, 2000.

[Jub99] L. Juban. Dichotomy theorem for the generalized unique satisfiability problem. In 12^{th} *Intl. Symp. Fundamentals of Computation Theory*, pages 327–337, 1999.

[KFL01] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Info. Theory*, 47:498–519, February 2001.

[Kir04] S. Kirkpatrick. On survey propagation. Personal communication, 2004.

[KKKS98] L. M. Kirousis, E. Kranakis, D. Krizanc, and Y. C. Stamatiou. Approximating the unsatisfiability threshold of random formulas. *Random Struct. Algorithms*, 12(3):253–269, 1998.

[KKL00] A. Kaporis, L. M. Kirousis, and E. G. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. In *Proceedings of 10'th Annual European Symposium on Algorithm*, pages 574–585, 2000.

[KKS$^+$01] A. C. Kaporis, L. M. Kirousis, Y. C. Stamatiou, M. Vamvakari, and M. Zito. Coupon collectors, q-binomial coefficients and the unsatisfiability threshold. In *Proceedings of* 7^{th} *Italian Conf. on Theoretical Computer Science*, pages 328–338, 2001.

[KMPS94] A. Kamath, R. Motwani, K. V. Palem, and P. G. Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. In *Proc.* 35^{th} *IEEE Symp. Foundations of Computer Science*, pages 592–603, 1994.

[LMS98] M. Luby, M. Mitzenmacher, and M. A. Shokrollahi. Analysis of random processes via and-or tree evaluation. In *Proc.* 9^{th} *ACM-SIAM Symp. Discrete Algorithms*, pages 364–373, 1998.

[Ḿ05] M. Mézard. Personal communication, Santa Fe Coding Workshop, 2005.

[MMW05] E. Maneva, E. Mossel, and M. J. Wainwright. A new look at survey propagation and its generalizations. In *Proc.* 16^{th} *ACM-SIAM Symp. Discrete Algorithms*, pages 1089–1098, 2005.

[MMZ05] T. Mora, M. Mézard, and R. Zecchina. Clustering of solutions in the random satisfiability problem. *Phys. Rev. Lett.*, 2005. In press.

[Mol03] M. Molloy. Models for random constraint satisfaction problems. *SIAM Jour. Comput.*, 32(4):935–949, 2003.

[Mol04] M. Molloy. The pure literal rule threshold and cores in random hypergraphs. In *Proc.* 15^{th} *ACM-SIAM Symp. Discrete Algorithms*, pages 672–681, 2004.

[MP03] M. Mézard and G. Parisi. The cavity method at zero temperatture. *J. Stat. Phys.*, 111:1, 2003.

[MPV87] M. Mézard, G. Parisi, and M. A. Virasoro. *Spin Glass Theory and Beyond*. World Scientific, Singapore, 1987.

[MPZ02] M. Mézard, G. Parisi, and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297, 812, 2002. (Scienceexpress published on-line 27-June-2002; 10.1126/science.1073287).

[MRTZ03] M. Mézard, F. Ricci-Tersenghi, and R. Zecchina. Two solutions to diluted p-spin models and xorsat problems. *J. Stat. Phys.*, 111:505, 2003.

[MW06] E. Martinian and M. Wainwright. Low density codes achieve therate-distortion bound. *Data Compression Conference*, pages 153–162, 2006.

[MY03] E. Martinian and J. Yedidia. Iterative quantization using codes on graphs. In *Proceedings of Allerton Conference on Control, Computing, and Communication*, 2003.

[MZ02] M. Mézard and R. Zecchina. Random k-satisfiability: from an analytic solution to an efficient algorithm. *Phys. Rev. E*, 66, 2002.

[MZK$^+$99] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Determining computational complexity from characteristic 'phase transitions'. *Nature*, 400:133–137, 1999.

[Pap91] C. H. Papadimitriou. On selecting a satisfying truth assignment. In *Proceedings of* 32^{nd} *IEEE Symp. Foundations of Computer Science*, pages 163–169, 1991.

[Par02] G. Parisi. On local equilibrium equations for clustering states. Technical report, 2002. Preprint at URL:http://lanl.arXiv.org/cs.CC/0212047.

[Pea88] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, Palo Alto, CA, 1988.

[RPF99] J.W. Rosenthal, J.M. Plotkin, and J. Franco. The probability of pure literals. *Journal of Computational Logic*, 9:501–513, 1999.

[RSU01] T. Richardson, A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity check codes. *IEEE Trans. Info. Theory*, 47:619–637, 2001.

[RU01] T. Richardson and R. Urbanke. The capacity of low-density parity check codes under message-passing decoding. *IEEE Trans. Info. Theory*, 47:599–618, 2001.

[Sch78] T.J. Schaefer. The complexity of satisfiability problems. In *Proc.* 10^{th} *ACM Symp. Theory of Computing*, pages 216–226, 1978.

[SM04] G. Semerjian and R. Monasson. A study of pure random walk on random satisfiability problems with "physical" methods. In E. Giunchiglia and A. Tachella, editors, *Lecture notes in computer science*, volume 2919 of *Proceedings of the SAT 2003 conference*, page 120. Springer Verlag, 2004.

[Sta97] R. P. Stanley. *Enumerative combinatorics*, volume 1. Cambridge University Press, Cambridge, UK, 1997.

[WM05] M. J. Wainwright and E. Maneva. Lossy source encoding via message-passing and decimation over generalized codewords of LDGM codes. In *Proceedings of IEEE International Symposium on Information Theory*, pages 1493–1497, 2005.

[YFW03] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *Exploring Artificial Intelligence in the New Millennium*, chapter 8. Science and Technology books, 2003.

[YFW05] J.S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Trans. Info. Theory*, 51(7):2282–2312, July 2005.