UNIVERSITY OF CALIFORNIA
SANTA CRUZ

UNSATISFIABILITY BOUNDS FOR RANDOM CONSTRAINT SATISFACTION PROBLEMS FROM AN ENERGETIC INTERPOLATION METHOD

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

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Ricardo Menchaca-Mendez

March 2013

The Dissertation of Ricardo Menchaca-Mendez
is approved:

Professor Dimitris Achlioptas, Chair

Professor Manfred K. Warmuth

Professor David P. Helmbold

Tyrus Miller
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

Abstract

Unsatisfiability Bounds for Random Constraint Satisfaction Problems from an Energetic Interpolation Method

by

Ricardo Menchaca-Mendez

The interpolation method, originally developed in statistical physics, transforms distributions of random Constraint Satisfaction Problems (CSPs) to distributions of much simpler problems while bounding the change in a number of associated statistical quantities along the transformation path. By now, it is known that, in principle, the method can yield rigorous unsatisfiability results if one "plugs in an appropriate functional distribution" to the derived expressions. A drawback of the method is that identifying the appropriate distribution leads to major analytical challenges as the relevant distributions are, in fact, infinite dimensional objects. We develop a variant of the interpolation method for random CSPs on arbitrary sparse degree distributions which trades accuracy for tractability. In particular, our bounds only require the solution of a 1-dimensional optimization problem (which typically turns out to be very easy) and as such can be used to compute explicit rigorous unsatisfiability bounds. We use this new method to analyze the performance of a number of algorithms on random 3-CNF formulas with n variables and $m = rn$ clauses. A long series of papers analyzing so-called "myopic" algorithms has provided a sequence of lower bounds for the satisfiability threshold, which is widely believed to be $r \approx 4.26$. Indeed, for each myopic algorithm $\mathcal{A}$ it is known

that there exists an algorithm-specific clause-density, $r_{\mathcal{A}}$, such that if $r < r_{\mathcal{A}}$, the algorithm finds a satisfying assignment in linear time. For example, $r_{\mathcal{A}}$ is $8/3 = 2.66...$ for ORDERRED-DLL and $3.003...$ for GENERALIZED UNIT CLAUSE. We prove that for densities well within the provable satisfiable regime, every backtracking extension of either of these algorithms takes exponential time. Specifically, all extensions of ORDERRED-DLL take exponential time for $r > 2.78$ and the same is true for GENERALIZED UNIT CLAUSE for all $r > 3.1$. Our results imply exponential lower bounds for many other myopic algorithms for densities similarly close to the corresponding $r_{\mathcal{A}}$.

To my parents, Reina and Rolando,

my siblings, Adriana and Rolando,

and my best friend, Veronica.

Acknowledgments

I want to thank UC MEXUS - CONACyT for their support during the first 5 years of my doctorate.

Chapter 1

Introduction

In this thesis we develop methods to analyze random Constraint Satisfaction Problems (CSPs). A CSP is defined by a set of variables and a set of constraints, each binding a subset of the variables by forbidding some of their possible joint values. The goal is to find an assignment to the variables so that all constraints are satisfied. Typically, CSPs involve variables taking values in a small domain, e.g., $\{0, 1\}$, with each constraint binding only a few of the variables. CSPs play an important role in Computer Science. Their applications range from program verification, where the unsatisfiability of a CSP built from the algorithm and its specification implies the correctness of the program; to error correcting codes, where a CSP is used both to detect erroneous messages (assignments), when they do not satisfy the CSP, and to correct them by finding the "closest" satisfying message to the erroneous one.

Perhaps the most studied examples of CSPs in computer science are the graph coloring and the satisfiability problems.

Definition (The q-coloring problem.)**.** A q-coloring of the vertices of a graph $G = (V, E)$ is a function c that maps each vertex in V to the set $\{1, \ldots q\}$. The q-coloring problem consist on finding a q-coloring such that for every edge $(u, v) \in E$ $c(u) \neq c(v)$ or to report that none exist.

Definition (The satisfiability problem)**.** A clause is a disjunction (logical OR) of literals (boolean variables or its negations) that forbids one of all the possible assignments of the variables. A conjunctive normal form (CNF) formula is a boolean formula written as a conjunction (logical AND) of clauses, for instance, $(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$ is a CNF formula on 4 variables where the symbols $\wedge$ and $\vee$ denote logical AND and logical OR respectively, and $\bar{x}$ denotes the negation of variable x. The satisfiability problem is to find an assignment to the variables appearing in a CNF formula that satisfies all clauses or to report that none exist. When all clauses operate on exactly k literals, the problem is known as k-SAT.

The satisfiability problem is central to the understanding of computational complexity. It was the first practically relevant problem that was shown to belong to the NP-complete class in the seminal papers of Cook [17] and Levin [37]. Moreover, the satisfiability problem is of practical interest as it arises naturally in many different settings.

1.1 Random CSPs

A natural representation of CSPs is as a bipartite constraint graph, known as factor graph, where vertices represent both the variables and constraints, and an edge $\{v, c\}$ reflects that variable v is bound by constraint c. There are many advantages with using this representation as many properties about a CSP instance translate into properties of the factor graph representation, for example when the factor graph of a CNF formula is a tree, then the problem can be solved in polynomial time.

In this thesis, we focus on CSPs where the constraint graph is random i.e., one has a set of n variables all with the same finite (and small) domain D and a set of m constraints, each of which binds a small ($O(1)$) randomly selected subset of variables. We will study random CSPs from an asymptotic point of view, i.e., as the number of variables grows to infinity. We will say that a sequence of random events $\mathcal{E}_n$ occurs with high probability (*w.h.p.*) if $\lim_{n\to\infty} \mathbf{P}[\mathcal{E}_n] = 1$ and with uniform positive probability (*w.u.p.p.*) if $\lim_{n\to\infty} \mathbf{P}[\mathcal{E}_n] > 0$. Random CSPs have been used as "simple" models to study the roots of the hardness of typical instances of NP-complete problems. For instance, random formulas have emerged as a mathematically tractable vehicle for studying proof systems and the performance of satisfiability algorithms. In particular, random k-SAT formulas have become a popular benchmark for testing satisfiability algorithms[1].

Definition (Random k-CNF formulas)**.** For a given set of n boolean variables, let $B_{k,n}$

[1] http://www.cs.ubc.ca/~hoos/SATLIB/benchm.html

denote the set of all possible $2^k\binom{n}{k}$ clauses of k distinct non-complementary literals. A random k-SAT formula $\mathcal{F}_k(n,m)$ is formed by selecting uniformly and independently m clauses from $B_{k,n}$ and taking their conjunction.

Let $g_k(n,m)$ denote the probability that $\mathcal{F}_k(n,m)$ is satisfiable. Observe that $g_k(n,m)$ is a monotone decreasing function of m since adding additional clauses can only reduce the number of satisfying assignments. Using the first moment method one can show that $\mathcal{F}_k(n,rn)$ is unsatisfiable *w.h.p.* when $r \geq 2^k \ln 2$. The first moment method bounds probabilities using Markov's inequality.

Theorem 1 (Markov's inequality.)**.** *Let X be a non negative random variable, then for all $a > 0$,*

$$\mathbf{P}(X \geq a) \leq \frac{\mathbb{E}[X]}{a} \ .$$

The unsatisfiability of $\mathcal{F}_k(n,rn)$ when $r \geq 2^k \ln 2$ is proved by setting $a = 1$ and letting X denote the number of satisfying assignments in $\mathcal{F}_k(n,rn)$ in Theorem 1: an arbitrary assignment satisfies the random formula with probability $(1-2^{-k})^{rn}$ since each one of the rn random clauses is satisfied independently with probability $(1-2^{-k})$; there are a total of 2^n assignments, therefore, by linearity of expectation $\mathbb{E}[X]$ is $(2(1-2^{-k})^r)^n$, a quantity that tends to zero exponentially fast when $r \geq 2^k \ln 2$. On the other hand, Chao and Franco [14] proved that if $r < 2^k/k$ then a very simple algorithm finds a solution *w.u.p.p.* These results show that the ratio of constraints-to-variables $r = m/n$, known as the constraint density, plays a fundamental role when studying the satisfiability of random k-CNF formulas.

It is believed that $g_k(n, rn)$ has a step like shape, that is, the transition from being very close to 1 to being almost 0 is very sharp with respect to r:

Conjecture 2. *[Satisfiability Threshold Conjecture] For each $k \geq 3$, there exists a constant r_k^* such that for any $\epsilon > 0$,*

$$\lim_{n\to\infty} g_k(n, (r_k^* - \epsilon)n) = 1, \quad \textit{and} \quad \lim_{n\to\infty} g_k(n, (r_k^* + \epsilon)n) = 0 \ .$$

Experimental and theoretical work [5, 23, 49] suggest that the computational complexity of many satisfiability algorithms also has a swift increase in a small window in r. The possible connection between sharp transitions and computational complexity, which motivates a large part of our work, has attracted a lot of attention in computer science, mathematics and statistical physics [44, 45, 43].

1.2 Satisfiability Bounds

At this point, neither the value, nor even the existence of r_k^* has been established. In a breakthrough result, Friedgut [26] gave a very general condition for a monotone property to have a sharp threshold. In particular, his result yields the statement of conjecture 2 if one replaces r_k^* with a function $r_k^*(n)$.

Theorem 3. *For each $k \geq 3$, there exists a sequence $r_k^*(n)$ such that for any $\epsilon > 0$,*

$$\lim_{n\to\infty} g_k(n, (r_k^*(n) - \epsilon)n) = 1, \quad \textit{and} \quad \lim_{n\to\infty} g_k(n, (r_k^*(n) + \epsilon)n) = 0 \ .$$

The last decade has seen a great deal of rigorous results on random CSPs, including a proliferation of upper and lower bounds for the satisfiability threshold of a

number of problems. The best upper bound for random k-SAT is

$$r_k^* \leq 2^k \ln 2 - \frac{1 + \ln 2}{2} + o(1)$$

due, independently, to Dubois and Boufkhad [21] and to Kirousis et. al. [34]. The proof is based on a clever application of the first moment method in which they focus on the random variable that counts only the number assignments in which no 0 can be switched to a 1 without violating satisfiability–locally maximum satisfying assignments. This choice of assignments works because every satisfiable formula has at least one locally maximum satisfying assignment, for example, the lexicographical greatest one.

The original method to find lower bounds for the satisfiability threshold of random k-SAT was to prove that an algorithm finds a solution *w.u.p.p.* (see [3] for a review). However, the best bounds found this way were of order $O(2^k/k)$. Later, better bounds were found by proving the existence of solutions by analyzing statistical properties of the entire solution space instead of trying to find an individual solution. In particular, Achlioptas and Peres [9] used the second moment method to prove the following lower bound

$$r_k^* \geq 2^k \ln 2 - (k+1)\frac{\ln 2}{2} - 1 + o(1)$$

The second moment method is used to get lower bounds on $\Pr(X > 0)$ for a non-negative random variable X. It is based on the following inequality.

Theorem 4. *For any non-negative random variable X,*

$$\Pr[X > 0] \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}$$

To use the above theorem to prove that $\mathcal{F}_k(n, rn)$ is *w.u.p.p.* satisfiable, one has to find a random variable $X(\mathcal{F})$ such that 1) $X > 0$ if and only if $\mathcal{F}_k(n, rn)$ is satisfiable and 2) $\mathbb{E}[X^2] = O(\mathbb{E}[X]^2)$ i.e., X must concentrate, to some extent, around its mean. Achlioptas and Peres applied the second moment successfully by focusing on satisfying assignments in which the number of satisfied literals is $km/2 + O(n)$, which they labeled as balance assignments. For more details on this result see[9].

The current gap between the densities for which algorithms find solutions *w.u.p.p.* $(O(2^k/k))$ and the densities at which solutions exist $(2^k \ln 2 - \Theta(k))$ raised the question of whether there exists an 'algorithmic' barrier below the satisfiability threshold that causes certain algorithms to fail. To properly address this question we first need to show that algorithms indeed behave poorly in the satisfiable regime. In this thesis we prove for the first time exponential running time for a large class of DPLL algorithms (described in section 1.2.2) at densities below the best known lower bound for the satisfiable threshold of random 3-SAT. The proof comes from getting new upper bounds on the satisfiability of formulas composed of a mixture of random 2- and 3-clauses.

1.2.1 The $k = 2$ case and the Unit Clause Propagation Algorithm

A very simple algorithm known as unit clause propagation (UCP) (see algorithm 1), solves 2-SAT in linear time. UCP finds a solution by building a partial assignment step by step. It commits to the assignments made at each step and operates on a *residual formula*, in which clauses already satisfied have been removed, while the remaining

clauses might have been shortened by the removal of a falsified literal. This in turn will create additional *forced* choices in the partial assignment, since the variables appearing in unit clauses have only one possible assignment if the formula is to be satisfied (step 4). The resulting tree of implications will finish either because a) the appearance of a contradiction i.e., the appearance of a 0-clause (step 6 is evaluated to TRUE) or because b) the lack of 1-clauses. In case a), the algorithm tries now to assign the opposite value to x (step 7), if this also results in a contradiction then the formula is UNSAT. In case b), a partial assignment has been found and it can be extended to a full satisfying assignment if and only if the formula is satisfiable because any satisfying assignment restricted to the clauses still present in the formula is compatible with the current partial assignment. This argument shows why there is no need to reconsider variables already analyzed.

A very useful representation of a 2-SAT formula F is the directed graph $D(F)$ where the set of vertices correspond to all possible $2n$ literals and for every clause, say $(x_1 \wedge \bar{x}_4)$, there is a pair of edges, $x_4 \rightarrow x_1$ and $\bar{x}_1 \rightarrow \bar{x}_4$, corresponding to the implications resulting by assigning to each variable the values that do not satisfy the clause. Observe that UCP fails if and only if there is a directed cycle in $D(F)$ containing a literal and its complement. Thus, the satisfiability of random 2-CNF formulas can be studied by analyzing the appearance of such directed cycles on random di-graphs. Using this connection Chvatal and Reed [15], Goerdt [29] and Fernandez de la Vega [22] independently proved that $r_2^* = 1$.

Algorithm 1 UCP(F)

1: **if** F is empty **then**
2: **return** SAT
3: **end if**
4: Select a variable $x \in F$
5: Let F' be the formula resulting from recursively satisfying all 1-clauses in $F_{x=0}$
6: **if** F' has a 0-clause (a contradiction) **then**
7: Let F' be the formula resulting from recursively satisfying all 1-clauses in $F_{x=1}$
8: **if** F' has a 0-clause (a contradiction) **then**
9: **return** UNSAT
10: **end if**
11: **end if**
12: **return** UCP(F')

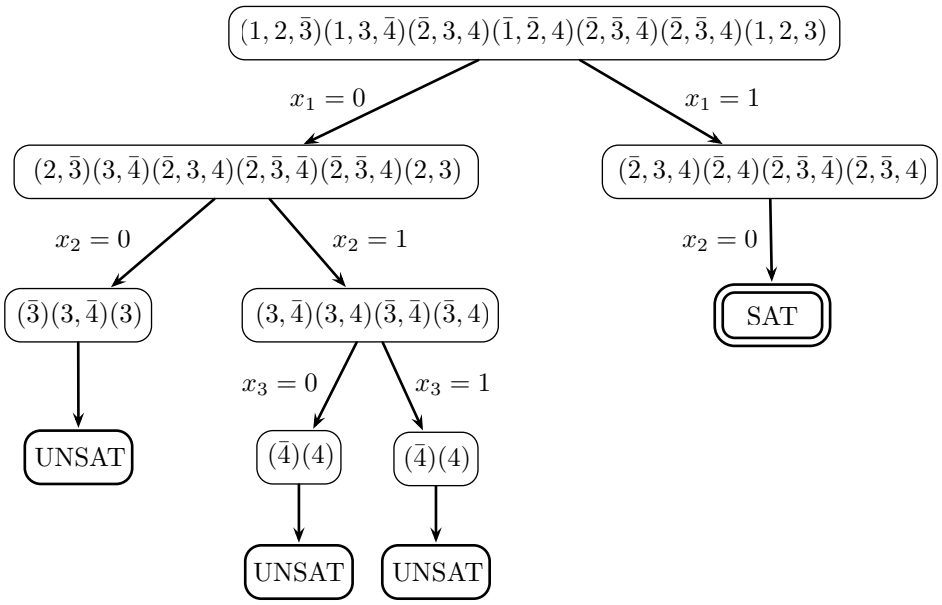

Figure 1.1: Search tree resulting from running ORDERED-DPLL on the instance $(1,2,\bar{3}) \wedge (1,3,\bar{4}) \wedge (\bar{2},3,4) \wedge (\bar{1},\bar{2},4) \wedge (\bar{2},\bar{3},\bar{4}) \wedge (\bar{2},\bar{3},4) \wedge (1,2,3)$, where we used $(1,2,3)$ to denote the clause $(x_1 \vee x_2 \vee x_3)$, and so on, to simplify notation.

1.2.2 The $k = 3$ case and DPLL algorithms

For $k \geq 3$ satisfiability is an NP-complete problem. For example, applying UCP to a 3-CNF might not find a solution even if the formula is satisfiable, because fixing the value of a variable could result in a residual formula containing clauses of size 2 not present in the original formula, i.e. the extension of the current partial assignment to a full assignment is no longer guaranteed.

The UCP algorithm can be modified so that when a contradiction is found the algorithm backtracks to the last choice for which one possibility was not explored. This will produce a decision tree where the nodes are residual formulas and each edge corresponds to a particular assignment of a variable. Different algorithms will result

depending on the rule used to select the next variable to be fixed given the current residual formula. This type of algorithms are known as DPLL[2] search algorithms. The general DPLL algorithm is given in Algorithm 2. Figure 1.1 shows the search tree resulting from applying a DPLL algorithm where the variables are selected in a predefined order and always assigned to the value 0 (known as ORDERED-DPLL) to the 3-CNF formula $(1,2,\bar{3})(1,3,\bar{4})(\bar{2},3,4)(\bar{1},\bar{2},4)(\bar{2},\bar{3},\bar{4})(\bar{2},\bar{3},4)(1,2,3)$, where we used $(1,2,3)$ to denote the clause $(x_1 \vee x_2 \vee x_3)$, and so on, to simplify notation.

Algorithm 2 DPLL(F)

1: Repeatedly satisfy any pure literals and 1-clauses.
2: **if** The resulting formula F' is empty **then**
3: **return** SAT
4: **end if**
5: **if** A contradiction (0-clause) is generated **then**
6: **return** UNSAT
7: **end if**
8: Select a variable $x \in F'$ and a value v for x
9: **if** DPLL($F'_{x=v}$) == UNSAT **then**
10: **return** DPLL($F'_{x=\neg v}$)
11: **else**
12: **return** SAT
13: **end if**

[2]Because the initials of their inventors: Davis, Putman, Logemann and Loveland

Most of the lower bounds for the satisfiability threshold of random 3-SAT, including the best bound found so far of $3.52 < r_3^*$, due to Diaz et. al. [19], come from showing that different versions of DPLL algorithms find a solution *w.u.p.p.* during the first branch of the search tree. The proofs are based on the analysis of the evolution of the number of clauses of sizes 2 and 3 present in the residual formulas produced during the execution of the algorithms.

Below we present the general intuition behind such analysis (see [3, 10] for more details). The analysis of the trajectory of 2- and 3-clauses is mathematically tractable when the studied DPLL algorithms satisfy the *uniform randomness* property: as long as the algorithm has never backtracked, the residual formula is uniformly random conditional on its number of 2- and 3-clauses (unit-clauses are satisfied as soon as they occur). We will refer to the family of DPLL algorithms satisfying this property as *myopic* algorithms.

Using the principle of deferred decisions, it is possible to show that an algorithm that satisfies the following properties is *myopic* [3]:

1. The decision of which variable to set is based only on the lengths of the clauses in which each variable participates.

2. Having decided which variable to set, the decision on how to set it depends only on the clauses containing the variable.

The simplest example myopic algorithm ignores any information about the residual formula and simply picks the next literal uniformly at random among the literals in

all current unassigned variables. It is known as Unit Clause (UC) (perhaps for being the simplest extension of UCP to a full backtracking algorithm). Another example is GENERALIZED UNIT CLAUSE (GUC) [27, 14], where in each step a random literal in a random shortest clause is assigned TRUE.

Observe that the trajectory of the 2- and 3-clause densities in the residual formulas produced by a myopic algorithm is a Markov chain. For example, the mean path of the Markov chain corresponding to the UC algorithm can be derived as follows: let $C_2(t)$ and $C_3(t)$ denote the number of 2- and 3-clauses after fixing t of the n variables and let $\mathrm{Bin}(N, s)$ denote the binomial random variable with N trials and probability of success s. The $t+1$ variable x_{t+1} to be fixed by UC is picked at random among the remaining $n-t$ variables and because UC is myopic, x_{t+1} appears independently in each of the current $C_3(t)$ 3-clauses with probability $3/(n-t)$; thus, the number X of 3-clauses to be removed from the current residual formula is distributed as $\mathrm{Bin}(C_3(t), 3/(n-t))$. Similarly, the number Z of 2-clauses to be removed from the current residual formula is distributed as $\mathrm{Bin}(C_2(t), 2/(n-t))$. Finally, with probability $1/2$ each of the 3-clauses in which x_{t+1} appears is unsatisfied by the random assignment of x_{t+1} and becomes a 2-clause i.e. the number Y of new 2-clauses in the residual formula is distributed as $\mathrm{Bin}(C_3(t), 3/(2(n-t)))$. In summary, the evolution of the vector $(C_2(t), C_3(t))$ is modeled by the following system of stochastic equations:

$$\begin{aligned} C_3(t+1) - C_3(t) &= -X \\ C_2(t+1) - C_2(t) &= Y - Z \ , \end{aligned}$$

where $X \sim \text{Bin}(C_3(t), 3/(n-t))$, $Y \sim \text{Bin}(C_3(t), 3/(2(n-t)))$ and $Z \sim \text{Bin}(C_2(t), 2/(n-t))$. The mean path equations of the above stochastic process are

$$\begin{aligned} \mathbb{E}[C_3(t+1) - C_3(t)] &= -\frac{3C_3(t)}{n-t} \\ \mathbb{E}[C_2(t+1) - C_2(t)] &= \frac{3C_3(t)}{2(n-t)} - \frac{2C_2(t)}{n-t} \end{aligned}$$

with

$$C_3(0) = rn \quad \text{and} \quad C_2(0) = 0 \ ,$$

The actual analysis of the trajectory of the 2- and 3-clause densities of the residual formula is done by approximating the mean path of the Markov chain associated with each algorithm with their corresponding set of differential equations. The justification is that in the large n limit the mean path is highly concentrated for all $t \leq t_e = (1-\epsilon)n$, where $\epsilon > 0$ is an arbitrary small constant, i.e. both of these densities behave as deterministic functions for every myopic algorithm [3]. This means that in the absence of backtracking, i.e., if the algorithm continues on after a 0-clause is generated, we can model the algorithm's behavior as a continuous 2-dimensional curve $(d_2^r(x), d_3^r(x))$ of the 2- and 3-clause density, where $x \in [0, t_e/n]$ denotes the fraction of assigned variables and r is the initial 3-clause density.

To pass to the realm of differential equations, we replace differences with derivatives in the normalized version of the mean path equations: if we let $x \equiv t/n$ and $c_k(x) \equiv C_k(xn)/n$ for $k = 2, 3$, then the resulting system of differential equations for UC is:

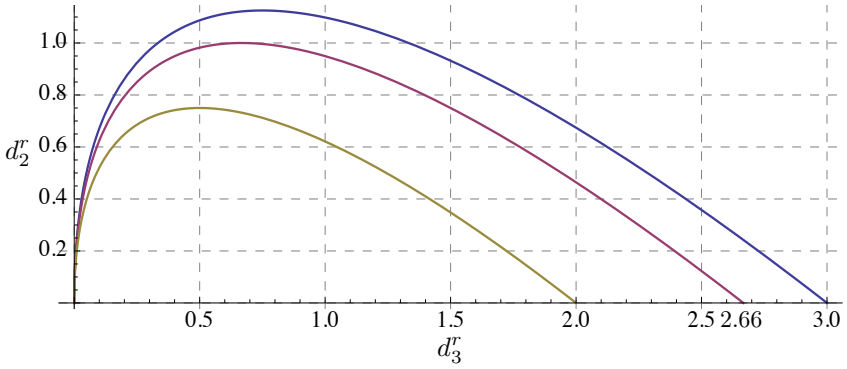

Figure 1.2: Mean path followed by UC for initial densities $r = \{2, 8/3, 3\}$.

$$\begin{aligned} \frac{dc_3}{dx} &= -\frac{3c_3(x)}{1-x} \\ \frac{dc_2}{dx} &= \frac{3c_3(x)}{2(1-x)} - \frac{2c_2(t)}{1-x} \end{aligned}$$

with

$$c_3(0) = r \quad \text{and} \quad c_2(0) = 0 \ .$$

The solution of the above system of differential equations is $c_3(x) = (1-x)^3$ and $c_2(x) = \frac{3}{2}rx(1-x)^2$. Note that $c_3(x)$ and $c_2(x)$ is the number of clauses of sizes 3 and 2 normalized by the initial number of variables and not the actual densities $d_2^r(x)$ and $d_3^r(x)$ of 2- and 3-clauses in the residual formula after a fraction x of variables have been fixed; multiplying $c_2(x)$ and $c_3(x)$ by $n/(n-t)$ recovers the correct densities $d_2^r(x) = c_2(x)n/(n-t) = \frac{3}{2}rx(1-x)$ and $d_3^r(x) = c_3(x)n/(n-t) = (1-x)^2$ (see figure 1.2 for the parametric plot of $(d_2^r(x), d_3^r(x))$ for initial densities $r = \{2, 8/3, 3\}$).

Since the 2-SAT satisfiability threshold is $r_2 = 1$, it follows that for any initial 3-clause density $r > 0$ and every $\tau > 0$ such that $d_2^r(x) < 1$ for all $x \in [0, \tau)$, the

probability that no 0-clause is ever generated is bounded away from 0 i.e., a myopic algorithm finds a solution *w.u.p.p.* for a given initial density r if $d_2^r(x) < 1$ for all $x \in [0,1)$. Thus, the best lower bound, $r_{\mathcal{A}}$, for the satisfiability threshold that can be obtained with a myopic algorithm $\mathcal{A}$ is given by the largest r such that $d_2^r(x) < 1$ for all $x \in [0,1)$. For UC this value is $8/3$.

The reason that the above results hold only *w.u.p.p.* is that each algorithm might generate a contradiction and backtrack destroying the uniform randomness of the formula. On the other hand, it is possible to show that when the density of the 2-clauses is below 1 i.e. when $r < r_{\mathcal{A}}$ this occurs *w.h.p.* for trivial local reasons that typically can be fixed by changing $O(1)$ of the variables. Frieze and Suen [27] introduced a meta-heuristic that can be added on top of myopic algorithms to maintain uniform randomness when backtracking occurs for local reasons: when a contradiction is reached, record the portion of the assignment between the last free choice and the contradiction; these literals become *hot*. After flipping the value of the last free choice, instead of making the choice that the original heuristic would suggest, give priority to the complements of the hot literals in the order that they appeared; once the hot literals are exhausted continue as with the original heuristic. From here on we assume that Frieze's meta-heuristic is being implemented.

The uniform randomness property can be extended [33], for example, by guarantying uniform randomness of the residual formula conditional on its number of 2- and 3-clauses and on its degree sequence. Indeed, the best bound found so far of $3.52 < r_3^*$, due to Diaz et. al. [19] was proved using this form of uniform randomness.

Let us now consider what happens if one gives as input to a myopic algorithm $\mathcal{A}$ a random 3-CNF formula of density $r > r_{\mathcal{A}}$, but only runs the algorithm for $x_0 \cdot n$ steps where x_0 is such that $d_2^r(x) < 1$ for all $x \in [0, x_0)$. Up to that point, the algorithm will have either not backtracked at all–or backtracked for trivial local reasons–so that the residual formula will be a mixture of random 2- and 3-clauses in which the 2-clauses alone are satisfiable. Naturally, if the residual formula is satisfiable the algorithm still has a chance of finding a satisfying assignment in polynomial time. But what happens if this mixture, as a whole, is unsatisfiable? How fast will it discover this and backtrack? The answer to this was was given in [4] by exploiting the relation between the resolution complexity of random formulas and the search trees produced by DPLL algorithms; we discuss this relation in the next section.

1.2.3 Resolution Complexity of Random k-CNF Formulas

Propositional logic formalizes simple models of reasoning that have been used in mathematics and philosophy for centuries. One of the most important tasks is to verify that a boolean formula is a tautology (evaluates to TRUE on every assignment) or equivalently, to verify that its complement is a contradiction (has no satisfying assignments).

Resolution is a procedure aimed to produce a proof that a formula is unsatisfiable. It is based on the resolution rule from logic which allows one to logically infer the clause $(A \vee B)$ from the clauses $(x \vee A)$ and $(\bar{x} \vee B)$–any truth assignment satisfying $(x \vee A)$ and $(\bar{x} \vee B)$ will also satisfy $(A \vee B)$ (see Table 1.1). A resolution derivation of

x	A	B	$x \vee A$	$\bar{x} \vee B$	$(x \vee A) \wedge (\bar{x} \vee B)$	$(A \vee B)$
0	0	0	0	1	0	0
0	0	1	0	1	0	1
0	1	0	1	1	**1**	**1**
0	1	1	1	1	**1**	**1**
1	0	0	1	0	0	0
1	0	1	1	1	**1**	**1**
1	1	0	1	0	0	1
1	1	1	1	1	**1**	**1**

Table 1.1: Resolution rule

a clause C from a CNF formula F is a sequence of clauses $C_1, C_2, \ldots, C_l = C$ such that each C_i is either a clause of F or follows from two clauses C_j and C_k for some $j, k < i$ using the resolution rule. A resolution proof of the unsatisfiability of F is a resolution derivation of the empty clause. The resolution complexity of a formula F, denoted by $res(F)$, is the size of the minimum resolution proof of unsatisfiability.

Any DPLL search tree of an unsatisfiable formula F can be converted to a resolution refutation proof of F where the pattern of inferences follows the structure of the tree from the leaves to the root as follows: 1. Label each leaf of the tree with the result of applying the resolution rule to the original clauses of F that were shown to be contradictory on that branch of the tree. For example, in Figure 1.3, the label above the left most leaf is the clause $(x_1 \vee x_2)$ which results from the resolution between the clauses $(x_1 \vee x_2 \vee \bar{x}_3)$ and $(x_1 \vee x_2 \vee x_3)$; 2. Label each internal node with the result of applying the resolution rule with the clauses labeling its children nodes. Using induction, one can show that this process generates an empty clause at the root when F is unsatisfiable i.e. a resolution refutation proof of F. In conclusion, lower bounds

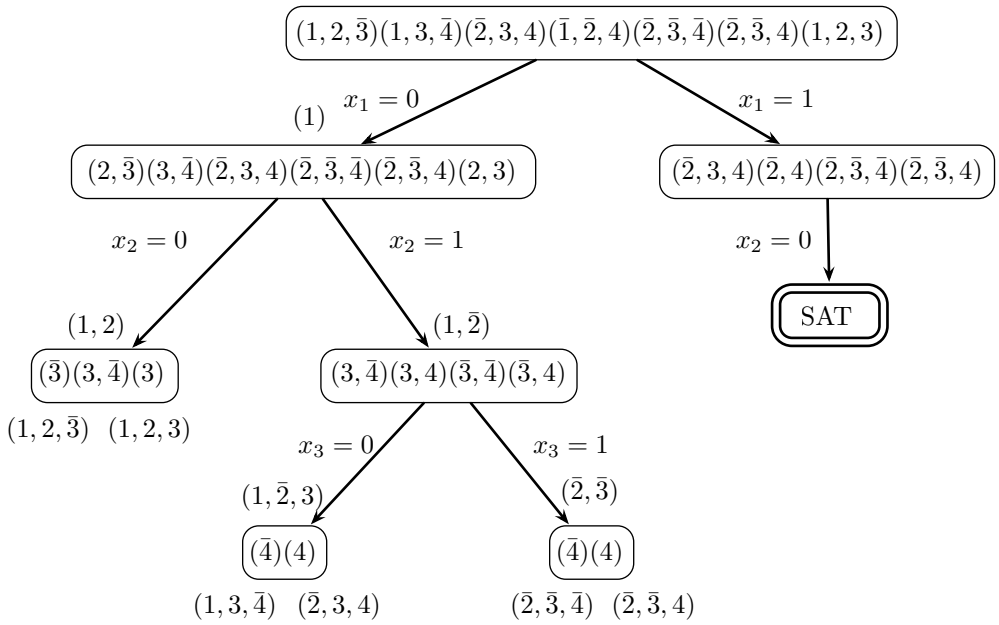

Figure 1.3: Resolution derivation for the unsatisfiability of the residual formula generated after setting $x_1 = 0$.

on the resolution complexity translate into lower bounds on the running time of DPLL algorithms.

It is well known that the resolution complexity of 2-SAT is linear in size. Random k-CNF formulas and sparse random CSPs in general are locally like a tree; the neighborhood of each variable is highly under-constrained but at the same time they are highly connected as a whole i.e., they are good expanders[3] if one prunes low degree vertices. In a seminal paper Chvatal and Szemeredi [16] used the expansion properties of random k-CNF formulas to establish that for $k \geq 3$ their resolution complexity is $(1+\epsilon)^n$ for some $\epsilon = \epsilon(k,r) > 0$, thus causing every DPLL algorithm to take *w.h.p.* exponential time on random k-CNF formulas in the unsatisfiable regime. On the other

[3]See [32] for an excellent review on expander graphs and its applications.

hand, adding $(1+\epsilon)n$ random 2-clauses to a random 3-CNF formula causes its resolution complexity to collapse from exponential to linear *w.h.p.*, since a formula with $(1+\epsilon)n$ random 2-clauses is by itself unsatisfiable *w.h.p.,* because the satisfiability threshold for random 2-CNF formulas is 1. Later, Achlioptas et. al. [5] proved that adding $(1-\epsilon)n$ random 2-clauses *w.h.p.* has no effect in the resolution complexity. We can summarize these results in terms of the running time of DPLL algorithm as follows:

Theorem 5.

1. *Any* DPLL *algorithm takes w.h.p. linear time on random formulas composed of* $(1+\epsilon)n$ *random* 2*-clauses and* Δn *3-clauses for any* $\epsilon, \Delta > 0$.

2. *Any* DPLL *algorithm takes* w.h.p. *exponential time on* **unsatisfiable** *random formulas composed of* $(1-\epsilon)n$ *random* 2*-clauses and* Δn *3-clauses with* $\epsilon, \Delta > 0$.

Theorem 5 highlights the importance of determining the satisfiability of formulas composed of a mixture of $(1-\epsilon)n$ 2-clauses and Δn 3-clauses for studying the running time of general DPLL algorithms on random 3-CNF formulas.

The expansion properties of random k-SAT formulas are at the root of the exponential lower bounds for the running time of general DPLL algorithms and we believe that the same expansion properties can explain similar drops in the efficiency of many other classes of algorithms. The idea is that when a constraint factor graph is an expander, difficulty arises from long range interactions that can not be explicitly represented in the instance.

1.2.4 Unsatisfiability of Mixture Formulas

Let $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$ denote a random CNF formula over n variables consisting of $(1-\epsilon)n$ random 2-CNF clauses and Δn random 3-CNF clauses. The equations describing the mean path of the density of 2- and 3-clauses on the residual formulas produced by the UC algorithm show that if the initial density of 3-clauses is $8/3$, then the trajectory is tangent to the $r_2 = 1$ line when the density of 3-clauses is $2/3$ (see figure 1.2), suggesting that UC will find a solution *w.h.p.* on mixtures with $\epsilon > 0$ and $\Delta \leq 2/3$. In [6], Achlioptas et. al. used these arguments to prove the following:

Theorem 6. *A random CNF formula with n variables, $(1-\epsilon)n$ 3-clauses and Δn 3-clauses is w.h.p. satisfiable for all $\epsilon > 0$ and all $\Delta \leq 2/3$.*

The same analysis can be done using different DPLL algorithms. Intriguingly, the mean path equations of all of the algorithms studied so far also give $\Delta = 2/3$ when the curve is tangent to the $r_2 = 1$ line. The intuition is that when the density of 2-clauses is very close to 1 and the density of 3-clauses approaches $2/3$, the formula is so constrained that even the most complex heuristics are no better than the rule that simply chooses a variable uniformly at random. Smarter heuristics find assignments for bigger initial densities because they delay the generation of these highly constrained mixtures. This indicates that for $\Delta > 2/3$ algorithms fail to find solutions simply because none exist, i.e.,

Conjecture 7. *For all $\Delta > 2/3$, there exist $\epsilon = \epsilon(\Delta) > 0$, such that a random CNF formula with n variables, $(1-\epsilon)n$ 2-clauses and Δn 3-clauses is w.h.p. unsatisfiable.*

If Conjecture 7 is true, then the running time of myopic DPLL algorithms exhibit a sharp threshold behavior in its running time at a density below the satisfiability threshold: for each myopic algorithm $\mathcal{A}$, let $r_A < r_3^*$ denote the initial density that produces a mean path that passes through $\Delta = 2/3$. Then $\mathcal{A}$ takes linear time for $r < r_{\mathcal{A}}$ and exponential time for $r > r_{\mathcal{A}}$ since $\mathcal{A}$ will produce *w.h.p.* a formula with the characteristics of 2) in Theorem 5.

Let $\Delta^* = \sup\{\Delta | \mathcal{F}_{2,3}(n, \epsilon, \Delta)$ is satisfiable *w.h.p.* for all $\epsilon > 0\}$. Theorem 6 shows that $2/3 \leq \Delta^*$ and Conjecture 7 claims that $2/3$ is tight. The best upper bound found before this thesis is $\Delta^* \leq 2.28$ due to Achlioptas et. al. [6]. Using this bound and the same mean path analysis shows that UC and GUC take exponential time for $r \geq 3.81$ and $r \geq 3.98$, respectively. Unfortunately, densities below 3.98 are only conjecture to be satisfiable i.e., it was not possible to claim exponential lower bounds on the running time of DPLL algorithms on *satisfiable* formulas.

In this work we take a technique from mathematical physics, the *interpolation method,* a remarkable tool originally developed by Guerra and Toninelli [31], and use it to derive end-to-end rigorous explicit upper bounds for the satisfiability threshold of a number of problems. To do so, we introduce a new version of the interpolation method that can be made computationally effective and extends the method to CSPs with arbitrary degree distributions [8].

Our method can be used to prove that $\Delta^* \leq 1.001$ [7], i.e.,

Theorem 8. *Let F be a random CNF formula on n variables with $(1-\epsilon)n$ random 2-clauses and $(1+\epsilon)n$ random 3-clauses. Then, F is unsatisfiable w.h.p. for $\epsilon = 10^{-4}$.*

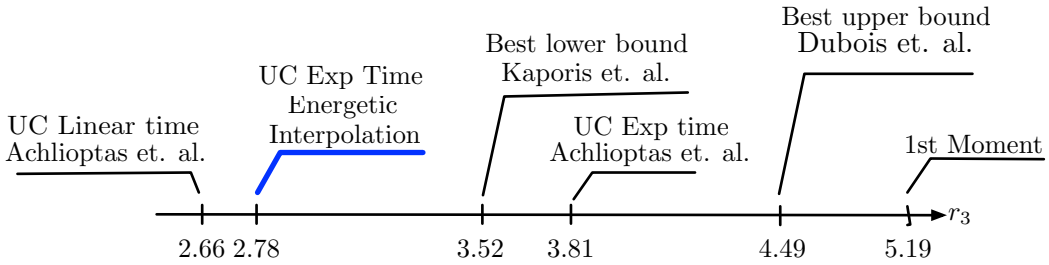

Figure 1.4: Rigorous results for the running time of the UC algorithm on random 3-CNF formulas in the context of the best known upper/lower bounds for the satisfiability threshold of random 3-SAT.

By improving $\Delta^* \leq 2.28$ to $\Delta^* \leq 1.001$ we establish that algorithms take exponential time on random 3-CNF formulas with densities which are not only in the provably satisfiable regime, but in fact not much greater than the densities for which the same algorithms find solutions in linear time. For example, UC succeeds in linear time for $r \leq 8/3 = 2.66...$. Our results imply that it requires exponential time for all $r \geq 2.78$, compared to 3.81 from [5] (see Figure 1.4). Similarly, for GUC while linear-time success is guaranteed for $r \leq 3.003$, we prove that exponential time is required for $r \geq 3.1$, compared to 3.98 from [5].

Similar results hold for other myopic algorithms, including, for example, all algorithms analyzed in [3] and [2]. Moreover, our extension of the proof to CSP on arbitrary degree sequences could be used to develop an analogous machinery to study myopic algorithms satisfying more complex forms of uniform randomness.

1.3 Past Work on the Interpolation Method for Random CSPs

Some of the most useful tools developed to study CSPs and random CSPs have come from statistical mechanics. The Markov chain Monte Carlo (MCMC) method [28, 48] and the use of message passing algorithms [38, 41] are two prominent examples. The tools we present in this thesis to compute upper bounds for the satisfiability threshold of random CSPs could be seen as the result of a rigorous analysis of message passing algorithms on random CSPs.

1.3.1 The Statistical Physics Approach to Random CSPs

In the statistical physics approach, the set of all assignments of a CSP is endowed with the Gibbs probability measure

$$P(\sigma|\beta, I) = \frac{1}{Z} e^{-\beta H(\sigma|I)} \ , \tag{1.1}$$

where $H(\sigma|I)$ is known as the energy function or Hamiltonian, that counts the number of unsatisfied constraints in the instance I under the assignment σ, $\beta > 0$ is a constant, and Z, known as the partition function, is the constant that makes $P(\sigma|\beta, I)$ a valid probability density

$$Z(\beta) = \sum_{\sigma \in D^n} \exp(-\beta H(\sigma|I)) \ .$$

The parameter β can be seen as a softening parameter that interpolates between the uniform distribution over the set of all assignments when $\beta \to 0$ to the uniform distri-

bution over the set of all *satisfying* assignments when $\beta \to \infty$. In other words, taking a random sample form the distribution $P(\sigma|\beta, I)$ can be viewed as a statistical soft-min operation that gets closer and closer to a hard min as β tends to infinity–what physicists call the temperature going to zero.

Now consider a random CSP, $\mathcal{I}(n, rn)$, formed by picking $m = rn$ constrains uniformly, independently and with replacement from a set C_n of constraints. For random k-SAT, $C_n = B_{k,n}$ denotes the set of all possible disjunctions of k distinct, non-complementary literals from its variables, and for random hyper-graph coloring C_n is the set of all possible k-element subsets of the set of n vertices. Now, the Gibbs distribution of the random instance $\mathcal{I}(n, rn)$ involves two levels of randomness. The first is due to the random nature of the CSP instance itself and the second is from endowing the instance with the Gibbs measure. A central goal in the statistical physics approach is to compute the quantity

$$f(\beta|r) \triangleq \lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\log\left(\sum_{\sigma\in D^n} \exp(-\beta H(\sigma|\mathcal{I}(n, rn)))\right)\right] \quad , \tag{1.2}$$

known as the *free entropy density*, where the expectation is taken with respect to the randomness of the random instance $\mathcal{I}(n, rn)$. Let us examine (1.2) carefully as it reveals a lot of what we are trying to achieve. First consider the case in which $H(\sigma|I)$ is not random, but the energy function of a single CSP instance I–for example, the number of unsatisfied clauses in a k-CNF formula with n variables and rn clauses. Since the Gibbs distribution weighs each assignment only through its energy, it is revealing to partition the set of assignments D^n into classes according to their energy

$H(\cdot|I)$ and then write the sum in $f(\beta|r)$ in terms of those classes; to this effect, let $N(a) = |\{\sigma \in D^n : a = H(\sigma|I)\}|$, then

$$\sum_{\sigma \in D^n} \exp(-\beta H(\sigma|I)) \quad = \quad \sum_{a=0}^{\infty} N(a) \exp(-\beta a) \ .$$

Note that the number of assignments grows exponentially with n, whereas the number of classes grows at most linearly with n, thus $N(a)$ is exponential for at least one value of a. Therefore it will be convenient to write $N(a)$ as $\exp(n \cdot s(a/n))$ where $s(a/n) \equiv \ln N(a)/n$ and the sum is then

$$\sum_{x \in \{(a/n)|a \in \mathbb{N}\}} \exp(n(s(x) - \beta x)) \ .$$

In the $n \to \infty$ limit, the above sum is dominated by $\sup\{\exp(n(s(x) - \beta x))\}$, i.e., by those value assignments having energy in some narrow window that depends on β. The idea being that assignments violating more constraints are penalized too heavily to contribute significantly to the sum, while assignment violating even fewer constraints are too rare to have substantial contribution. In this context β plays the role of exchange rate between energy density x and entropy (number of assignments at a given energy) density $s(x)$. Note that as β is increased, the function $f(\beta|r)$ places more and more weight to assignments violating fewer constraints, recovering the number of solutions as $\beta \to \infty$. Taking the log and normalizing is then just a procedure to extract the quantity of interest $\sup_x\{s(x) - \beta x\}$. The function $s(\cdot)$ captures a lot of relevant information about the CSP. For example if $s(0) = 0$ then the CSP is unsatisfiable. However, it is very hard to derive, in fact, for most of the cases one approximates $f(\beta|r)$ to gain access to $s(x)$ through the Legendre transform $\sup_\beta\{f(\beta) - \beta x\}$.

When the CSP is random, martingale arguments imply that if for any finite $\beta > 0$ we have $f(\beta|r) < 0$, then *w.h.p.* no solutions exist, i.e., upper bounds for $f(\beta|r)$ that are less than zero can be used to compute upper bounds for the satisfiability threshold of the random CSP at hand. Let

$$F(\beta|n,r) = \log\left(\sum_{\sigma\in D^n} \exp(-\beta H(\sigma|\mathcal{I}(n,rn)))\right) \quad .$$

A key property about the random variable $F(\beta|n,r)$ is that it concentrates around its mean in an $o(n)$ window. This concentration result is a direct consequence of the following theorem:

Theorem 9 (Independent bound differences Theorem [39][4])**.** *Let $X = (X_1, X_2, ..., X_n)$ be a family of independent random variables with X_i taking values in a set A_i for each i. Suppose that the real-valued function g defined on the product space $\prod A_i$ satisfies the Lipschitz condition*

$$\left|g(x) - g(x')\right| \leq c_i$$

whenever the vectors x and x' differ only on the ith coordinate. Then for any $t \geq 0$,

$$\mathbf{P}(g(X) - \mathbb{E}[g(X)] \geq t) \leq e^{-2t^2/\sum c_i} \tag{1.3}$$

and

$$\mathbf{P}(g(X) - \mathbb{E}[g(X)] \leq -t) \leq e^{-2t^2/\sum c_i} \quad . \tag{1.4}$$

If we let X correspond to the set of constraints of the random CSP, then $F(\beta|n,r)$ satisfies the Lipschitz condition with $c_i = \beta$, since the modification of a single

[4]Page 10, Theorem 3.1

constraint can increase the number of unsatisfied constraints of each assignment by at most one. Thus, if $\mathbb{E}[F(\beta|n,r)] < 0$, then by setting $t = -\mathbb{E}[F(\beta|n,r)] \geq 0$ in Equation (1.3) we get

$$\mathbf{P}(F(\beta|n,r) \geq 0) \leq e^{-2t^2/(\beta r n)} \ .$$

Consequently, if t increases at least linearly with n, then $\mathbf{P}(F(\beta|n,r) \geq 0)$ tends to zero, i.e., $F(\beta|n,r) < 0$ *w.h.p.;* but $f(\beta|r) < 0$ implies that[5] $t = \Omega(n)$, and $F(\beta|n,r) < 0$ implies that no solutions exist, therefore, r is an upper bound for the satisfiability threshold of $\mathcal{I}(n, rn)$.

1.3.2 The Cavity Method

The cavity method is a non-rigorous but mathematically sophisticated method from statistical mechanics used to approximate $f(\beta|r)$ [38]. It corresponds to an increasingly sophisticated hypothesis about the nature of the induced Gibbs measure. In the simplest approximation, known as the Replica Symmetric (RS) approximation, it is assumed that the distribution of the Gibbs measure is equivalent to the distribution of an instance whose factor graph is a tree. The next level of complexity, called One Step Replica Symmetry Breaking (1RSB), assumes that the distribution of the Gibbs measure can be expressed as a convex combination of RS distributions.

The cavity computations can be seen as a variational method in which the original distribution is being approximated by distributions of increasing complexity. It has been very hard to prove rigorous results about such approximations in the general

[5] We say $f = \Omega(g(n))$ if there is an $\epsilon > 0$ and an integer n_0 such that for all $n > n_0$ $|f(n)| > \epsilon g(n)$ [30]

case. Nonetheless, in the average case, some rigorous results have been obtained through the use of the *interpolation method*. In particular, Franz and Leone [24], in a very important paper, applied it to random k-SAT and random k-XOR-SAT to prove that the cavity method for these problems can, in principle, be used to derive upper bounds for the free entropy density of each problem. As we will see, though, doing so involves the solution of certain functional equations that appear beyond analytical penetration. Moreover, the results only hold for finite β, i.e., they can not be applied in the zero temperature limit. We will refer to this application of the interpolation method as the *entropic* interpolation method, since it is used to bound the free entropy density. In [47], Panchenko and Talagrand showed that the results of Franz and Leone can be derived in a simpler and uniform way, unifying the treatment of different levels of replica symmetry breaking.

Rather than working with the free entropy density, we work with the quantity

$$\xi_r \triangleq \lim_{n\to\infty} \frac{1}{n} \min_{\sigma\in D^n} H(\sigma|\mathcal{I}(n, rn)) \ ,$$

known as the *ground state energy density*. That is, rather than working with a soft min we work with the actual min. In this case, a rigorous lower bound on ξ_r that is bigger than zero implies the lack of solutions i.e. an upper bound for the satisfiability threshold. The use of the interpolation method to bound ξ_r, which we called the *energetic* interpolation method, results in expressions that are much simpler, specifically, they just require the solution of a 1-dimensional optimization problem, which typically turns out to be trivial, and as such can be used to compute explicit rigorous unsatisfiability

bounds. Unfortunately, this analytical tractability comes with the cost of accuracy. This means that for some problems our method will not deliver bounds as strong as those that, in principle, can be given by the *entropic* interpolation method. At the same time, though, we are the first to derive fully rigorous end-to-end explicit bounds via the interpolation method for random CSPs.

A crucial ingredient in all the above applications of the interpolation method is a Poissonization device exploiting that in Erdos-Renyi (hyper)graphs the degrees of the vertices behave like, essentially independent, Poisson random variables. Franz, Leone, and Tonnineli [25] extended the interpolation method to other degree sequences, but at the cost of introducing another level of complexity (multi-overlaps), thus placing the method even further out of reach in terms of explicit computations. In [46], Montanari gave a simpler method for dealing with degree sequences in the context of error-correcting codes, which proceeds by approximating the intended degree distribution "in chunks". This, unfortunately, requires the number of approximation steps to go to infinity (so that the chunk size goes to zero) in order to give results for the original problem. In this thesis we give a very simple and general device for extending (all flavors of) the interpolation method to arbitrary degree sequences, showing that to derive explicit bounds it suffices to work with a univariate CSP whose variables have the same degree distribution as the original CSP [8].

Finally, in a recent paper, Bayati, Gamarnik and Tetali [11], showed that a combinatorial analogue of the interpolation method can be used to elegantly derive an approximate subadditivity property for a number of CSPs on Erdos-Renyi and regular

random graphs. This allowed them to prove the *existence* of a number of limits on these problems. The simplicity of that approach, though, comes at the cost of losing the capacity to give any bounds for the associated limiting quantities.

The remaining of the thesis is organized as follows: In chapter 2, we give a uniform, highly explicit presentation of the interpolation method for a wide array of random CSPs, including random graph and hypergraph coloring, which is the first time that the method has been successfully applied to CSPs with non-binary domains. In our exposition, we highlight both the intuition behind the method and the underlying technical challenges in using the method to compute explicit bounds. We hope that our presentation makes the method accessible to a much wider community, both in combinatorics and theoretical computer science. In chapter 3, we introduce our simpler "energetic" flavor of the interpolation method and present the techniques used for extending (all flavors of) the interpolation method to arbitrary degree sequences. In chapter 4, we prove Theorem 8, thus completing our proof about the exponential running time of myopic DPLL algorithms on satisfiable formulas. Finally, in chapter 5, we give the conclusions.

Chapter 2

The Entropic Interpolation Method

In this chapter we present the original entropic interpolation method (EIM) as applied to random CSPs to obtain upper bounds for the free entropy density [47, 24]. These bounds coincide with the expressions obtained with the RS cavity method approximation[38, 41]. The exposition presented in this chapter, however, can be read without having any knowledge about the cavity method.

Our goal is to capture the bare bones of the method, for example, we observed that one of the assumptions presented in the original papers [47, 24] is not necessary. This allowed us to successfully apply the EIM to random k-uniform hyper-graph $q$coloring which is the first time that the method has been applied to random CSP with non-binary variables.

2.1 The Entropic Interpolation Method

For simplicity of exposition we focus on the case where all constraints have the same arity[1] $k \geq 2$. Let $C_{k,n}$ denote the set of all possible k-constraints on n variables for the CSP at hand and D denote the domain of each variable. For random k-SAT, $C_{k,n}$ denotes the set of all possible disjunctions of k literals from its variables, and for random hyper-graph coloring, $C_{k,n}$ is the set of all possible k-element subsets of the set of n vertices. Here we will use a slightly different definition of a random CSP than the one used in the introductory chapter. A random CSP instance $\mathcal{I}_k(n, m)$ is still a conjunction of m constraints taken independently with replacement from the set $C_{k,n}$, but now m is a *Poisson random variable* with mean $\mathbb{E}[m] = rn$. Note that in the previous model m is fixed (not a random variable). Since, the standard deviation of the Poisson distribution is the square root of its mean we have $m = (1 + o(1))rn$ *w.h.p.* Therefore, this change does not affect any asymptotic results regarding densities. At the same time, as we will highlight, along with the Poissonization of the variable degrees, this is key to the original development of the method. Eliminating the need for Poisson variable degrees and allowing arbitrary (sparse) degree sequence, as we do in Section 3.2, is part of the technical contribution in our work.

As we described in Chapter 1, one way to compute upper bounds for the satisfiability threshold is by computing negative upper bounds for the free entropy

[1]Later, in chapter 4, we extend the method to work with constraints of different sizes to analyze random CNF formulas with clauses of sizes 2 and 3.

density

$$f_k(\beta|r) = \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\log\left(\sum_{\sigma\in D^n}\exp\left(-\beta H_k(\sigma|n,r)\right)\right)\right] \quad . \tag{2.1}$$

The goal of the entropic interpolation method is to give such upper bounds. For a given σ, the energy function that counts the number of unsatisfied constraints $H_k(\sigma|n,r)$ is written as the sum of m functions $u_a(x_a)$, one for each constraint, where x_a denotes the vector of variables appearing in constraint a. Thus, $u_a(x_a) = 1$ if the associated constraint is not satisfied and 0 otherwise. For random k-SAT, each of the m independent constraints is,

$$u_a(x_{a,1},...,x_{a,k}) = \prod_{j=1}^{k}\frac{1+J_{a,j}x_{a,j}}{2} \quad , \tag{2.2}$$

where each of the k variables is picked independently and uniformly from the set of n variables and each $J_{a,j} \in \{+1,-1\}$ is an independent binary random variable with $\mathbf{P}(J_{a,j} = +1) = \mathbf{P}(J_{a,j} = -1) = 1/2$ that represents the sign of literal $x_{a,j}$ in the random clause ($+1$ if the literal is negated and -1 otherwise). Note that in this model a random constraint might be improper, meaning that its corresponding random clause might contain repeated and contradictory literals. At the same time, the probability that the ith selected variable is a repeat is equal to $(i-1)/n$. Thus, by the union bound, the probability of an improper constraint is less than k^2/n. Therefore, the number of improper constraints is $o(n)$ *w.h.p.* Also, proper constraints are selected uniformly among the set of all proper constraints. As a result, if $\mathcal{I}_k(n,rn)$ is satisfiable *w.h.p.* for a given r, then for $m = rn - o(n)$ the same is true in the model where we select only among proper constraints. Similarly, for k-uniform hypergraph q-coloring ($k = 2$ being

graph coloring) each random constraint has the associated function

$$u_a(x_{a,1}, ..., x_{a,k}) = \mathbf{1}_{x_{a,1}=x_{a,2}=...=x_{a,k}} \quad , \tag{2.3}$$

where each of the k variables are picked independently from the set of n variables.

The basic object of the interpolation method is a modified energy function that interpolates between $H_k(\sigma|n, r)$ and the energy function of a dramatically simpler (and fully tractable) model. Specifically, for $t \in [0, 1]$, let

$$\begin{aligned} \beta H_k(\sigma|n, r, t) &\equiv \beta H_k(x_1, \ldots, x_n|n, r, t) \\ &= \sum_{m=1}^{m_t} \beta \cdot u_{a_m}(x_{a_m}) + \sum_{i=1}^{n} \sum_{j=1}^{k_{i,t}} \log\left(\hat{v}_{i,j}(x_i)\right) \quad , \end{aligned} \tag{2.4}$$

where m_t is a Poisson random variable with mean $\mathbb{E}[m_t] = trn$, the $k_{i,t}$'s are i.i.d. Poisson random variables with mean $\mathbb{E}[k_{i,t}] = (1-t)kr$, and the functions $\hat{v}_{i,j}(\cdot)$ are i.i.d. random functions distributed as the function of equation (2.6) below.

Before delving into the meaning of the random functions $\hat{v}_{i,j}(\cdot)$, which are the heart of the method, let us first make a few observations about (2.4). First, note that for $t = 1$, equation (2.4) is simply the energy function of the original model. On the other hand, for $t < 1$, we expect $(1-t)m$ of the k-constraints be replaced by k times as many univariate functions, each taking as input a single variable x_i of the assignment σ. A helpful way to think about this replacement is as a decombinatorialization of the energy function wherein k-ary functions are replaced by univariate, and therefore, independent functions. As one can imagine, for $t = 0$ the model is fully tractable. In

particular, letting

$$f_k(t, \beta|r) = n^{-1}\mathbb{E}\left[\log\left(\sum_{\sigma\in D^n} \exp\left(-\beta H_k(\sigma|n, r, t)\right)\right)\right] \ , \tag{2.5}$$

one can readily compute $f_k(0, \beta|r)$ since one can compute $H_k(\sigma|n, r, 0)$ by examining one variable at a time. To relate the two models, the plan is to give a lower bound for the change in $f_k(t, \beta|r)$ as t goes from 1 to 0, hence the name interpolation, thus bounding $f_k(1, \beta|r)$ by $f_k(0, \beta|r)$ plus a term depending on our bound on the derivative.

The main idea of the interpolation method is to select the (still mysterious) univariate functions $\hat{v}_{i,j}(\cdot)$ independently from a probability distribution that reflects aspects of the geometry of the underlying solution space. The more accurate the reflection, the better the bound. One, of course, needs to guess this geometry and here is where the insights from statistical physics are valuable. A practical aspect of the interpolation method is that it projects all information about the geometry of the solution space into a single object, a distribution γ as defined below. With that in mind, we now define the random univariate functions, *but without specifying the all-important distribution* γ. This is because the method gives a valid bound for *any* γ, i.e., the choice of γ affects the quality but not the validity of the derived bound.

Let $v(x)$ denote the density function of a random variable over D, where the probabilities $p_1, \ldots, p_{|D|}$ are themselves chosen at random from a distribution γ with support on the unit $(|D| - 1)$-dimensional simplex. Let $\hat{v}(x)$ be a random univariate function defined as follows

$$\hat{v}(x) = \sum_{y_1,\ldots,y_{k-1}} \exp\left(-\beta u(y_1, ..., y_{k-1}, x)\right) \prod_{j=1}^{k-1} v_j(y_j) \ , \tag{2.6}$$

where $u(\cdot)$ is a constraint-function picked uniformly at random from $C_{k,n}$ and the functions $v_i(\cdot)$ are i.i.d. with the same distribution as $v(x)$.

To interpret the function in (2.6) it helps to think of its argument x as corresponding to a particular occurrence of a variable in a constraint c, e.g., a literal occurrence in a random k-clause. The idea is for (2.6) to simulate the biases that this particular occurrence of x "feels" from its presence in c. To do this we replace c with a brand new random constraint (appearing as $u(\cdot)$ in (2.6)) containing $k-1$ new variables $y_1, \ldots, y_{k-1}$ which are "private" to $u(\cdot)$, i.e., they will not occur in any other constraint in the interpolating energy function. To simulate the statistical joint behavior of the $k-1$ original variables in c due to their participation in constraints other than c, we assume that since the underlying random hypergraph is sparse, these $k-1$ new variables are independent in the absence of $u(\cdot)$, hence the product in (2.6). Finally, specifying the probability distribution γ governing the behavior of each ersatz variable is precisely what reflects our beliefs about the geometry of the space of solutions.

Now, we will show that as t goes from 1 to 0, we can control in the change of $f_k(t, \beta|r)$. For concreteness, we will demonstrate the final (and non-generic) part of the proof for random k-SAT and random k-uniform hypergraph q-coloring, i.e., for functions (2.2) and (2.3), but it will be clear that a similar proof can be used in many other CSPs.

Theorem 10. *For the functions (2.2) and (2.3) and* every *density function γ over the*

unit $(|D|-1)$*-dimensional simplex,*

$$f_k(\beta|r) \leq f_k(0,\beta|r) - r(k-1)\mathbb{E}\left[\log\left(v_c\right)\right] \quad , \tag{2.7}$$

where

$$v_c = \sum_{y_1,\ldots,y_k} \exp(-\beta u(y_1,\ldots,y_k)) \prod_{j=1}^{k} v_j(y_j) \quad .$$

Proof.

Part I. Since $f_k(1,\beta|r) = f_k(0,\beta|r) + \int_0^1 \frac{\partial}{\partial t}\left(f_k(t,\beta|r)\right) dt$, it suffices to prove that for every $t \in [0,1]$,

$$\frac{\partial}{\partial t}\left(f_k(t,\beta|r)\right) + r(k-1)\mathbb{E}\left[\log\left(v_c\right)\right] \leq 0 \quad . \tag{2.8}$$

Let

$$Z = Z(\beta) = \sum_{\sigma \in D^n} \exp\left(-\beta H_k(\sigma|n,r,t)\right)$$

be the so-called partition function. Recall that Z is a random variable depending on the random choices of m_t, $k_{i,t}$ for $i = 1,2,\ldots,n$ as well as the random choices needed to build the random energy-constraints and the random univariate factors. We define

$$Z_m = Z|_{m_t=m}$$

$$Z_{k_i} = Z|_{k_{i,t}=k_i}$$

$$Z_{m,k_1,\ldots,k_n} = Z|_{m_t=m,k_{1,t}=k_1,\ldots,k_{n,t}=k_n} \quad ,$$

as the value of Z conditioned on the specific values of m_t and $k_{i,t}$ for $i = 1,2,\ldots,n$. Denote the Poisson probability density function with mean μ as $\mathrm{Poi}(\mu,z) = e^{-\mu}(\mu^z/z!)$.

Since the random variable m_t and the random variables $k_{i,t}$ are independent, we can write the expectation in (2.5) as

$$f_k(t,\beta|r) = n^{-1} \sum_{m,k_1,\ldots,k_n} \left(\text{Poi}(trn,m) \prod_{i=1}^{n} \text{Poi}((1-t)rk,k_i) \right) \mathbb{E}\left[\log Z_{m,k_1,\ldots,k_n}\right] \quad . \quad (2.9)$$

Taking the derivative of the joint density in (2.9) with respect to t yields $n+1$ terms, as follows:

$$\begin{aligned} \frac{\partial}{\partial t}\left(\text{Poi}(trn,m) \prod_{i=1}^{n} \text{Poi}((1-t)rk,k_i) \right) &= \\ \frac{\partial}{\partial t}\left(\text{Poi}(trn,m)\right) \prod_{i=1}^{n} \text{Poi}((1-t)rk,k_i) &+ \quad (2.10) \\ \text{Poi}(trn,m) \frac{\partial}{\partial t}\left(\text{Poi}((1-t)rk,k_1)\right) \prod_{i=2}^{n} \text{Poi}((1-t)rk,k_i) &+ \quad (2.11) \\ \ldots & \\ \left(\text{Poi}(trn,m) \prod_{i=1}^{n-1} \text{Poi}((1-t)rk,k_i) \right) \frac{\partial}{\partial t}\left(\text{Poi}((1-t)rk,k_n)\right) & \quad . \end{aligned}$$

Summing the first of the $n+1$ terms of the derivative (see rhs of (2.10)) over all values of $m, k_1, \ldots, k_n$ yields

$$\begin{aligned} \sum_{m,k_1,\ldots,k_n} \frac{\partial}{\partial t} \text{Poi}(trn,m) \left(\prod_{i=1}^{n} \text{Poi}((1-t)rk,k_i) \mathbb{E}\left[\log Z_{m,k_1,\ldots,k_n}\right] \right) = \\ \sum_{m=0}^{\infty} \frac{\partial}{\partial t}\left(\text{Poi}(trn,m)\right) \mathbb{E}\left[\log Z_m\right] \quad . \quad (2.12) \end{aligned}$$

Summing the second term, (2.11), over all values of $m, k_1, \ldots, k_n$ yields

$$\begin{aligned} \sum_{m,k_1,\ldots,k_n} \frac{\partial}{\partial t}\left(\text{Poi}((1-t)rk,k_1)\right) \left(\text{Poi}(trn,m) \prod_{i=2}^{n} \text{Poi}((1-t)rk,k_i) \mathbb{E}\left[\log Z_{m,k_1,\ldots,k_n}\right] \right) = \\ \sum_{k_1=0}^{\infty} \frac{\partial}{\partial t}\left(\text{Poi}((1-t)rk,k_1)\right) \mathbb{E}\left[\log Z_{k_1}\right] \quad . \end{aligned}$$

$$(2.13)$$

Proceeding similarly with the remaining $n-1$ terms corresponding to $k_2, \ldots, k_n$ we conclude that

$$\frac{\partial}{\partial t}\left(f_k(t,\beta|r)\right) = n^{-1}\sum_{m=0}^{\infty}\frac{\partial}{\partial t}\mathrm{Poi}(trn,m)\mathbb{E}[\log Z_m] + n^{-1}\sum_{i=1}^{n}\sum_{k_i=0}^{\infty}\frac{\partial}{\partial t}\mathrm{Poi}((1-t)rk,k_i)\mathbb{E}[\log Z_{k_i}]$$

Recall now that $(\partial/\partial t)\mathrm{Poi}(trn,m) = -rn\mathrm{Poi}(trn,m) + rn\mathrm{Poi}(trn,m-1)$. Here is where the "memoryless" property of the Poisson distribution comes to play: the derivative with respect to t in (2.12) can be written as

$$\begin{aligned} & r\sum_{m=0}^{\infty}\left(\mathrm{Poi}(trn,m-1)-\mathrm{Poi}(trn,m)\right)\mathbb{E}[\log Z_m] \\ = \; & r\sum_{m=0}^{\infty}\mathrm{Poi}(trn,m)\left(\mathbb{E}\left[\log Z_{m+1}\right]-\mathbb{E}\left[\log Z_m\right]\right) \quad . \end{aligned} \tag{2.14}$$

Similarly, the derivatives in the double sum in (2.13) with respect to t can be written as

$$\begin{aligned} & -rkn^{-1}\sum_{i=1}^{n}\sum_{k_i=0}^{\infty}\left(\mathrm{Poi}((1-t)rk,k_i-1)-\mathrm{Poi}((1-t)rk,k_i)\right)\mathbb{E}\left[\log Z_{k_i}\right] \\ = \; & -rkn^{-1}\sum_{i=1}^{n}\sum_{k_i=0}^{\infty}\mathrm{Poi}((1-t)rk,k_i)\left(\mathbb{E}\left[\log Z_{k_i+1}\right]-\mathbb{E}\left[\log Z_{k_i}\right]\right) \quad . \end{aligned} \tag{2.15}$$

Now, a crucial observation is that (2.14) is r times the expected value of the change in $\log Z$ after adding a random constraint, while (2.15) is $-rk$ times the expected value of the change in $\log Z$ after adding a single $\log \hat{v}$ function, whose argument is a variable selected uniformly at random. Thus, to establish (2.8) we need to show that the expected change in $\log Z$ caused by adding a random constraint minus k times

the expected change caused by adding a random $\log \hat{v}$ function is at most $-r(k-1)\mathbb{E}\left[\log\left(v_c\right)\right]$. Observe that the definition of the $\log \hat{v}$ functions is irrelevant so far.

Part II. To make the comparison and prove the assertion above we need to:

1. Select: (i) a random formula H from the distribution $H_k(\cdot|n,r,t)$, (ii) a random constraint c, (iii) a random variable $x \in \{x_1,\ldots,x_n\}$, and (iv) a random $\hat{v}$-function.

2. Let $H'(\sigma) = H(\sigma) + u_c$, $H'' = H(\sigma) + \log \hat{v}(x)$ and let Z' and Z'' denote the respective partition functions.

3. Let $Y = (\log Z' - \log Z) - k(\log Z'' - \log Z)$.

4. Prove that $\mathbb{E}Y$, over the choice of $H, u, v, \hat{v}$, is at most $-(k-1)\mathbb{E}\left[\log\left(v_c\right)\right]$.

The averaging task in the last step above appears quite daunting, as we need to average over H. The reason we can establish the desired conclusion is that something far stronger holds. Namely, we will prove that for *every* realization of H, the conditional expectation of Y, i.e., the expectation over only u, v and $\hat{v}$, satisfies the desired inequality.

Specifically, let H_0 denote any realization of $H_k(\cdot|n,r,t)$ and let

$$Z_0 = \sum_{\sigma \in D^n} \exp\left(-\beta H_0(\sigma)\right)$$

be its partition function. We are going to compute the expected value in the change of $\log Z_0$ after adding a new random function $u_a(x_a)$ and after adding a new random univariate function $\log\left(\hat{v}_i(\cdot)\right)$ to a uniformly selected variable and show that the following

inequality holds for any realization H_0:

$$\left(\mathbb{E}\left[\log Z'\right] - \log Z_0\right) - k\left(\mathbb{E}\left[\log Z''\right] - \log Z_0\right) - (k-1)\mathbb{E}\left[\log\left(v_c\right)\right] \leq 0 \ . \tag{2.16}$$

It will be convenient to write $\exp(\beta u_a)$ as follows

$$\exp\left(\beta u_a(x_a)\right) = 1 - (1 - e^{-\beta})u_a(x_a) \ . \tag{2.17}$$

Note that, the above expression evaluates to $e^{-\beta}$ when the constraint is unsatisfied and to 1 when it is not, as it should.

The partition function resulting after adding a random constraint to H_0 is equal to

$$\begin{aligned} Z' &= \sum_{\sigma \in D^n} \Big(1 - (1 - e^{-\beta})u_a(x_a)\Big) \exp\left(-\beta H_0(\sigma)\right) \\ &= Z_0 - (1 - e^{-\beta}) \sum_{\sigma \in D^n} u_a(x_a) \exp\left(-\beta H_0(\sigma)\right) \ . \end{aligned}$$

To lighten notation, below, we won't make explicit the variables that appear in $u_a(\cdot)$ and just write $u_a(\sigma)$ instead. The expected value of the change in the partition function is thus

$$\mathbb{E}\left[\log Z'\right] - \log Z_0 = \mathbb{E}\left[\log\left(1 - (1 - e^{-\beta})\frac{\sum_{\sigma \in D^n} u_a(\sigma) \exp\left(-\beta H_0(\sigma)\right)}{Z_0}\right)\right] \ . \tag{2.18}$$

Using the Taylor series expansion of $\log(1-x)$ for $0 \leq x < 1$, i.e., $\log(1-x) =$

$-\sum_{i=1}^{\infty} x^i/i$ we rewrite (2.18) as

$$\mathbb{E}\left[-\sum_{i=1}^{\infty}\frac{(1-e^{-\beta})^i}{i}\left(\frac{\sum_{\sigma\in D^n}u_a(\sigma)\exp\left(-\beta H_0(\sigma)\right)}{Z_0}\right)^i\right] =$$

$$\mathbb{E}\left[-\sum_{i=1}^{\infty}\frac{(1-e^{-\beta})^i}{i}\cdot\frac{\sum_{\sigma_1}\cdots\sum_{\sigma_i}\left[\prod_{j=1}^{i}u_a(\sigma_j)\exp\left(-\beta\sum_{j=1}^{i}H_0(\sigma_j)\right)\right]}{Z_0^i}\right] =$$

$$-\sum_{i=1}^{\infty}\frac{(1-e^{-\beta})^i}{i}\cdot\frac{\sum_{\sigma_1}\cdots\sum_{\sigma_i}\left[\mathbb{E}\left[\prod_{j=1}^{i}u_a(\sigma_j)\right]\exp\left(-\beta\sum_{j=1}^{i}H_0(\sigma_j)\right)\right]}{Z_0^i} \quad . \qquad (2.19)$$

The expression for the change in the partition function after the addition of a new random univariate factor is computed similarly. In this case, the partition function that results from adding the univariate factor is

$$Z'' = \sum_{\sigma\in D^n}\exp(\log\hat{v}_i(x_i))\exp\left(-\beta H_0(\sigma)\right) = \sum_{\sigma\in D^n}\hat{v}_i(x_i)\exp\left(-\beta H_0(\sigma)\right) \quad .$$

This is the first point where some aspect of the choice of the $\hat{v}_i(\cdot)$ functions becomes relevant. Recall that

$$\hat{v}_i(x_i) \quad = \quad \sum_{y_1,\ldots,y_{k-1}}\exp\left(-\beta u(y_1,\ldots,y_{k-1},x_i)\right)\prod_{j=1}^{k-1}v_j(y_j) \quad ,$$

i.e., each univariate function contains as a factor a random u-constraint, albeit one chosen independently of all other constraints and in which $k-1$ of the variables, $y_1,\ldots,y_{k-1}$ are "private" to the constraint. Using the representation of constraints given by equation (2.17), we finally rewrite the univariate factor as

$$\begin{aligned}\hat{v}_i(x_i) &= \sum_{y_1,\ldots,y_{k-1}}\Big(1-(1-e^{-\beta})u(y_1,\ldots,y_{k-1},x_i)\Big)\prod_{j=1}^{k-1}v_j(y_j)\\ &= 1-(1-e^{-\beta})\sum_{y_1,\ldots,y_{k-1}}u(y_1,\ldots,y_{k-1},x_i)\cdot\prod_{j=1}^{k-1}v_j(y_j) \quad . \qquad (2.20)\end{aligned}$$

The computation of $\mathbb{E}\left[\log Z''\right] - \log Z_0$ is done following exactly the same procedure as for computing $\mathbb{E}\left[\log Z'\right] - \log Z_0$ and employing the expression in (2.20). It yields that $\mathbb{E}\left[\log Z''\right] - \log Z_0$ is

$$-\sum_{i=1}^{\infty} \frac{(1-e^{-\beta})^i}{i} \cdot \left(Z_0^i\right)^{-1} \sum_{\sigma_1} \cdots \sum_{\sigma_i}$$
$$\mathbb{E}\left[\prod_{j=1}^{i}\left[\sum_{y_1,\ldots,y_{k-1}} u(y_1,\ldots,y_{k-1},\sigma_i)\cdot\prod_{j=1}^{k-1} v_j(y_j)\right]\right]\exp\left(-\beta\sum_{j=1}^{i} H_0(\sigma_j)\right) \qquad (2.21)$$

Finally, in an entirely analogous manner, we get

$$\begin{aligned} -(k-1)\mathbb{E}\left[\log\left(v_c\right)\right] &= \\ -(k-1)\mathbb{E}\left[\log\left(1-(1-e^{-\beta})\sum_{y_1,\ldots,y_k} u(y_1,\ldots,y_k)\cdot\prod_{j=1}^{k} v_j(y_j)\right)\right] &= \\ -\sum_{i=1}^{\infty}\frac{(1-e^{-\beta})^i}{i}(k-1)\mathbb{E}\left[\left(\sum_{y_1,\ldots,y_k} u(y_1,\ldots,y_k)\cdot\prod_{j=1}^{k} v_j(y_j)\right)^i\right] & . \qquad (2.22) \end{aligned}$$

Since $\sum_{\sigma_1}\cdots\sum_{\sigma_i}\left[\exp\left(-\beta\sum_{j=1}^{i} H_0(\sigma_j)\right)\right] = Z_0^i$, we can rewrite (2.22) as

$$-r\sum_{i=1}^{\infty}\frac{(1-e^{-\beta})^i}{i}\cdot\left(Z_0^i\right)^{-1}\sum_{\sigma_1}\cdots\sum_{\sigma_i} \qquad (2.23)$$
$$\mathbb{E}\left[(k-1)\left(\sum_{y_1,\ldots,y_k} u(y_1,\ldots,y_k)\cdot\prod_{j=1}^{k} v_j(y_j)\right)^i\right]\exp\left(-\beta\sum_{j=1}^{i} H_0(\sigma_j)\right)$$

We are now left to compare the difference between the quantities in (2.19) and (2.21), with the quantity in (2.23). Observe that the only difference between these three expressions is in the quantity inside the expectation. This was precisely the reason for introducing (2.17) as, via the Taylor expansion, it allows us to decompose the comparison along moments ("replicas" in the language of statistical physics). This is the end of the general part of the proof. Now, to succeed in our goal it suffices to

focus on these three expectations and prove that they satisfy the desired inequality for the equations (2.2) and (2.3).

2.1.1 Applying Entropic Interpolation to Random CSPs

At this point, we need to start looking at the problem-specific univariate functions $\hat{v}(x)$ and, as result, the proof splits per problem. Nevertheless, we will see that the proofs for the different problems are highly parallel and the structure of the overall computations is identical.

2.1.1.1 Random k-SAT

In equation (2.19) the expectation is equal to the probability that the constraint associated with the function $u_a(\cdot)$ is unsatisfied by all assignments $\sigma_1, ..., \sigma_i$. Thus, if l denotes the number of variables in which all the assignments $\sigma_1, ..., \sigma_i$ agree, then the probability of $u_a(\cdot)$ not satisfying all of them is equal to

$$\left(\frac{l}{2n}\right)^k \quad .$$

In equation (2.21) the expectation is

$$\mathbb{E}\left[\prod_{j=1}^{i}\left[\sum_{y_1,\ldots,y_{k-1}} u(y_1,\ldots,y_{k-1},\sigma_i)\cdot\prod_{j=1}^{k-1} v_j(y_j)\right]\right] =$$
$$\mathbb{E}\left[\prod_{j=1}^{i}\left[\left(\frac{1+J_i x_j}{2}\right)\cdot\prod_{l=1}^{k-1}\left(\sum_{y_l}\left(\frac{1+J_l y_l}{2}\right)v_l(y_l)\right)\right]\right] \quad .$$

Note that each of the terms inside the innermost product of the r.h.s. are i.i.d. random

variables in $[0,1]$. We will use the letter A to refer to them, that is,

$$A_l = \sum_{y_l} \left(\frac{1 + J_l y_l}{2} \right) v_l(y_l) \ .$$

We will also define $g(x_i) = (1+J_i x_i)/2$ and, again, we will suppress the explicit argument writing $g(\sigma)$ instead. Thus, the r.h.s. of the above equation simplifies to

$$\begin{aligned}
\mathbb{E}\left[\prod_{j=1}^{i}\left[g(\sigma_i)\prod_{l=1}^{k-1}A_l\right]\right] &= \mathbb{E}\left[\prod_{l=1}^{k-1}A_l^i\prod_{j=1}^{i}[g(\sigma_i)]\right] \\
&= \prod_{l=1}^{k-1}\mathbb{E}\left[A_l^i\right]\mathbb{E}\left[\prod_{j=1}^{i}g(\sigma_i)\right] \qquad (2.24)\\
&= \left(\mathbb{E}\left[A_l^i\right]\right)^{k-1}\mathbb{E}\left[\prod_{j=1}^{i}g(\sigma_i)\right] \qquad (2.25)\\
&= \left(\mathbb{E}\left[A_l^i\right]\right)^{k-1}\left(\frac{l}{2n}\right) \ .
\end{aligned}$$

For (2.24) we used the independence of the random variables involved in the expectation, while for (2.25) we used the fact that the A's are i.i.d. random variables. The rightmost expectation in (2.25) is just the probability that all assignments $\sigma_1, ..., \sigma_i$ do not satisfy the function $g(\sigma_i)$, i.e., $(l/2n)$.

Finally, the expectation in equation (2.23) is

$$\begin{aligned}
\mathbb{E}\left[-(k-1)\left(\sum_{y_1,\ldots,y_k}u(y_1,\ldots,y_k)\cdot\prod_{j=1}^{k}v_j(y_j)\right)^i\right] = \\
-(k-1)\mathbb{E}\left[\left(\prod_{j=1}^{k}A_j\right)^i\right] = \\
-(k-1))\left(\mathbb{E}\left[A_l^i\right]\right)^k
\end{aligned}$$

Combing (2.19), (2.21), and (2.23) we get that

$$\left(\mathbb{E}\left[\log Z'\right] - \log Z_0\right) - k\left(\mathbb{E}\left[\log Z''\right] - \log Z_0\right) - (k-1)\mathbb{E}\left[\log\left(v_c\right)\right]$$

is

$$-\left(\left(\frac{l}{2n}\right)^k - k\left(\frac{l}{2n}\right)\left(\mathbb{E}\left[A_l^i\right]\right)^{k-1} + (k-1)\left(\mathbb{E}\left[A_l^i\right]\right)^k\right) \ ,$$

which is always less than or equal to 0 since the polynomial $F(x,p) = x^k - kp^{k-1}x + (k-1)p^k \geq 0$ for all $0 \leq x, p \leq 1$. To see this last statement note that (i) $F(0,p)$, $F(1,p)$, $F(x,0)$, $F(x,1) \geq 0$ and, (ii) the derivative of F with respect to p is 0 only when $p = x$, in which case $F(x,x) = 0$.

2.1.1.2 Random Graph Coloring

To compute the expectation in equation (2.19) define the following equivalence relation on $[n]$: given $j_1, j_2 \in [n]$, j_1 is related to j_2 if they have the same color in all the colorings $\sigma_1, ..., \sigma_i$, that is, $j_1 \sim j_2$ if $\sigma_l(j_1) = \sigma_l(j_2)$, for all $l = 1, \ldots, i$. Let O_s^*, $1 \leq s \leq J$, for some $1 \leq J \leq n$, be the corresponding equivalence classes. A random k-uniform hyper-edge is unsatisfied by all the colorings $\sigma_1, ..., \sigma_i$ if the selected vertices are in the same equivalence class i.e., the probability is equal to

$$\sum_{1 \leq s \leq J} \left(\frac{|O_s^*|}{n}\right)^k \ .$$

In equation (2.21) the expectation is

$$\begin{aligned}
\mathbb{E}\left[\prod_{l=1}^{i}\left[\sum_{y_1,\ldots,y_{k-1}} u(y_1,\ldots,y_{k-1},\sigma_i(x)) \cdot \prod_{j=1}^{k-1} v_j(y_j)\right]\right] &= \mathbb{E}\left[\prod_{l=1}^{i}\left[\prod_{j=1}^{k-1} v_j(\sigma_l(x))\right]\right] \\
&= \prod_{j=1}^{k-1} \mathbb{E}\left[\prod_{l=1}^{i} v_j(\sigma_l(x))\right] \\
&= \left(\mathbb{E}\left[\prod_{l=1}^{i} v_1(\sigma_l(x))\right]\right)^{k-1}
\end{aligned}$$

where x is the index of a vertex picked at random. In the last equalities, we used the fact that the $v(\cdot)$ functions are i.i.d. By conditioning the selection of the random vertex on the equivalence classes defined above, we get

$$\mathbb{E}\left[\prod_{l=1}^{i}[v(\sigma_l(j))]\right] = \sum_{s=1}^{J}\frac{|O_s^*|}{n}\left(\mathbb{E}\left[\prod_{l=1}^{i}v_1(\sigma_l(x_s^*))\right]\right)^{k-1} ,$$

where x_s^* is any vertex in O_s^*. We will say that $\mathbb{E}\left[\prod_{l=1}^{i}v(\sigma_l(x_s^*))\right]$ is the profile of a vertex in O_s^*.

Finally, the expectation in equation (2.23) is

$$\begin{aligned}
\mathbb{E}\left[-\left(\sum_{y_1,\ldots,y_k}u(y_1,\ldots,y_k)v_1(y_1)\ldots v_k(y_k)\right)^i\right] &= \\
-\mathbb{E}\left[\left(\sum_{y}v_1(y)\ldots v_k(y)\right)^i\right] &= \\
-\mathbb{E}\left[\sum_{y_1}\cdots\sum_{y_i}\prod_{l=1}^{i}v_1(y_l)\cdots\prod_{l=1}^{i}v_k(y_l)\right] &= \\
-\sum_{y_1}\cdots\sum_{y_i}\mathbb{E}\left[\prod_{l=1}^{i}v_1(y_l)\right]\cdots\mathbb{E}\left[\prod_{l=1}^{i}v_2(y_l)\right] &= \\
-\sum_{y_1}\cdots\sum_{y_i}\left(\mathbb{E}\left[\prod_{l=1}^{i}v_1(y_l)\right]\right)^k .
\end{aligned}$$

In the second to last equation, we used the independence between the $v(\cdot)$ functions. Note that the summation in above equation is over the k-th power of all possible profiles, thus, we can partition the above summation as the sum of the k-th power of the J profiles given by the colorings $\sigma_1,\ldots,\sigma_i$ plus a positive constant P equal to the sum of the k-th power of the remaining profiles giving

$$-\sum_{1\le s\le J}\left(\mathbb{E}\left[\prod_{l=1}^{i}v(\sigma_l(x_s^*))\right]\right)^k - P \ .$$

Combining (2.19), (2.21), and (2.23) we get that

$$\left(\mathbb{E}\left[\log Z'\right] - \log Z_0\right) - k\left(\mathbb{E}\left[\log Z''\right] - \log Z_0\right) - (k-1)\mathbb{E}\left[\log\left(v_c\right)\right]$$

is

$$-\left(\sum_{1\leq s\leq J}\left(\frac{|O_s^*|}{n}\right)^k - \sum_{1\leq s\leq J}\frac{|O_s^*|}{n}\left(\mathbb{E}\left[\prod_{l=1}^{i} v(\sigma_l(x_s^*))\right]\right)^{k-1} + \sum_{1\leq s\leq J}\left(\mathbb{E}\left[\prod_{l=1}^{i} v(\sigma_l(x_s^*))\right]\right)^k + P\right) =$$

$$\sum_{1\leq s\leq J} -\left(\left(\frac{|O_s^*|}{n}\right)^k - \frac{|O_s^*|}{n}\left(\mathbb{E}\left[\prod_{l=1}^{i} v(\sigma_l(x_s^*))\right]\right)^{k-1} + \left(\mathbb{E}\left[\prod_{l=1}^{i} v(\sigma_l(x_s^*))\right]\right)^k\right) - P$$

The quantity inside parenthesis is a polynomial like the one for random k-SAT p, i.e., always non-negative. Therefore the whole expression is non-positive since $-P$ is a non-positive constant. □

2.2 Difficulties in Using the EIM to Obtain Rigorous Upper Bounds

At this point, we have a rigorous bound that is a functional of a distribution over a d-dimensional simplex ($d = 2$ for random k-SAT and $d = q$ for random q-Coloring). A direct optimization of the functional of Theorem (10) seems very daunting. Indeed, the form of the functional does not fall into any of the well known patterns for which a methodology for solving the problem is already available [50, 18].

Rather than trying to find the right distribution via analytical methods, one approach is to obtain a distribution with finite support and then plug it into equation 2.7. If for a given value of r the free entropy density is negative, then such r would be a rigorous upper bound for the satisfiability threshold of the random CSP under study because Theorem 10 is valid for any distribution γ.

The EIM bounds coincide with the expressions obtained with the RS cavity method approximation with the exception that the distribution γ is not a parameter, but the solution of a system of stochastic equations. It is very hard to solve these stochastic equations analytically too, so one has to use a numerical method called Population Dynamics [1] (In Appendix A we describe the RS cavity method and the Population Dynamics algorithm).

The distributions γ obtained through Population Dynamics are therefore natural choices to plug in into the EIM bound. Unfortunately, Population Dynamics breaks, a denominator goes to zero, before the free entropy density goes below zero. Division by zero occurs when elements in the population start taking values of either 0 or 1, meaning that the empirical distribution of γ puts weight greater than zero to the probability that a variable is *frozen* to a specific value. Figures 2.1 and 2.2 show the value of the RS approximation as a function of r for random k-SAT with $k = 3$ and $k = 2$, respectively. All the points were computed using Monte Carlo Methods, that is, sample equation (2.7) multiple times and report the sample average. To get rigorous bounds, we must compute the expectations explicitly (which has its own problems). In both plots, bigger densities make Population Dynamics fail. Observe that the RS

estimate of f for random 3-SAT goes very close to zero (0.003), however for random 2-SAT Population Dynamics fails when the estimate of f is 0.38. Physicists use the density at which Population Dynamics fails to converge as the transition point predicted by the method i.e., according to the plots, the RS predictions of the threshold for random 2- and 3-SAT is approximately 1.0 and 4.67 respectively. The idea is that Population Dynamics fails because the number of solutions is zero i.e. the uniform distribution on the set of satisfying assignments ($\beta \to \infty$) is not well defined anymore.

Since Population Dynamics fails with the appearance of frozen variables, one could try to write the distribution γ parametrically so that the fraction of variables b is explicitly defined, and then take the limit of the expression in (2.7) when b goes to zero. In [12], the authors used this idea to prove bounds for random k-SAT using the replica method, another method from statistical mechanics for which no formal results are known. However, the expressions used in the replica method are simpler than (2.7) and the same technique could not be applied. In Appendix C, we give a brief introduction to the replica method and the difficulties we encountered when trying to use the above idea with the bound of Equation (2.7).

One possibility at this point is to find the maximum density for which Population Dynamics works and store the corresponding density γ_{MAX}. Then, keep γ_{MAX} fixed and compute Equation 2.7 for increasing values of r until the density become negative. This alternative works well with random 3-SAT giving a bound of[2] 4.69. However, for random 2-SAT the bound obtained is ~ 1.91 i.e. way above the known value of 1.0.

[2]This bound can be improved to 4.67 by a more careful numerical computation.

Another alternative to make f go below zero is to add a small fraction δn of constraints. The hope is that if f is small enough, then the price in density δ we have to pay is very small compared to the original density. Lemma (11) gives the exact amount of δ needed for a given value of f.

Lemma 11. *Let Z count the number of satisfying assignments in an random k-CNF formula $\mathcal{F}_k(n, rn)$, let $(1/n)\mathbb{E}\log Z \leq f$, and let Z_{δ_k} count the number of satisfying assignments after adding $\delta_k \cdot n$ new clauses of size k to $\mathcal{F}_k(n, rn)$. Then,*

$$\frac{1}{n}\mathbb{E}[\log Z_{\delta_k}] \leq f + \delta_k \log(1 - \frac{1}{2^k}) \ .$$

Proof.

$$\begin{aligned}
\mathbb{E}[\log Z_{\delta_k}] &= \mathbb{E}[\mathbb{E}[\log Z_{\delta_k}|Z]] \\
&\leq \mathbb{E}[\log \mathbb{E}[Z_{\delta_k}|Z]] \\
&= \mathbb{E}[\log((1 - \frac{1}{2^k})^{\delta_k n} Z)] \\
&= \delta_k n \log(1 - \frac{1}{2^k}) + \mathbb{E}[\log Z] \\
&\leq \delta_k n \log(1 - \frac{1}{2^k}) + nf
\end{aligned}$$

The first equality follows from the rule of iterated expectation. The inequality of the second line follows from Jensen's inequality, since $\log(\cdot)$ is concave. Each of the satisfying assignments implied by Z will satisfy the new formula with probability $(1-\frac{1}{2^k})^{\delta_k n}$ giving the third line. Finally, we applied the hypothesis of the Lemma to prove the result. □

Unfortunately, this approach does not give good results for random 2-SAT either. So we turn to a different approach in the next chapter.

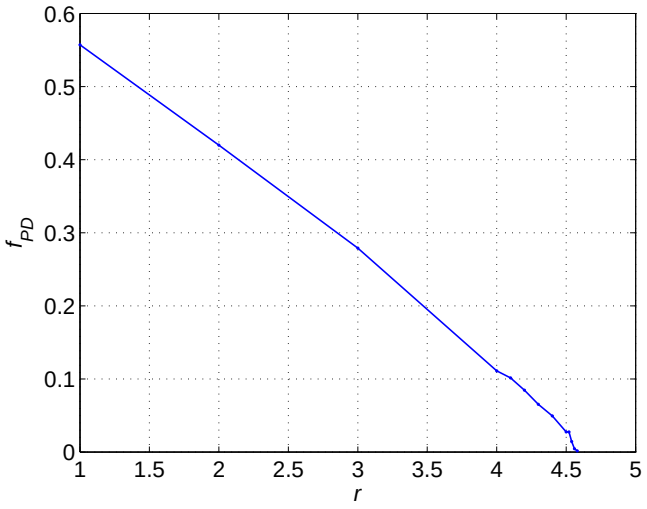

Figure 2.1: Free entropy density estimate obtained by Population dynamics (f_{PD}) for random 3-SAT plotted as a function of the density r.

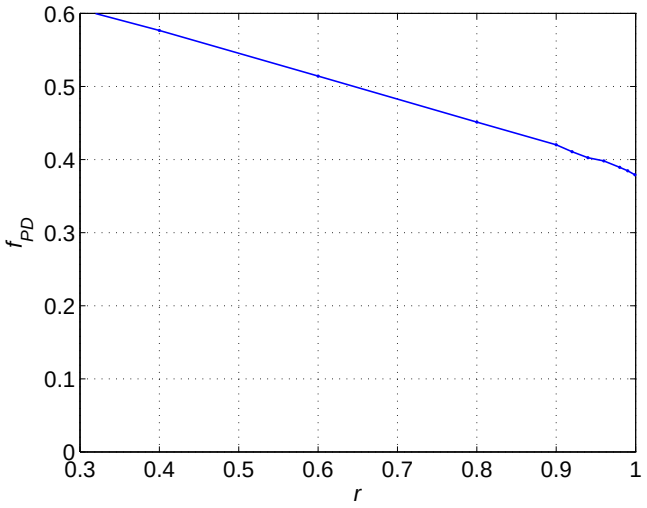

Figure 2.2: Free entropy density estimate obtained by Population dynamics (f_{PD}) for random 2-SAT plotted as a function of the density r. Recall that the satisfiability threshold for random 2-SAT is 1.

Chapter 3

Energetic Interpolation Bounds

The two main reasons that motivated our derivation of a different, so-called, energetic interpolation method are 1) to avoid the analytical difficulties of the entropic interpolation method; we will see that the energetic approach leads to bounds which can be derived essentially analytically, and 2) to capture in a simple model the continuous transition of the fraction of frozen variables in problems like random 2SAT, random MAX-2-SAT, random MAX-2-LIN-2, and $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$. More precisely, we model the phenomenon in which a frozen variable that appears in a constraint "the wrong way" causes the other variable in the constraint to also freeze. This percolative type of behavior causes the fraction of frozen variables to takeoff smoothly in such problems, a situation that can be captured by a simple model that focuses explicitly on the propagation of frozen variables.

3.1 Energetic Interpolation for General CSPs

To develop an energetic interpolation method we replace the (far richer) free entropy density of the previous section with the following simpler quantity

$$\xi_{k,r} \triangleq \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\min_{\sigma\in D^n} H_k(\sigma|n,r)\right] \ , \tag{3.1}$$

known as ground-state energy density, which simply tells us what fraction of constraints is violated by the optimal (least-violating) assignment.

The random variable $\min_{\sigma\in D^n} H_k(\sigma|n,r)$ concentrates around its expectation in a $o(n)$ window. This result can be proven using the independent bound differences theorem (which we repeat below as Theorem 12) in a similar way as it was used to prove the concentration result for the free entropy.

Theorem 12 (Independent bound differences Theorem [39][1])**.** *Let $X = (X_1, X_2, ..., X_n)$ be a family of independent random variables with X_i taking values in a set A_i for each i. Suppose that the real-valued function g defined on the product space $\prod A_i$ satisfies the Lipschitz condition*

$$\left|g(x) - g(x')\right| \leq c_i \ ,$$

whenever the vectors x and x' differ only on the i-th coordinate. Then for any $t \geq 0$,

$$\mathbf{P}(g(X) - \mathbb{E}[g(X)] \geq t) \leq e^{-2t^2/\sum c_i} \tag{3.2}$$

and

$$\mathbf{P}(g(X) - \mathbb{E}[g(X)] \leq -t) \leq e^{-2t^2/\sum c_i} \ . \tag{3.3}$$

[1]Page 10, Theorem 3.1

If the set of constraints correspond to the set of random variables X in Theorem 12, then $\min_{\sigma \in D^n} H_k(\sigma|n,r)$ satisfies the Lipschitz condition with $c_i = 1$, since changing any constraint cannot change the minimum by more than 1. Thus, by setting $t = \mu = \mathbb{E}\left[\min_{\sigma \in D^n} H_k(\sigma|n,r)\right]$ in equation 3.3 we get

$$\mathbf{P}(\min_{\sigma \in D^n} H_k(\sigma|n,r) \leq 0) \leq e^{-2\mu^2/(rn)} \ .$$

Consequently, if μ increases at least linearly with n, then $\mathbf{P}(\min_{\sigma \in D^n} H_k(\sigma|n,r) \leq 0)$ tends to 0, i.e., $\min_{\sigma \in D^n} H_k(\sigma|n,r) > 0$ *w.h.p.*. But $\xi_{k,r} > 0$ implies that $\mu = \Omega(n)$ and $\min_{\sigma \in D^n} H_k(\sigma|n,r) > 0$ implies that no solutions exist, therefore r is an upper bound for the satisfiability threshold for $\mathcal{I}_k(n, rn)$.

The application of the interpolation method to $\min_{\sigma \in D^n} H_k(\sigma|n,r)$ shares many ideas from its application to the free entropy. In particular, the idea of interpolating between $H_k(\sigma|n,r)$ and a simpler model composed of univariate factors only. The univariate factors in the energy interpolation method are given as follows:

- For $1 \leq j \leq |D|$, let I_j denote the indicator function that the input is the j-th element of D, i.e., I_j is 1 if its input is d_j and 0 otherwise.
- Let I_* denote the function that assigns 0 to all elements of D.
- Let $b(x)$ be a random function in $\{I_1, \ldots, I_{|D|}, I_*\}$ with $\Pr(\mathtt{h}(\cdot) = I_*) = 1 - p$ and $\Pr(\mathtt{h}(\cdot) = I_j) = p/|D|$.

The analogue of 2.6 is now defined to be

$$\hat{h}(x) = \min_{y_1, \ldots, y_{k-1}} \left\{ u(y_1, .., y_{k-1}, x) + \sum_{i=1}^{k-1} b_i(y_i) \right\} \ , \tag{3.4}$$

where $u(\cdot)$ is the energy function of a random constraint while the functions $b_i(\cdot)$ are i.i.d. random functions distributed as $b(x)$.

Observe that the energy interpolation method models all information about the geometry of the solution space into a single probability p, which can be interpreted as the probability that a variable picked at random will be frozen, i.e., have the same value in all optimal assignments. If that occurs for all $k-1$ variables $y_1, .., y_{k-1}$ and they all happen to be frozen "the wrong way" as far as u is concerned, then unless variable x takes the value desired by u the function $\hat{h}(x)$ will evaluate to 1. When, at the end of the interpolation, we will have replaced all k-ary constraints with univariate random functions $\hat{h}$, the optimal overall assignment is simply found by assigning to each variable the value that makes the majority of its $\hat{h}$ functions evaluate to 0. The method delivers a valid bound for *any* choice of $p \in [0,1]$ and the bound is then optimized by choosing the best value of p, i.e., performing a single-parameter search.

At this point, we can give lower bounds on (3.1) for random CSPs defined on Erdos-Renyi (hyper)graphs by exploiting the same Poissonization device as in the derivation of the Entropic Interpolation method presented in the previous chapter. Specifically, the interpolation function $H_k(\sigma|n,r,t)$ in the energetic version is as follows: for $t \in [0,1]$, let

$$H_k(\sigma|n,r,t) = \sum_{m=1}^{m_t} u_{a_m}(x_{a_{m,1}}, ..., x_{a_{m,k}}) + \sum_{i=1}^{n} \sum_{j=1}^{k_{i,t}} \hat{h}_{i,j}(x_i) \ , \tag{3.5}$$

where m_t and the $k_{i,t}$'s are defined in the same way as in the entropic method. Namely, m_t is a Poisson random variable with mean $\mathbb{E}[m_t] = trn$ and the $k_{i,t}$'s are i.i.d. Poisson

random variables with mean $\mathbb{E}[k_{i,t}] = (1-t)kr$. The difference is in the functions $\hat{h}_{i,j}(\cdot)$ which in this case are i.i.d. random functions distributed as the function defined in 3.4. Let

$$\xi_{k,r}(t) = \lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\min_{\sigma\in D^n} H_k(\sigma|n,r,t)\right] \quad .$$

The energetic bounds come from the following theorem.

Theorem 13. *For any choice of* $p \in [0,1]$, $\xi_{k,r} = \xi_{k,r}(1) \geq \xi_{k,r}(0) - r(k-1)\mathbb{E}[h_c]$, *with*

$$h_c = \min_{y_1,\ldots,y_k} \left\{ u(y_1,..,y_k) + \sum_{i=1}^{k} b_i(y_i) \right\} \quad . \tag{3.6}$$

To prove Theorem 13, we would start and proceed exactly as in the proof of Theorem 10 up to (2.16), except that we would replace $\log Z$ (and its different conditionals) with $\min H$ (and its respective conditionals). That is, in the context of the energetic interpolation method, the proof reduces to comparing the expected change in $\min H_0$ after adding a random energy-function associated with a random k-constraint and the expected change in $\min H_0$ after adding a random univariate factor $\hat{h}(x)$ to an arbitrary energy function H_0. The fact that the proof holds for every H_0 suggest that it should be possible to extend the proof to random CSPs with an arbitrary degree sequences. In the next section, we present a different combinatorial proof that bypass the use of Poisson random variables which we use to extend the method to (sparse) degree distributions. This enables the treatment not only of regular graphs but, more importantly, of degree sequences that result by first applying some preprocessing of the random CSP. For example, to prove that random graphs with a certain average degree are not q-colorable (an Erdos-Renyi CSP) it is desirable to apply the interpolation

method to the q-core of the random graph, i.e., to first iteratively remove all vertices of degree strictly less than q.

3.2 The Interpolation Method on Sparse Degree Sequences

Let d_x denote the number of times variable x should appear in the random instance and let $L_x = \{x_j\}_{j=1}^{d_x}$ denote its actual set of occurrences. The occurrences can be decorated so that, for example in k-SAT, we can specify how many of the L_x occurrences correspond to positive literals and how many to negative literals. In the models discussed here, decorations will be assigned uniformly at random, for example, in the case of k-SAT, each literal has sign selected uniformly at random from $\{+1, -1\}$. It will be helpful to think of each occurrence as a piece of paper carrying the index of the underlying variable along with any desired decoration. We will refer to $\mathcal{L} = \bigcup L_x$ as the degree sequence of a random CSP. We will focus on random CSPs with sparse degree distributions, that is, the total number of occurrences $|\mathcal{L}|$ is of order $O(n)$. Let $\mathcal{I}_k(n, r, \mathcal{L})$ denote the family of CSPs over n variables, rn constraints and degree sequence $\mathcal{L}$. To form a random instance $\mathcal{I}_k(n, r, \mathcal{L})$, we simply choose a random permutation of all krn elements of $\mathcal{L}$ and consider the first k to specify the first constraint, the next k to specify the second constraint, etc. Let $H_k(\sigma|, n, r, \mathcal{L})$ denote the corresponding energy function i.e. the number of clauses not satisfied and let

$$\xi_{k,r,\mathcal{L}} \triangleq n^{-1}\mathbb{E}\left[\min_{\sigma \in D^n} H_k(\sigma|, n, r, \mathcal{L})\right]$$

denote its ground state energy density.

Instead of defining the interpolation function in terms of Poisson random variables, we use an algorithm that builds an energy function where the degrees of the variables is given by $\mathcal{L}$ and the number of random constraints and random univariate factors like the ones in (3.4) is specified by a *interpolation* sequence $w \in \{\texttt{con}, \texttt{uni}\}^t$ of length t (t will be at most the cardinality of $\mathcal{L}$). The sequence w tells the algorithm what type of factor to add at each step: if the i-th element of w is a *const* then a new random constraint is added with probability $1/k$ and if the element is a *uni* then a random univariate factor is added with probability 1. Adding a random constraint with probability $1/k$ is what allows us to compare the change in the ground state energy after adding a random constraint versus k times the change after adding a single univariate factor as in the Poisson case. The complete algorithm is given below.

Let $\mathcal{H}(\mathcal{L}, w)$ denote the random energy functions H produced by the interpolation algorithm 3. Observe that when $t = |\mathcal{L}|$ and $w = \texttt{uni}, \ldots \texttt{uni}$, the energy functions produced by the algorithm has variable degree distribution given by $\mathcal{L}$ and consist of univariate factors only. On the other hand when $t = |\mathcal{L}|$ and $w = \texttt{con}, \ldots, \texttt{con}$ the resulting energy functions consist of $\tilde{m}$ energy constraint functions of arity k where $\tilde{m}$ is a Binomial random variable with krn trials and probability of success $1/k$, conditioned on having at most rn successes (constraints). Observe that the expected value of ground state energy $\mathcal{H}(\mathcal{L}, w)$ is a lower bound of the one for the original problem $H_k(\sigma|, n, r, \mathcal{L})$ because the probability that a formula is satisfiable is a monotone decreasing function, that is, when $\tilde{m} < rn$ the instances can only became easier to satisfy.

The goal now is to relate the ground state energy of these two extreme cases. A

Algorithm 3 INTERPOLATION ALGORITHM

Input: Set of literals $\mathcal{L}$, Interpolation sequence w

1: $H = \emptyset$, $i = 1$, and $j = 1$

2: Select a random permutation π of the elements of $\mathcal{L}$

3: **while** $j \leq |\mathcal{L}|$ and $i \leq |w|$ **do**

4: **if** $w[j] =$ `uni` **then**

5: Add a random univariate factor to H with argument $\pi(j)$

6: $j \leftarrow j + 1$

7: **end if**

8: **if** $w[t] =$ `con` **then**

9: With probability $1/k$, add a random k-constraint to H on literals $\pi(j), \ldots,$ $\pi(j + k - 1)$

10: $j \leftarrow j + k$

11: **end if**

$i \leftarrow i + 1$

12: **end while**

13: **return** H

key property, which will allow us to establish such relation, is that $\mathcal{H}(\mathcal{L}, w)$ is invariant under any permutation $\pi(w)$ of the elements in w as long as the number of unconsumed literals by the interpolation algorithm is bigger than k.

Lemma 14. *Let $\pi(w)$ denote an arbitrary permutation of the sequence w and b denote the number of unused literals by the algorithm, then the random energy functions $\mathcal{H}(\mathcal{L}, w)$ and $\mathcal{H}(\mathcal{L}, \pi(w))$ are equivalent conditional on $b > k$.*

Proof. We show that interchanging any adjacent pair of elements in w results in the same family of energy functions when the number of available literals is bigger than k. Write $w = x, \texttt{uni}, \texttt{con}, v$, $w' = x, \texttt{con}, \texttt{uni}, v$ and consider the execution of the algorithm on both sequences. Couple the random choices of the two runs of the algorithm on inputs w and w' for the first $|x|$ iterations so that the partial energy function H and the set of current unused literals L are the same in both executions. Note that the processing of the following two symbols, in both cases, results in the addition of a new univariate factor and possibly the addition of a new energy-clause. Thus, in order to prove that the families are equivalent is enough to show that 1) the probability $p_u(l)$ that the new univariate factor is connected to literal l is the same after processing the next two symbols for both sequences and 2) the probability $p_c(l)$ that a energy-clause of size k is connected to literal l is also the same for both sequences. By the principle of deferred decisions, we can think of the permutation of the literals at step 2) of the algorithm as generated on-the-fly, i.e., as we need occurrences to consume.When we add the univariate factor first, $p_u(l)$ is simply the probability of selecting the literal l among

the $|L|$ currently available literals, therefore,

$$p_u(l) = \frac{1}{|L|}$$

whereas when we add the univariate factor in the second step, the value of $p_u(l)$ depends, in principle, on whether a clause were added or not, events whose probabilities are denoted by $\Pr(c)$ and $\Pr(\neg c)$ respectively, and if a clause was indeed added, on whether the literal l was selected selected or not, thus

$$p_u(l) \quad = \quad p_u(l|\neg c)\Pr(\neg c) + \Pr(l|c)\Pr(c)$$

If no clause was added, no variable was used, thus

$$p_u(l|\neg c)\Pr(\neg c) = \frac{1}{|L|} \cdot \left(1 - \frac{1}{k}\right)$$

and if a clause was added,

$$\begin{aligned}
\Pr(c)p_u(l|c) &= \frac{1}{k} \cdot \left(\frac{1}{|L|-k} \cdot \Pr(\neg l)\right) \\
&= \frac{1}{k} \cdot \left(\frac{1}{|L|-k} \cdot \frac{|L|-1}{|L|} \cdot \frac{|L|-2}{|L|-1} \cdots \frac{|L|-k}{|L|-(k-1)}\right) \\
&= \frac{1}{k} \cdot \frac{1}{|L|}
\end{aligned}$$

thus

$$p_u(l) = \frac{1}{|L|}$$

The analysis for $p_c(l)$ is very similar. When the univariate factor is added second, $p_c(l)$ is equal to:

$$p_c(l) = \frac{1}{k} \cdot \left(1 - \frac{|L|-k}{|L|}\right) = \frac{1}{|L|}$$

and when the univariate factors is added first, $p_c(l)$ depends on whether the univariate factors added in previous step was connected to literal l or not:

$$p_c(l) \quad = \quad \frac{|L|-1}{|L|}\left(\frac{1}{k}\cdot\left(1-\frac{|L|-k-1}{|L|-1}\right)\right) = \frac{1}{|L|}$$

□

Lemma 15. *When $t = kr(1-\epsilon)n$ with $\epsilon > 0$, then the number of unused literals resulting after the execution of the algorithm is bigger than ϵrn w.h.p.*

Proof. The sequence $w = \texttt{con}^t$ maximizes the possible number of used literals, therefore is enough to prove the statement of the lemma is true when the input of the algorithm is $w = \texttt{con}^t$. Let X be the number of times that the algorithm tries to add a random k-constraint. Then X is a binomial random variable with $t = kr(1-\epsilon)n$ trials and probability of success $1/k$. Thus, X concentrates and the number of unused literals is at least linear in n. Formally, the probability that the algorithm leaves at most $\frac{1}{2}\epsilon krn$ unused literals is

$$\mathbf{P}(X - r(1-\epsilon)n \geq \frac{1}{2}\epsilon rn) =$$
$$\mathbf{P}(X - r(1-\epsilon)n \geq \frac{\epsilon}{2(1-\epsilon)} r(1-\epsilon)n) \leq$$
$$\exp\left(-r(1-\epsilon)n \cdot \left(\frac{\epsilon}{2\,(1-\epsilon)}\right)^2 /3\right)$$

In the last inequality we used the Chernoff bound

$$\mathbf{P}(X - \mu \geq \delta\mu) \leq \exp\left(-\mu\cdot\delta^2/3\right)$$

with $\delta \in (0,1)$. $\square$

Since the ordering does not matter, then for any $\mathcal{L}$ and any $s \leq t$, let us write $H_{\mathcal{L}}(\sigma|t,s)$ to denote the energy functions generated by the algorithm when we take t steps in total, $t-s$ of which are additions of univariate factors. Let

$$\xi_{k,r,\mathcal{L}}(t,s) = n^{-1}\mathbb{E}\left[\min_{\sigma \in D^n} H_{\mathcal{L}}(\sigma|t,s)\right] \ .$$

Observe that $\xi_{k,r,\mathcal{L}}(krn,krn)$ corresponds to the original ground state energy, whereas $\xi_{k,r,\mathcal{L}}(krn,0)$ corresponds to the ground state energy of the model composed of univariate factors only.

Our lower bounds come from the following theorem.

Theorem 16. *For any choice of* $p \in [0,1]$,

$$\xi_{k,r,\mathcal{L}} \geq \xi_{k,r,\mathcal{L}}(krn,krn) \geq \xi_{k,r,\mathcal{L}}(krn,0) - r(k-1)\mathbb{E}\left[h_c\right] - o(1) \ , \tag{3.7}$$

where

$$h_c = \min_{y_1,\ldots,y_k} \{u(y_1,..,y_k) + \sum_{i=1}^{k} b_i(y_i)\} \ .$$

To prove this we will show that as s goes from t to 0, we can control the change of $\xi_{k,r,\mathcal{L}}(t,s)$. Specifically,

Lemma 17. *If* $t = (kr-\epsilon)n$, *then for any* $\epsilon > 0$ *and all* $0 \leq s \leq t$,

$$\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,s-1)\}\right] - (k-1)k^{-1}\mathbb{E}\left[h_c\right] \leq \mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,s)\}\right] + o(1)$$

Proof of Theorem 16. Iteratively applying Lemma 17 so that we can increase the number of univariate factors from 0 to $t = (kr - \epsilon)n$ yields

$$\begin{aligned}\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,0)\}\right] - rn(k-1)\mathbb{E}\left[h_c\right] + \epsilon n(k-1)k^{-1}\mathbb{E}\left[h_c\right] \quad &\leq \\ \mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,t)\}\right] + n \cdot o(1) \ .\end{aligned}$$

The expectation $\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|krn,0)\}\right]$ is upper bounded by $\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,0)\}\right] + \epsilon n$. To see this, couple the execution of the interpolation algorithm for the first $t = (kr-\epsilon)n$ steps. In the worse case the next ϵn steps will make the ground state energy increase by one each time, thus

$$\begin{aligned}\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|krn,0)\}\right] - \epsilon n - rn(k-1)\mathbb{E}\left[h_c\right] + \epsilon n(k-1)k^{-1}\mathbb{E}\left[h_c\right] \quad &\leq \\ \mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,t)\}\right] + n \cdot o(1) \ .\end{aligned}$$

Note that $\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,t)\}\right]$ is an increasing function of t since adding constrains can only make the instances harder to satisfy i.e.,

$$\begin{aligned}\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|krn,0)\}\right] - \epsilon n - rn(k-1)\mathbb{E}\left[h_c\right] + \epsilon n(k-1)k^{-1}\mathbb{E}\left[h_c\right] \quad &\leq \\ \mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|krn,krn)\}\right] + n \cdot o(1) \ .\end{aligned}$$

Multiply the above inequality by $1/n$ to obtain

$$\begin{aligned}\xi_{k,r,\mathcal{L}}(krn,0) - \epsilon - r(k-1)\mathbb{E}\left[h_c\right] + \epsilon(k-1)k^{-1}\mathbb{E}\left[h_c\right] \quad &\leq \\ \xi_{k,r,\mathcal{L}}(krn,krn) + o(1) \ .\end{aligned}$$

Finally, taking the $\epsilon \to 0$ limit gives the bound of Theorem 16

$$\xi_{k,r,\mathcal{L}}(t,0) - r(k-1)\mathbb{E}\left[h_c\right] - o(1) \leq \xi_{k,r,\mathcal{L}}(krn,krn) \ .$$

□

Proof of Lemma 17. Let $H_{\mathcal{L}}(\sigma|t-1, s-1)$ denote the energy function resulting from executing $t-1$ steps of the algorithm where $t-(s-1)$ of such steps correspond to adding a univariate factor. The key observation is that $H_{\mathcal{L}}(\sigma|t, s-1)$ and $H_{\mathcal{L}}(\sigma|t, s)$ can be obtained from $H_{\mathcal{L}}(\sigma|t-1, s-1)$ by executing an additional step of the algorithm: $H_{\mathcal{L}}(\sigma|t, s-1)$ corresponds to the processing of a `con` element and $H_{\mathcal{L}}(\sigma|t, s)$ corresponds to the precessing of a `uni` element. We will show that conditional on any realization of H_0 of $H_{\mathcal{L}}(\sigma|t-1, s-1)$, we have

$$\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t, s-1)\}|H_0\right] - (k-1)k^{-1}\mathbb{E}\left[h_c\right] \leq \mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t, s)\}|H_0\right] + o(1) \ , \quad (3.8)$$

that is, the proof reduces to comparing the effect of adding with probability $1/k$ a single constraint to adding a single univariate factor–exactly as in the entropic interpolation method.

The proof of 3.8 is problem specific. Next we prove it for random k-SAT and random Max-k-Lin-2. For all other random CSPs with binary domains, the proof is very similar. □

3.3 Applying Energetic Interpolation to Random CSPs

Below we give the proofs of (3.8) for random k-SAT and random Max-k-Lin-2.

3.3.1 Random k-SAT

Let $C^* \subseteq \{0,1\}^n$ be the set of optimal assignments in H_0. A variable x is frozen if its value is the same in all optimal assignments. The processing of a c symbol will increase the value of the minimum by at most 1 only if the following two conditions hold: 1) a new random clause is added, which occurs with probability $1/k$, and 2) all the literals appearing in the new clause correspond to frozen variables with signs opposite to the frozen value. By the principle of deferred decisions, we can think of the permutation π as generated on-the-fly, i.e., as we need occurrences to consume. Therefore, if the number of remaining occurrences is $g(n) = \Omega(n)$ and z denotes the fraction of them that are associated with frozen variables corresponding to H_0, then, since the variables are selected without replacement, the probability of 2) is

$$2^{-k} \prod_{i=0}^{k-1} \frac{z \cdot g(n) - i}{g(n) - i} = 2^{-k} \prod_{i=0}^{k-1} (z - \frac{i}{g(n)})(1 - \frac{i}{g(n)})^{-1} \ .$$

The Taylor series expansion of $(1 - i/n)^{-1}$ at infinity is $1 + \frac{i}{n} + \frac{i^2}{n^2} + \ldots$, thus the above expression reduces to

$$2^{-k} z^k + O(\frac{1}{n}) \ .$$

Thus, the change in the ground state energy is given as follows:

$$\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,s)\}|H_0\right] - \min\{H_0\} = k^{-1} 2^{-k} z^k + O(\frac{1}{n}) \ .$$

Similarly, the processing of a u symbol will increase the value of the minimum by 1 if 1) the chosen literal corresponds to a frozen variable and 2) the variable takes the opposite of its frozen value in order to minimize the new factor $\hat{h}(x)$. Thus, the

expected change is

$$\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t, s-1)\}|H_0\right] - \min\{H_0\} = 2^{-k}p^{k-1}z \ .$$

Finally,

$$\mathbb{E}\left[h_c\right] = \mathbb{E}\left[\min_{y_1,\ldots,y_k}\{u(y_1,..,y_k) + \sum_{i=1}^{k} b_i(y_i)\}\right] = 2^{-k}p^k \ .$$

By combining the above equations and adding $-(k-1)k^{-1}2^{-k}p^k$, we get

$$\begin{aligned}\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t, s-1)\}|H_0\right] - (k-1)k^{-1}2^{-k}p^k - \mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t, s)\}|H_0\right] \\ = k^{-1}2^{-k}\left(kp^{k-1}z - z^k - (k-1)p^k\right) + O(1/n) \\ \leq O(1/n) \ ,\end{aligned}$$

since the polynomial $F(x,p) = x^k - kp^{k-1}x + (k-1)p^k \geq 0$ for all $0 \leq x, p \leq 1$. To see this last statement, note that (i) $F(0,p), F(1,p), F(x,0), F(x,1) \geq 0$ and (ii) the derivative of F with respect to p is 0 only when $p = x$, in which case $F(x,x) = 0$.

3.3.2 Random Max-k-Lin-2

In the Max-k-Lin-2 problem, the goal is to maximize the number of satisfied linear equations mod 2, where each equation has exactly k variables. The constraints in the Max-k-Lin-2 problem are chosen uniformly from the set of all $2n^k$ possible boolean equations on n variables, i.e., the k variables are chosen at random with replacement and the required parity is equally likely to be 0 or 1. Let $C^* \subseteq \{0,1\}^n$ be the set of optimal assignments in H_0. A variable x is frozen if its value is the same in all optimal assignments. The processing of a c symbol will increase the value of the minimum by at

most 1 only if the following three conditions hold: 1) a new Boolean equation is added, which occurs with probability $1/k$, 2) all the literals appearing in the new random factor correspond to frozen variables, and 3) the parity of the frozen variables is different from the one required by the new equation. As in the proof for random k-SAT above, if the number of remaining literals is $\Omega(n)$ and z denotes the fraction of them that are associated with frozen variables corresponding to H_0, then

$$\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,s)|H_0\} - \min\{H_0\} = k^{-1}2^{-1}z^k + O(1/n)\right] ,$$

where the last term is due to the fact that we are selecting without replacement. Similarly, the processing of a c symbol can increase the value of the minimum by 1 if the chosen literal corresponds to a frozen variable and the variable must take the opposite of its frozen value to minimize the added factor $\hat{h}(x)$. Thus, the expected change is given by

$$\mathbb{E}\left[\min\{H_{\mathcal{L}}(\sigma|t,s-1)\}|H_0\right] - \min\{H_0\} = 2^{-1}p^{k-1}z \ .$$

Finally,

$$\mathbb{E}\left[h_c\right] = \mathbb{E}\left[\min_{y_1,\ldots,y_k}\{u(y_1,..,y_k) + \sum_{i=1}^{k} b_i(y_i)\}\right] = 2^{-1}p^k \ .$$

Combining the above equations and adding $-(k-1)k^{-1}2^{-1}p^k$, we get

$$\begin{aligned}\mathbb{E}\left[\min\{H_{\mathcal{L},t,s-1}(\sigma)\}|H_0\right] - (k-1)k^{-1}2^{-k}p^k - \mathbb{E}\left[\min\{H_{\mathcal{L},t,s}(\sigma)\}|H_0\right] \\ = k^{-1}2^{-1}\left(kp^{k-1}z - z^k - (k-1)p^k\right) + O(1/n) \\ \leq O(1/n) \ ,\end{aligned}$$

where the r.h.s. of the equality entails the same polynomial as for random k-SAT.

3.4 Computing Explicit Energetic Interpolation Bounds for k-SAT

In this section, we compute the energetic interpolation bound for standard random k-SAT, that is, for the case in which the degree distribution of each variable is given by an independent Poisson random variable with mean rk. Applying Theorem 13, we get that

$$\begin{aligned}\xi_{k,r,\mathcal{L}}(0) &= \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\min_{\sigma\in D^n} H_{\mathcal{L}}(\sigma|krn,0)\right] \\ &= \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\min_{\sigma\in D^n}\sum_{i=1}^{n}\sum_{j=1}^{d_i}\hat{h}_{i,j}(x_i)\right] \quad ,\end{aligned}$$

where the d_i's are independent Poisson random variables with mean rk. Since all the variables are independent of each other, the minimum can be computed one variable at a time. Moreover, the expected value is the same for all variables because they are equally distributed. Thus, the above expression reduces to

$$\xi_{k,r,\mathcal{L}}(0) = \mathbb{E}\left[\min_{x\in\{0,1\}}\left(\sum_{j=1}^{s}\hat{h}_j(x)\right)\right] \quad , \tag{3.9}$$

where s is a Poisson random variables with mean kr, and the functions $\hat{h}_j(\cdot)$ are random functions in $\{I_0, I_1, I_*\}$ with $\Pr(\hat{h}_j(\cdot) = I_1) = \Pr(\hat{h}_j(\cdot) = I_0) = 2^{-k}p^{k-1}$.

Let l_0, l_1, and l_* denote the number I_0, I_1, and I_* functions respectively among the $\hat{h}_j(\cdot)$ functions inside the summation of equation (3.9). Conditional on the value of s, the random vector (l_0, l_1, l_*) is distributed as a multinomial random vector and,

therefore,

$$\xi_{k,r,\mathcal{L}}(0) \;=\; \sum_{x=0}^{\infty}\sum_{l_0=0}^{x}\sum_{l_1=0}^{x-l_0} \min\{l_0,l_1\} \times \mathrm{Poi}(kr,x)\mathrm{Multi}(l_0,l_1,x-l_0-l_1) \;\;,$$

where $\mathrm{Multi}(\cdot,\cdot,\cdot)$ denotes the multinomial density function.

Changing the limits of all summations to infinity does not change the value of $\xi_{k,r,\mathcal{L}}(0)$ since $\mathrm{Multi}(\cdot,\cdot,\cdot)$ evaluates to zero for negative numbers, hence, we can interchange the order of the summations to get

$$\xi_{k,r,\mathcal{L}}(0) \;=\; \sum_{l_0=0}^{\infty}\sum_{l_1=0}^{\infty} \min\{l_0,l_1\} \times \sum_{x=0}^{\infty} \mathrm{Poi}(kr,x)\mathrm{Multi}(l_0,l_1,x-l_0-l_1) \;\;.$$

The last equation can be simplified by summing out the randomness in the Poisson random variable. The result is that l_0 and l_1 become two independent Poisson random variables with mean $\frac{k}{2^k}rp^{k-1}$. Thus,

$$\xi_{k,r,\mathcal{L}}(0) = \sum_{l_0=0}^{\infty}\sum_{l_1=0}^{\infty} \min\{l_0,l_1\} \times \mathrm{Poi}\left(\frac{k}{2^k}rp^{k-1},l_0\right) \times \mathrm{Poi}\left(\frac{k}{2^k}rp^{k-1},l_1\right) \;\;,$$

i.e., $\xi_{k,r,\mathcal{L}}(0)$ is the expected value of the minimum of two independent Poisson random variables l_0, l_1 with mean $\lambda = \frac{k}{2^k}rp^{k-1}$. Consequently, the bound becomes

$$\xi_{k,r,\mathcal{L}} \geq \mathbb{E}\left[\min\{l_0,l_1\}\right] - \frac{(k-1)}{2^k}rp^k \;\;. \tag{3.10}$$

Finally, we note that

$$\mathbb{E}\left[\min\{l_0,l_1\}\right] = \sum_{i=0}^{\infty} i\left(2\mathrm{Poi}(\lambda,i)\left(1-\sum_{j=0}^{i-1}\mathrm{Poi}(\lambda,j)\right) - \left(\mathrm{Poi}(\lambda,i)\right)^2\right) \;\;. \tag{3.11}$$

To compute a rigorous lower bound for (3.11), one now simply truncates the sum at any desired level of accuracy.

Chapter 4

Energy Interpolation Bounds for $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$

The energetic interpolation method[1] can be naturally extended to random CSPs composed of clauses of different sizes either in its Poisson or in its combinatorial version. The key idea is that in both types of proofs it is possible to compare the change in the ground state energy resulting from adding a new random constraint and k times adding a new univariate factor for any current energy function H_0. This makes it possible to interpolate independently the 2 and 3 clauses. In this chapter, we show how to apply this generalization for $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$ formulas using the Poisson version of the proof. Recall that $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$ denotes a random CNF formula over n variables consisting of $(1 - \epsilon)n$ random 2-CNF clauses and Δn random 3-CNF clauses. As we discussed in Chapter 1, bounds for the satisfiability of $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$ immediately imply

[1] The entropic interpolation method can be extended as well using the same ideas.

exponential lower bounds for myopic DPLL algorithms on random 3-CNF satisfiable–with density smaller than known lower bounds for the satisfiability threshold–formulas.

As in Chapter 2, we modify the random model $\mathcal{F}_{2,3}(n, \epsilon, \Delta)$ by 'poissonizing' the total number of clauses of each size. In particular, let $\mathbb{F}_{2,3}(n, \epsilon, \Delta)$ denote a random CNF formula over n variables consisting of m_2 random 2-CNF clauses and m_3 random 3-CNF clauses, , where m_2 is a Poisson random variable with mean $\mathbb{E}[m_2] = (1-\epsilon)n$ and m_3 is a Poisson random variable with mean $\mathbb{E}[m_3] = \Delta n$. Thus, the energy function is now

$$H_{2,3}(\sigma|n, \epsilon, \Delta) = H_2(\sigma|n, 1-\epsilon) + H_3(\sigma|n, \Delta) \ .$$

Similarly, the interpolation function is the sum of the two independent interpolation functions corresponding to $k = 2$ and $k = 3$, i.e.,

$$H_{2,3}(\sigma|n, \epsilon, \Delta, t) = H_2(\sigma|n, 1-\epsilon, t) + H_3(\sigma|n, \Delta, t) \ . \tag{4.1}$$

Letting

$$\xi_{\epsilon,\Delta}(t) = \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\min_{\sigma\in\{0,1\}^n} H_{2,3}(\sigma|n, \epsilon, \Delta, t)\right] \ , \tag{4.2}$$

the analogue of Theorem 10 for random mixtures of 2- and 3-clauses is the following:

Theorem 18. *For every value of* $p \in [0, 1]$,

$$\xi_{\epsilon,\Delta} \geq \xi_{\epsilon,\Delta}(0) - \frac{1}{4}(1-\epsilon)p^2 - \frac{1}{4}\Delta p^3 \ . \tag{4.3}$$

Proof. As with Theorem 10, the joint probability distribution implicit in the expectation of $\xi_{\epsilon,\Delta}(t)$ can be written as the product of Poisson random functions, due to the independence among the random variables appearing in $H_{2,3}(\sigma|n, \epsilon, \Delta, t)$. Now, the

derivative with respect to t gives rise to two independent set of equations similar to the ones in (2.14) and (2.15) for $k = 2$ and $k = 3$, where the base energy function is $H_{2,3}(\sigma|n, \epsilon, \Delta, t)$. Since all the relevant properties of the mixture are captured by its set of frozen variables, the theorem follows simply by applying the proof of (2.8) in Theorem 10 twice. □

4.1 Explicit Computation of the Bound

We conclude this chapter by computing explicit bounds for the r.h.s. of equation (4.3). In particular, we show that $\xi_{\epsilon,\Delta} > 0$ for $\epsilon = .0001$ and $\Delta = 1.0001$, thus proving

Theorem 19. *Let F be a random CNF formula on n variables with $(1-\epsilon)n$ random 2-clauses, and $(1+\epsilon)n$ random 3-clauses. W.h.p. F is unsatisfiable for $\epsilon = 10^{-4}$.*

The computation of $\xi_{\epsilon,\Delta}(0)$ is very similar to its analogue in Section(3.4).

$$\begin{aligned}
\xi_{\epsilon,\Delta}(0) &= \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\min_{\sigma\in\{0,1\}^n} H_{2,3}(\sigma|n,\epsilon,\Delta,0)\right] \\
&= \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\min_{\sigma\in\{0,1\}^n}\left(\sum_{i=1}^{n}\left(\sum_{j=1}^{k_{2,i}}\hat{h}_{2,i,j}(x_i)+\sum_{j=1}^{k_{3,i}}\hat{h}_{3,i,j}(x_i)\right)\right)\right] \\
&= \lim_{n\to\infty} n^{-1}\mathbb{E}\left[\sum_{i=1}^{n}\min_{x_i\in\{0,1\}}\left(\sum_{j=1}^{k_{2,i}}\hat{h}_{2,i,j}(x_i)+\sum_{j=1}^{k_{3,i}}\hat{h}_{3,i,j}(x_i)\right)\right] \\
&= \lim_{n\to\infty} n^{-1}\sum_{i=1}^{n}\mathbb{E}\left[\min_{x_i\in\{0,1\}}\left(\sum_{j=1}^{k_{2,i}}\hat{h}_{2,i,j}(x_i)+\sum_{j=1}^{k_{3,i}}\hat{h}_{3,i,j}(x_i)\right)\right] \quad ,
\end{aligned}$$

where the $k_{2,i}$'s and the $k_{3,i}$'s are Poisson random variables with means $2(1-\epsilon)$ and 3Δ, respectively, as defined in (4.1). Note that the n expectations in the above summation

are identical, thus

$$\xi_{\epsilon,\Delta}(0) = \mathbb{E}\left[\min_{x\in\{0,1\}}\left(\sum_{j=1}^{s_2}\hat{h}_{2,j}(x) + \sum_{j=1}^{s_3}\hat{h}_{3,j}(x)\right)\right] \ , \tag{4.4}$$

where s_2 and s_3 are Poisson random variables with means $2(1-\epsilon)$ and 3Δ, respectively; and the functions $\hat{h}_{2,j}(\cdot)$ and $\hat{h}_{3,j}(\cdot)$ are i.i.d. copies of the function $\hat{h}(\cdot)$ in (3.4) for $k = 2$ and $k = 3$, respectively, i.e., they are random functions in $\{I_0, I_1, I_*\}$ with $\Pr(\hat{h}_{k,j}(\cdot) = I_1) = \Pr(\hat{h}_{k,j}(\cdot) = I_0) = 2^{-k}p^{k-1}$.

Let $l_{k,0}$, $l_{k,1}$, and $l_{k,*}$ denote the number of I_0, I_1, and I_* functions, respectively among the $\hat{h}_{k,j}(\cdot)$ functions inside the summation in (4.4). Conditional on the value of s_k, the random vector $(l_{k,0}, l_{k,1}, l_{k,*})$ is distributed as a multinomial random vector with s_k trials and probability vector $(2^{-k}p^{k-1}, 2^{-k}p^{k-1}, 1-2^{-k+1}p^{k-1})$, therefore,

$$\begin{aligned}\xi_{\epsilon,\Delta}(0) \ = \ & \sum_{x=0}^{\infty}\sum_{y=0}^{\infty}\sum_{l_{2,0}=0}^{x}\sum_{l_{2,1}=0}^{x-l_{2,0}}\sum_{l_{3,0}=0}^{y}\sum_{l_{3,1}=0}^{y-l_{2,0}} \min\{l_{2,0}+l_{3,0}, l_{2,1}+l_{3,1}\} \times \\ & \mathrm{Poi}(2(1-\epsilon), x)\mathrm{Multi}(l_{2,0}, l_{2,1}, x-l_{2,0}-l_{2,1}) \times \\ & \mathrm{Poi}(3\Delta, y)\mathrm{Multi}(l_{3,0}, l_{3,1}, y-l_{3,0}-l_{3,1}) \ ,\end{aligned}$$

where $\mathrm{Multi}(\cdot,\cdot,\cdot)$ denotes the multinomial density function.

Changing the limits of all summations to infinity, does not change the value of $\xi_{\epsilon,\Delta}(0)$, since $\mathrm{Multi}(\cdot,\cdot,\cdot)$ evaluates to zero for negative numbers, hence, we can

interchange the order of the summations to get

$$\begin{aligned}\xi_{\epsilon,\Delta}(0) &= \sum_{l_{2,0}=0}^{\infty}\sum_{l_{2,1}=0}^{\infty}\sum_{l_{3,0}=0}^{\infty}\sum_{l_{3,1}=0}^{\infty}\min\{l_{2,0}+l_{3,0},l_{2,1}+l_{3,1}\}\times \\ &\quad \sum_{x=0}^{\infty}\mathrm{Poi}(2(1-\epsilon),x)\mathrm{Multi}(l_{2,0},l_{2,1},x-l_{2,0}-l_{2,1})\times \\ &\quad \sum_{y=0}^{\infty}\mathrm{Poi}(3\Delta,y)\mathrm{Multi}(l_{3,0},l_{3,1},y-l_{3,0}-l_{3,1}) \ .\end{aligned}$$

The last equation can be simplified by summing out the randomness in the Poisson random variables. The result is that $l_{2,0}$ and $l_{2,1}$ become two independent Poisson random variables with mean $\frac{1}{2}(1-\epsilon)p$, that is,

$$\begin{aligned}&\sum_{x=0}^{\infty}\mathrm{Poi}(2(1-\epsilon),x)\mathrm{Multi}(l_{2,0},l_{2,1},x-l_{2,0}-l_{2,1}) \ = \\ &\qquad \mathrm{Poi}((1-\epsilon)p/2,l_{2,0})\times\mathrm{Poi}((1-\epsilon)p/2,l_{2,1}) \ .\end{aligned}$$

Similarly, $l_{3,0}$ and $l_{3,1}$become two independent Poisson random variables with mean $\frac{3}{8}\Delta p^2$. Moreover, letting

$$\lambda = \frac{1}{2}(1-\epsilon)p+\frac{3}{8}\Delta p^2 \ ,$$

we see that $l_0 = l_{2,0}+l_{3,0}$ is itself a Poisson random variable with mean λ, since the sum of two independent Poisson random variables with means λ_1 and λ_2 is a Poisson random variable with mean $\lambda=\lambda_1+\lambda_2$. Thus,

$$\xi_{\epsilon,\Delta}(0)=\sum_{l_0=0}^{\infty}\sum_{l_1=0}^{\infty}\min\{l_0,l_1\}\times\mathrm{Poi}\left(\lambda,l_0\right)\times\mathrm{Poi}\left(\lambda,l_1\right) \ ,$$

i.e., $\xi_{\epsilon,\Delta}(0)$ is the expected value of the minimum of two independent Poisson random variables l_0,l_1 with mean λ. Consequently, the bound of Theorem 18 becomes

$$\xi_{\epsilon,\Delta}\geq\mathbb{E}\left[\min\{l_0,l_1\}\right]-\frac{1}{4}(1-\epsilon)p^2-\frac{1}{4}\Delta p^3 \ . \tag{4.5}$$

Finally, we note that

$$\mathbb{E}\left[\min\{l_0, l_1\}\right] = \sum_{i=0}^{\infty} i \left(2\mathrm{Poi}(\lambda, i) \left(1 - \sum_{j=0}^{i-1} \mathrm{Poi}(\lambda, j) \right) - \left(\mathrm{Poi}(\lambda, i)\right)^2 \right) \quad . \tag{4.6}$$

Thus, to compute lower bounds for (4.5) it is enough to truncate (4.6) at any value of i. In particular, by letting $\epsilon = 0.0001$, $\Delta = 1.0001$ and $i = 50$, we get that for $p = 1.2 \cdot 10^{-3}$ the truncated version of (4.5) is greater than 0, implying that a random CNF formula with $0.9999n$ 2-clauses and $1.0001n$ 3-clauses is unsatisfiable w.h.p.

Chapter 5

Conclusions and Future Work

In this thesis, we used the interpolation method to prove unsatisfiability bounds for random CSPs. The core idea of the interpolation method is to set up a process that transforms a complex quantity into a manageable quantity in a controlled way. Franz and Leone [24] used the free entropy density as the main quantity and derived upper bounds for random k-SAT and random k-XOR-SAT that coincide with the ones derived using the cavity method from statistical physics. In particular, their RS cavity bounds come from using the free entropy of a model where all variables are independent, as the manageable quantity. However, it was difficult to compute explicit rigorous bounds due to the complexity of the resulting expressions. More precisely, the RS bound is a functional of a probability density γ of a single variable. The bound is valid for all possible densities γ, therefore, the best result is obtained by minimizing the functional with respect to γ. Unfortunately, solving the optimization problem analytically proved very difficult, which is the reason why physicists use stochastic numerical methods

instead. Although numerical methods gave 'reasonable' explicit bounds for random 3-SAT, they were not suited to manage the continuous transition of the fraction of frozen variables present in models like the mixture model composed of random CNF formulas of sizes 2 and 3..

We applied the interpolation method directly to the energy function to avoid the analytical difficulties of the entropic interpolation method by explicitly modeling the propagation of frozen variables. This allowed us to compute new rigorous upper bounds for the satisfiability threshold of random formulas composed of $(1-\epsilon)n$ random 2-clauses and Δn random 3-clauses in which we where able to prove that for $\Delta > 1.0001$ there exist an $\epsilon > 0$ such that the resulting mixtures are unsatisfiable w.h.p.. This improvement over the old result in which 1.0001 is replaced by 2.28 helped us to prove exponential lower bounds on the running time of DPLL algorithms on random 3-CNF formulas at densities that can be solved with positive probably by existing algorithms.

Two key elements in the application of the entropic interpolation method to random CSPs are 1) that the proof reduces to establishing a bound in the change of the free entropy when a new constraint is added versus k times the change when a new univariate factor is added and 2) that the proof of such a bound holds for any arbitrary initial energy function H_0. Identifying these two points helped us to apply the entropic interpolation method to random k-hyper graph coloring, which is the first time that the method is successfully applied to models with non-binary variables. But more importantly, it helped us to extend both the entropic and the energetic interpolation method to random CSPs with an arbitrary sparse degree sequence. We hope

this expands the applicability of the method, particularly, to CSPs that result from preprocessing standard random CSPs.

5.1 Towards Better Satisfiability Upper Bounds

In principle, we should be able to produce better bounds with the interpolation method by increasing the complexity of the object towards which we want to interpolate. In all versions of the interpolation method studied in this thesis we used quantities based on a model in which all the variables are independent of each other. A natural increase in complexity arises from choosing a model that is a convex combination of independent models. In the context of the entropic interpolation method, the resulting bounds using this new model coincide with the 1-RSB cavity bounds when the appropriate convex combination is chosen[24, 47]. In this case, the resulting expressions are functionals of a distribution over the set of all possible single variable distributions, which makes the explicit computation of bounds very difficult, even using numerical methods only. We were also able to prove energetic interpolation bounds using a convex combination of independent models. However, we could not find a convex combination that would give better bounds than the ones presented in this thesis.

From a message passing point of view (see Appendix (A)), the 1-RSM cavity method can be seen as the result of applying Belief Propagation to the factor graph that represents the fixed point equations of Belief Propagation when applied to the factor graph of the original CSP. In this new factor graph, the variables are the

messages and the factors enforce that the Belief Propagation fixed point equations are satisfied and also give each fixed point a weight exponential to its free entropy. The idea is to capture the existence of multiple Belief Propagation fixed points. However, even running Population Dynamics in this new factor graph is very inefficient. A more efficient approach is to apply Belief Propagation to the factor graph that represents the Warning Propagation fixed point messages instead [42]. From using this procedure, it is possible to estimate the number of Warning Propagation fixed points at each energy density i.e., if the results were rigorous, then the satisfiability threshold could be determined by finding the density at which the number of fixed points at zero energy goes from a positive number to 0. Using this method, it was estimated that the satisfiability threshold for random 3-SAT is 4.267 [40]. Also, our own calculation using this method gives that $\Delta_c = 2/3$, where Δ_c is defined as the largest Δ such that for every $\epsilon > 0$, a mixture of $(1-\epsilon)n$ 2-clauses and Δn 3-clauses is satisfiable *w.h.p.* Recall that in [6] the authors proved that $2/3 \leq \Delta_c < 2.28...$ Therefore, if the method is right then it would imply that 2/3 is tight. We believe the method produces correct upper bounds and our current research is focused on formalizing it using the interpolation method.

Appendix A

The RS Cavity Method

In this chapter we discuss the RS cavity method approximation of the free entropy density. We begin by describing the RS cavity method approximation of the free entropy density of a single instance, then, we assume that the instance is taken from a random ensemble and analyze its average behavior as the number of variables tends to infinity.

A.1 Computation of the Free Entropy of Single Instances

Let $I(X, C)$ denote a CSP instance with set of constraints C and set of variables X. For notational simplicity, we assume that all variables $x \in X$ share the same[1] finite domain D. We will use x_a to denote the set of variables that appear in constraint $a \in C$ and ∂x to denote the set constraints in which variable $x \in X$ participates. We will write the energy function $H(\sigma|I)$ as the sum of the $|C|$ functions $u_a(x_a)$, one for each

[1]The more general case in which each variable has its own domain can also be analyzed using the methods described here.

constraint $a \in C$, i.e. $u_a(x_a) = 1$ if constraint a is unsatisfied and 0 otherwise. For example, for k-SAT, we take the domain of the variables to be $\{+1, -1\}^n$ and for each k-clause a, with set of variables $x_a = \{x_{a,1}, \ldots, x_{a,k}\}$, we let

$$u_a(x_a) = \prod_{j=1}^{k} \frac{1 + J_{a,j} x_{a,j}}{2} \ , \tag{A.1}$$

where $J_{a,j} \in \{+1, -1\}$ represents the sign of literal $x_{a,j}$ in the clause: $+1$ if the literal is negated and -1 otherwise. Similarly, for k-uniform hypergraph coloring, $k = 2$ being graph coloring, each constraint is

$$u_a(x_a) = \mathbf{1}_{x_{a,1}=x_{a,2}=\ldots=x_{a,k}} \ , \tag{A.2}$$

where $\mathbf{1}_{\mathcal{E}}$ is the indicator function for the event $\mathcal{E}$.

Recall that in the statistical physics approach, the set of all assignments of a CSP is endowed with the Gibbs probability measure

$$P(\sigma|\beta, I) = \frac{1}{Z} e^{-\beta H(\sigma|I)} \ , \tag{A.3}$$

where $H(\sigma|I)$, known as the energy function or hamiltonian, counts the number of unsatisfied constraints in the instance I under the assignment σ, $\beta > 0$ is a constant, and Z, known as the partition function, is the constant that makes $P(\sigma|\beta, I)$ a valid probability density,

$$Z(\beta) = \sum_{\sigma \in D^n} \exp(-\beta H(\sigma|n, r)) \ .$$

Observe that the Gibbs distribution $P(\sigma|\beta, I)$ factorizes according to the constraints in C, that is,

$$P(\sigma|\beta, I) = \frac{1}{Z} \prod_{a \in C} \exp(-\beta u_a(x_a)) \ .$$

To lighten the notation, we will sometimes use $\psi_a(x_a)$ to denote the factor $\exp(-\beta u_a(x_a))$, hence,

$$P(\sigma|\beta, I) = \frac{1}{Z} \prod_{a \in C} \psi_a(x_a) \ .$$

A lot of effort has been put into developing tools to analyze probability distributions like $P(\sigma|\beta, I)$, that is, probability distribution that factor according to a bipartite graph where the factor vertices are connected to a small number of variables. The idea is to take advantage of the independence relations imposed by the graph in order to develop efficient algorithms for solving the following *inference* problems:

1. Compute the marginal distribution $P(A|\beta, I)$ over a particular subset $A \subset X$ of the variables in the CSP i.e.

   $$P(A|\beta, I) = \frac{1}{Z} \sum_{X \setminus A} \prod_{a \in C} \psi_a(x_a) \ .$$

   Note that if these marginals can be computed, then conditional distributions of the form $P_I(A|B, \beta, I)$ for a disjoint subset A and B can be computed as well, since $P(A|B, \beta, I) = P(A, B|\beta, I)/P(B|\beta, I)$.

2. Computing the maximum or mode of the joint density

   $$\max_{\sigma \in D^n} \{P(\sigma|\beta, I)\}$$

   and the assignment σ^* that achieves it

   $$\sigma^* = \arg\max_{\sigma \in D^n} \{P(\sigma|\beta, I)\} \ ,$$

   i.e. find the assignment that minimizes the function $H(\cdot|I)$.

3. Compute the free entropy

$$F(\beta|I) = \log\left(\sum_{\sigma}\prod_{a\in C}\psi_a(x_a)\right) \ ,$$

Among the most successful approaches at solving the above problems are the Markov chain Monte Carlo (MCMC) framework [28, 48] and message passing type of algorithms [38, 41]. The latter, as we show in the next section, can solve the problems exactly when the underlying factor graph is a tree, but when applied to general graphs message passing algorithms are used just as approximations. In [51] it was shown that many message passing algorithms can be seen as a variational method. Still, in the general case, the computations can not be used to give rigorous upper/lower bounds of the desired quantities.

A.2 Belief Propagation and the Computation of the Free Entropy of Arbitrary Instances

Consider the probability distribution

$$P(\sigma|\beta, I) = \frac{1}{Z}\prod_{a\in C}\psi_a(x_a) \ .$$

defined in the previous section. The Belief Propagation algorithm was designed to compute the marginal distribution of each of the variables in $P(\cdot)$ when its associated factor graph is a tree[36]. The solution is based on dynamic programing over a tree i.e. to compute the marginal distribution of a variable, we first root the tree at such variable and for each edge of type $a \to x$ (from factor node to variable node) we define

the subproblem $\hat{v}_{a\to x}(x)$, or message in the language of BELIEF PROPAGATION, as the marginal distribution of variable x for the factor tree $T_{a\to x}$ composed of the sub-tree under and including the edge $a \to x$ (see Figure A.1). Similarly for an edge of type $y \to a$ (from variable node to factor node) we define the message $v_{y\to a}(y)$ as the marginal distribution of variable y for the sub-tree T_y rooted at variable node y (see Figure A.1). Both messages can be computed in constant time ($O(k)$) from the messages coming from its children. In the following $V(T)$ and $C(T)$ denote the set of variables and constraints respectively contained in the tree T. By definition, the message (marginal) from constraint to variable is computed by summing out all the variables in $V(T_{a\to x})\backslash x$:

$$\hat{v}_{a\to x}(x) \cong \sum_{V(T_{a\to x})\backslash x} \left(\prod_{c\in C(T_{a\to x})} \psi_c(x_c) \right) \ .$$

Observe that $V(T_y) \cap V(T_z) = \emptyset$ for each pair of variables $y, z \in x_a\backslash x$, therefore, the above summation can be organized as follows:

$$\begin{aligned} \hat{v}_{a\to x}(x) &\cong \sum_{x_a\backslash x} \psi_a(x_a) \prod_{y\in x_a\backslash x} \left[\sum_{V(T_y)\backslash y} \left(\prod_{c\in C(T_y)} \psi_c(x_c) \right) \right] \\ &= \sum_{x_a\backslash x} \psi_a(x_a) \prod_{y\in x_a\backslash x} v_{y\to a}(y) \ . \end{aligned}$$

In other words the joint density $P(x_a\backslash x)$ can be written in product form because the variables in $x_a\backslash x$ are independent of each other when the constraint a is removed from the tree. The message from variable x to factor a can be computed similarly resulting in the following expression:

$$v_{x\to a}(x) \cong \prod_{b\in \partial x\backslash a} \hat{v}_{b\to x}(x) \ .$$

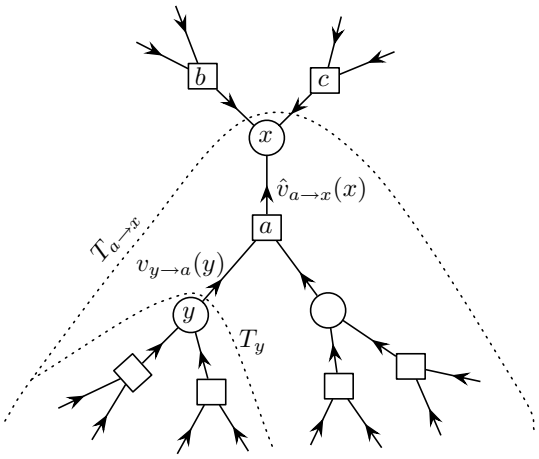

Figure A.1: Messages in the Belief Propagation algorithm.

Finally, the marginal of the variable node at the root is proportional to the product of the probabilities computed by its children factor nodes.

In order to compute all the marginals we could run the algorithm n times, however, it is possible to compute the marginals more efficiently by noting that trees rooted at different variables share some directed subtrees i.e. the message associated with those subtrees could be computed only once. One way to implement this idea is by viewing the messages in dynamic programing as variables of a discrete time dynamical system whose dynamics is governed in the same way as the computation of the messages in dynamic programming. More precisely, there are, at iteration t, two messages $v_{a\to x}^{(t)}$ and $\hat{v}_{x\to a}^{(t)}$ for each edge (a, x) in the factor graph and its dynamics is given by the following equations:

$$v_{x\to a}^{(t+1)}(x) \cong \prod_{b\in\partial x\setminus a} \hat{v}_{a\to x}^{(t)}(x) \tag{A.4}$$

$$\hat{v}^{(t)}_{a\to x}(x) \cong \sum_{x_a \setminus x} \psi_a(x_a)) \prod_{y \in x_a \setminus x} v^{(t)}_{y\to a}(y) \tag{A.5}$$

A Belief Propagation fixed point of these equations is the set of t-independent messages $v_{a\to x}$ and $\hat{v}_{x\to a}$ which satisfy the previous equations. In the case of a tree factor graph the correctness of the dynamic programming algorithm suffices to prove the convergence of the dynamical system.

Theorem 20. *(*Belief Propagation *is exact on trees [41]) Consider a tree factor graph with diameter*[2] t_**. Then*

1. Irrespective to the initial conditions, the Belief Propagation update rules converge after at most t_* iterations. In other words, for any edge (x, a) and any $t > t_*$, we have $v^{(t)}_{a\to x} = v_{a\to x}$ and $\hat{v}^{(t)}_{x\to a} = \hat{v}_{x\to a}$.

2. The fixed point equations provide the exact marginals

$$P(x) \cong \prod_{a \in \partial x} \hat{v}_{a\to x}(x)$$

Once the Belief Propagation has converged, the free entropy can by computed using the fixed point messages v and $\hat{v}$ according to the following equation:

$$F(\beta|I) = F(\beta|v, \hat{v}) = \sum_{a\in C} F_a(v) + \sum_{x\in V} F_x(\hat{v}) - \sum_{(x,a)\in E} F_{x,a}(v, \hat{v}) \tag{A.6}$$

where

$$F_a(v) = \log(\sum_{x_a} \psi_a(x_a) \prod_{y\in x_a} v_{y\to a}(y))$$

[2]The diameter of a graph is the length of the greatest shortest path in the graph [20].

$$F_x(\hat{v}) = \log(\sum_x \prod_{b \in \partial x} \hat{v}_{b \to x}(x))$$

$$F_{x,a}(v, \hat{v}) = \log(\sum_x v_{x \to a}(x) \hat{v}_{a \to x}(x)) \ ,$$

and E denotes all the of edges in the factor graph associated to I i.e. $(x, a) \in E$ iff $x \in x_a$. The main idea behind the derivation of Eq. A.6 is that the probability distribution of a tree factor graph can be expressed as the product of local marginals as the following theorem shows.

Theorem 21. *In a tree factor graph T, the joint probability distribution $P(\sigma)$ of all the variables can be written in term of the marginals $P(x_{\partial a})$ and $P(x)$ as follows:*

$$P(\sigma) = \prod_{a \in C} P(x_{\partial a}) \prod_{x \in V} P(x)^{1-|\partial x|} \ .$$

Therefore, the free entropy of a tree factor graph can be written in terms of local marginals too, which in turn can be expressed in terms of the fixed point messages. For a complete derivation see [41].

Note that the Belief Propagation equations are properly defined even when the factor graph is not a tree. Thus, one could try to approximate the marginals of a non-tree factor graph using the same iterative procedure. The hope is that the joint densities $P(x_a \backslash x)$ of the graph obtained after removing a factor node can be approximated by writing the joint density in product form. Indeed, this is what happens in locally like tree CSPs until long range correlations start to emerge.

The behavior of Belief Propagation and other message passing algorithms on factor graphs with cycles can be understood in the context of variational methods

[51]. The idea is to express the free entropy as the solution of an optimization problem. Then, the optimization problem can be relaxed either by approximating the function to be optimized or by approximating the set over which the function is being optimized. Yedida et. al. [52] were able to show that Belief Propagation can be seen as a Lagrangian method applied to an optimization problem in which both the function to be optimized and the set over which the optimization takes places are being approximated. Unfortunately, a consequence of those approximations is that there is no guarantee of convergence, and the Belief Propagation algorithm can converge to one of possibly many different fixed points. Also, there is the possibility that some of the fixed points do not correspond to a valid probability distributions. All in all, this prevents the use of the computations made by Belief Propagation as valid upper/lower bounds.

In summary, the RS computation of the free entropy consist of iterating equations A.4 and A.5 until convergence and then use Eq. A.6 to compute the free entropy. This computation is based on the following assumptions:

1. Belief Propagation computes the right 'local' marginals, namely, it computes the right values of the single variable marginals $P(x)$ and the k-variable marginals $P(x_a)$ associated with the k-variables that appear in each one of the factors.

2. The free entropy can be expressed as in Eq. (A.6) i.e. as the free entropy of a tree factor graph.

A.3 Entropic RS Approximation of the Free Entropy Density

We described the RS computation of the free entropy for an arbitrary instance. Now consider the random family $\mathcal{I}_k(n, rn)$ of CSP instances formed by picking $m = rn$ constrains uniformly, independently and with replacement from the set $C_{k,n}$ composed of all possible constraints of k variables that can be built from n variables. For random k-SAT $C_{k,n} = B_{k,n}$ denotes the set of all possible disjunctions of k distinct, non-complementary literals from its variables, and for random hyper-graph coloring $C_{k,n}$ is the set of all possible k-element subsets of the set of n vertices.

When the instance comes from the family $\mathcal{I}_k(n, rn)$ the free entropy becomes a random object and we are interested in the asymptotic growth of its expected value relative to the number of variables

$$f_k(\beta|r) = \lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[F_k(\beta|n, r)\right] \quad , \tag{A.7}$$

because an negative upper bound for $f_k(\beta|r)$ implies that r is an upper bound for the satisfiability threshold of the random CSP under study (see Section 1.3).

The RS cavity method approach to estimate the free entropy density is based on the asymptotic analysis of the distribution $\gamma^{(t)}$ of the Belief Propagation message passing at iteration t through a random variable-to-factor edge of the factor graph of a random CSP–recall that a message is itself a distribution on D i.e. $\gamma^{(t)}$ is a distribution over the possible distributions on D–In the case of an arbitrary instance, the message

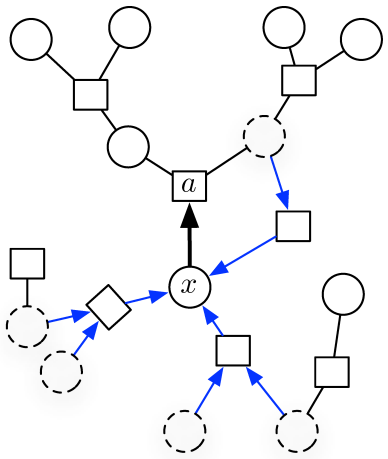

Figure A.2: Neighborhood or radius 1 for the edge $x \to a$.

passing through an edge $e = (x, a)$ at iteration t depends only on the initial variable-to-factor messages coming out from variable nodes connected to x through a directed path whose number of factor nodes is at most t (Figure A.2 shows an example with $t = 1$). We define the neighborhood of directed edge $e = (x, a)$ of radius t, denoted by $B_I(e, t)$, as the sub-graph of I composed of all the variable nodes at a distance–the distance is the number of factor nodes in a directed path between two variable nodes–at most t to x and all the factor nodes connected *only* to those variables.

The RS computation approximates $\gamma^{(t)}$ with the distribution of the fixed point messages of a random edge of an infinite random factor tree with the same local structure as the family $\mathcal{I}_k(n, rn)$. The local structure of radius t of a random edge $x \to a$ in the random infinite tree is described using the following generative procedure: To build a neighborhood of radius one, generate l new random factor nodes and connect them to variable node x, where l is a random variable with the same distribution as the random

variable describing the number of factor nodes adjacent to a random edge $x \to a$ in the $\mathcal{I}_k(n, rn)$ model i.e. a Poisson random variable with mean rk. Then for each new factor node generate $k-1$ new variable nodes and attach them to it. To generate a radius two neighborhood apply the same procedure to the variable nodes just generated in the previous step. Follow this procedure recursively to obtain a neighborhood of any radius t. One consequence of this recursive characterization is that the local structure of all the edges in the infinite tree has the same distribution. This allows us to express the distribution of the fixed point message flowing through a random edge $x \to a$ as a transformation of the BP fixed point equations into distributional equations. More precisely, γ and $\hat{\gamma}$ satisfy the following distributional equations

$$v(x) \cong \prod_{j \leq l} \hat{v}_j(x) \tag{A.8}$$

$$\hat{v}(x) \cong \sum_{y_1,\ldots,y_{k-1}} \psi(y_1, ..., y_{k-1}, x) \prod_{j \leq k-1} v_j(y_j) \ , \tag{A.9}$$

if 1) $\hat{v}(\cdot)$ has distribution $\hat{\gamma}$ when $\{v_j(\cdot)\}_{j=1}^{k-1}$ are i.i.d. random functions with distribution γ and $\psi(\cdot)$ is the factor associated with a random constraint and 2) $v(\cdot)$ has distribution γ when $\{\hat{v}_j(\cdot)\}_{j=1}^{l}$ are i.i.d. random functions with distribution $\hat{\gamma}$ and l is a Poisson random variable with mean rk. It is very hard to solve these distributional equations analytically, so one has to use numerical methods like the Population Dynamics algorithm [1]. Before describing the algorithm, we show how the free entropy density looks like once one takes the expected value of Eq. A.6 with respect to the the distributions γ and $\hat{\gamma}$:

$$\mathbb{E}[F_k(\beta|n,r)] = \mathbb{E}[F_k(\beta|v,\hat{v})] = \sum_{a\in C}\mathbb{E}[F_a(v)] + \sum_{x\in V}\mathbb{E}[F_x(\hat{v})] - \sum_{(x,a)\in E}\mathbb{E}[F_{x,a}(v,\hat{v})] \ .$$

Since all the constraints in a $\mathbb{I}_k(n, rn)$ instance have the same distribution, then in the $n \to \infty$, the expected value $\mathbb{E}[F_a(v)]$ is the same for all but a finite set of constraints, and similarly for the expectations $\mathbb{E}[F_x(\hat{v})]$ and $\mathbb{E}[F_{x,a}(v,\hat{v})]$. Thus the RS free entropy density is given by

$$\begin{aligned} f_k^{(RS)}(\beta|r) &= f_k^{(RS)}(\beta|\gamma,\hat{\gamma}) \\ &= \lim_{n\to\infty}\frac{1}{n}\mathbb{E}[F_k(\beta|v,\hat{v})] \\ &= r\mathbb{E}[F_a(v)] + \mathbb{E}[F_x(\hat{v})] - rk\mathbb{E}[F_{x,a}(v,\hat{v})] \ , \end{aligned} \tag{A.10}$$

with

$$\mathbb{E}[F_a(v)] = \mathbb{E}\left[\log(\sum_{x_1,\ldots,x_k}\psi(x_1,\ldots,x_k)\prod_{j=1}^{k}v_j(x_j))\right]$$

$$\mathbb{E}[F_x(\hat{v})] = \mathbb{E}\left[\log(\sum_{x_i}\prod_{j=1}^{l}\hat{v}_j(x_j))\right]$$

$$\mathbb{E}[\mathbb{F}_{x,a}(v,\hat{v})] = \mathbb{E}\left[\log(\sum_{x}v(x)\hat{v}(x))\right] \ ,$$

where $v(x)$ and $\{v_j(\cdot)\}_{j=1}^{k}$ are i.i.d. random functions with distribution γ, $\psi(\cdot)$ is the factor associated with a random constraint, $\hat{v}(x)$ and $\{\hat{v}_j(\cdot)\}_{j=1}^{l}$ are i.i.d. random functions with distribution $\hat{\gamma}$ and l is a Poisson random variable with mean rk. The above

expression for the free entropy density is the one commonly used for numerical computations like Population Dynamics, however, it can be further simplified by writing it in terms of γ only. The resulting expression coincides with the rigorous bound obtained using the entropic interpolation method of Section (2.1).

A.3.1 Population Dynamics Algorithm

The goal of the Population Dynamics algorithm is to generate sets of i.i.d. samples from γ (and $\hat{\gamma}$). To do so, it maintains at each iteration t, two sets $\gamma_{PD}^{(t)}$ and $\hat{\gamma}_{PD}^{(t)}$ of size n_{PD}, which represent the current approximation of the algorithm. To improve the estimate $\hat{\gamma}_{PD}^{(t-1)}$, the algorithm assumes that the current estimate $\gamma_{PD}^{(t-1)}$ is correct and use it to sample a new set $\hat{\gamma}_{PD}^{(t)}$ using equation A.9, then the set $\gamma_{PD}^{(t-1)}$ is updated using the new estimate $\hat{\gamma}_{PD}(t)$ and equation A.8. The hope is that, after a reasonable number of iterations, this Markov Chain will give samples that correctly represent the empirical distribution of γ and $\hat{\gamma}$. The pseudo-code of the Algorithm is given below. The inputs are the size of the populations n_{PD} and the total number of iteration n_T. We use $\hat{\gamma}_{PD}^{(t)}[i]$ and $\gamma_{PD}^{(t)}[i]$ to denote the i-th sample in the sets $\hat{\gamma}_{PD}^{(t)}$ and $\gamma_{PD}^{(t)}$ respectively.

Figures A.3, A.4 and A.5 show the empirical distribution of γ obtained by applying Population Dynamics to random 3-SAT for $r = 0.1$, $r = 0.3$ and $r = 0.45$ respectively. Note, that for small densities most of the probability mass is around $1/2$, but as more constraints are added the probability that a variable is biased towards either 0 or 1 increases i.e. more and more variables are forced to take a particular value in order to satisfy all the constraints. It is important to mention that non of the samples

Algorithm 4 Population Dynamics

Input: n_{PD}, n_T,

Initialize $\gamma_{PD}(0)$

for $t = 1, ..., n_T$ **do**

for $i = 1, ..., n_{PD}$ **do**

Draw $s_1, ..., s_{k-1}$ uniformly in $\{1, ..., n_{PD}\}$

Draw a random contraint-function $\psi(\cdot)$

Set $\hat{\gamma}_{PD}^{(t)}[i](x) \approx \sum_{y_1,...,y_{k-1}} \psi(y_1, ..., y_{k-1}, x) \prod_{j \leq k-1} \gamma_{PD}^{(t-1)}[s_j](y_j)$

end for

for $i = 1, ..., n_{PD}$ **do**

Draw a random sample l from Poi(rk)

Draw $s_1, ..., s_l$ uniformly in $\{1, ..., n_{PD}\}$

Set $\gamma_{PD}^{(t)}[i](x) \approx \prod_{j \leq l} \hat{\gamma}_{PD}^{(t)}[s_j](x)$

end for

end for

return $\gamma_{PD}^{(n_T)}$ and $\hat{\gamma}_{PD}^{(n_T)}$.

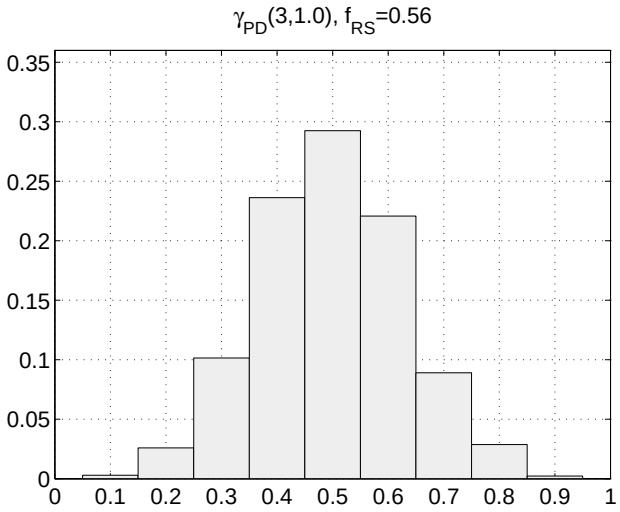

Figure A.3: Distribution γ obtained by Population Dynamics for Random 3-SAT with $r = 1.0$, $n_{PD} = 8000$ and $n_T = 3000$.

used in the previous histograms correspond to a completely bias values of 0 or 1. Indeed, any of those values would make Population Dynamics fail. A similar phenomenon occurs for random 2-SAT as shown in figures A.6 and A.7. However, observe that the mass around biased values for random 3-SAT is considerably bigger that for 2-SAT. This suggest that if we increase the density then the fraction of biased values will change from 0 to a positive value disdcontinually for random 3-SAT and continuously for random 2-SAT. This has been proved rigorously for random 2-SAT [13] and there is substantial experimental and non-formal evidence for random 3-SAT [12, 35, 40].

In the next section we show that any set of empirical distributions can be used to give rigorous bounds for the free entropy density. The hope is that Population Dynamics will provide distributions that will give "good" bounds.

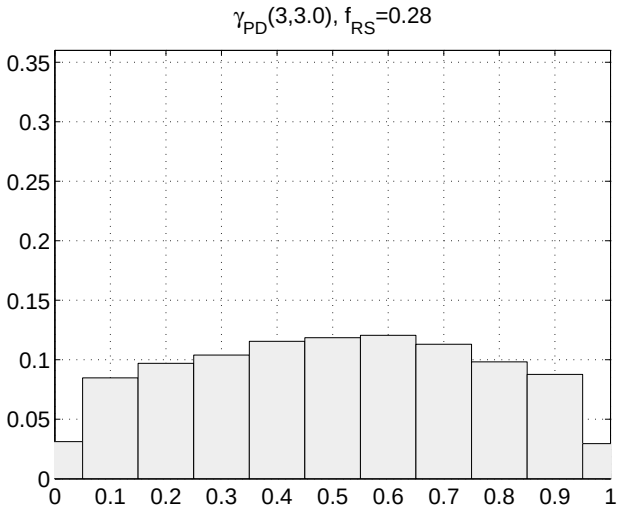

Figure A.4: Distribution γ obtained by Population Dynamics for Random 3-SAT with $r = 2.0$, $n_{PD} = 8000$ and $n_T = 3000$.

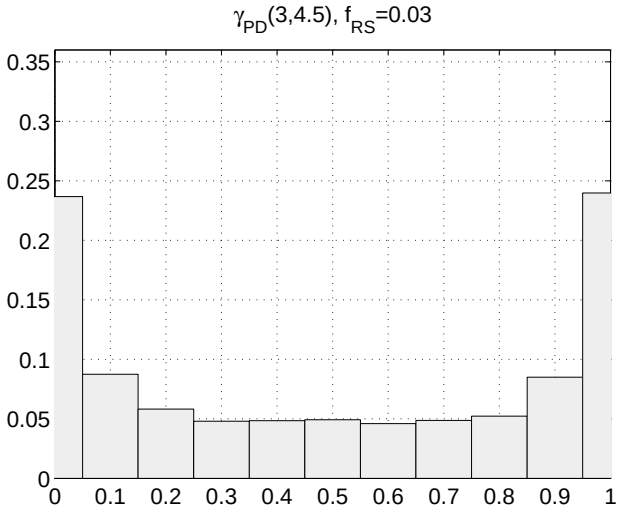

Figure A.5: Distribution γ obtained by Population Dynamics for Random 3-SAT with $r = 4.5$, $n_{PD} = 8000$ and $n_T = 3000$.

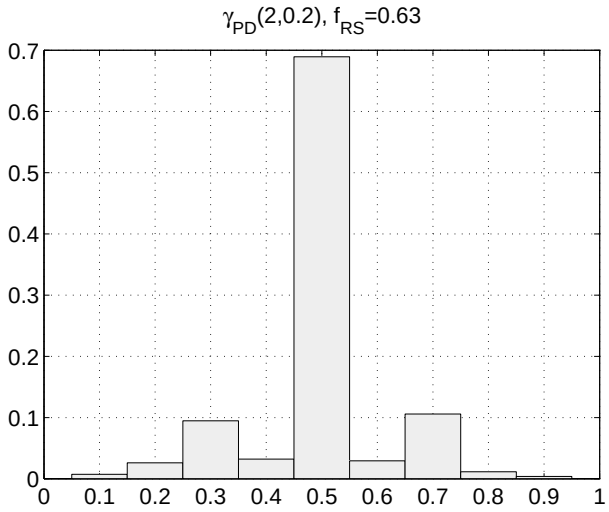

Figure A.6: Distribution γ obtained by Population Dynamics for Random 2-SAT with $r = 0.2$, $n_{PD} = 8000$ and $n_T = 3000$.

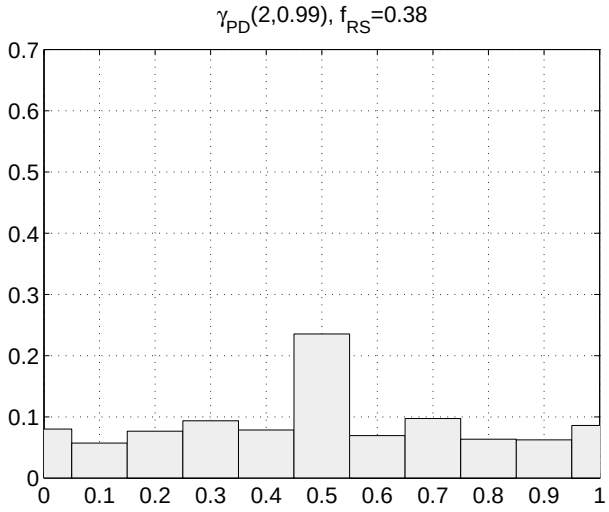

Figure A.7: Distribution γ obtained by Population Dynamics for Random 2-SAT with $r = 0.99$, $n_{PD} = 8000$ and $n_T = 3000$.

Appendix B

Population Dynamics for random k-SAT

The first section contains the implementation details of Population Dynamics for random k-SAT in the context of the entropic interpolation method.

Experimental Results for Random k-SAT

Below, we repeat the pseudocode of Population Dynamics for general random CSPs–with constraints of size k– we introduced in section A.3.1. Note that only line 5 is problem specific: in the case of random k-SAT, drawing a random constraint function $\psi(\cdot)$ corresponds to picking the sign of the k literals in the clause, thus

$$\begin{aligned} \psi(x_a) &= \exp\left(\beta u_a(x_a)\right) \\ &= 1 - (1 - e^{-\beta}) u_a(x_a) \\ &= 1 - (1 - e^{-\beta}) \prod_{j=1}^{k} \frac{1 + J_{a,j} x_{a,j}}{2} \ , \end{aligned} \tag{B.1}$$

where the $J_{a,j}$ are binary random variables with uniform distribution on $\{\pm 1\}$.

Algorithm 5 Population Dynamics

Input: n_{PD}, n_T,

Initialize $\gamma_{PD}(0)$

for $t = 1, ..., n_T$ **do**

  for $i = 1, ..., n_{PD}$ **do**

    Draw $s_1, ..., s_{k-1}$ uniformly in $\{1, ..., n_{PD}\}$

    Draw a random contraint-function $\psi(\cdot)$

    Set $\hat{\gamma}_{PD}^{(t)}[i](x) \approx \sum_{y_1,...,y_{k-1}} \psi(y_1, ..., y_{k-1}, x) \prod_{j \leq k-1} \gamma_{PD}^{(t-1)}[s_j](y_j)$

  end for

  for $i = 1, ..., n_{PD}$ **do**

    Draw a random sample l from Poi(rk)

    Draw $s_1, ..., s_l$ uniformly in $\{1, ..., n_{PD}\}$

    Set $\gamma_{PD}^{(t)}[i](x) \approx \prod_{j \leq l} \hat{\gamma}_{PD}^{(t)}[s_j](x)$

  end for

end for

return $\gamma_{PD}^{(n_T)}$ and $\hat{\gamma}_{PD}^{(n_T)}$.

On the other hand, it is possible to optimize the code by taking advantage of the specifics of the problem. In the general algorithm, every element of the population–a sample message passing trough a random edge– is a vector of dimension equal to the cardinality of the domain D of the variables minus one since the vectors correspond to discrete probability densities i.e. the values have to sum to 1, so in the case of random k-SAT we need to store only one value. We could choose to store either the value corresponding to the input +1 or -1, however, as we will see, it is more convenient to choose an alternative parametrization in which the value of $\gamma_{PD}^{(t)}[i]$ corresponds to the value of x that unsatisfies the clause associated with the message.

With this parameterization, step 6 of the algorithm can be computed as follows: first observe that when x takes the value that satisfies the random clause, then Eq. B.1 evaluates always to 1 and therefore the whole summation evaluates to 1 as well, since the $\gamma_{PD}^{(t-1)}[s]$ are all probability densities; and when x unsatisfies the clause then Eq. B.1 takes the value 1 for all values of the auxiliary variables with the exception of the case where all the auxiliary variables y_j unsatisfy the clause giving $e^{-\beta}$ as result, thus the whole summation can be written as 1 minus the correction term $\prod_{j\leq k-1}\gamma_{PD}^{(t-1)}[s_j]$ plus the correct value $e^{-\beta}\prod_{j\leq k-1}\gamma_{PD}^{(t-1)}[s_j]$. Finally the normalized value of $\hat{\gamma}_{PD}[i]$ is

$$\hat{\gamma}_{PD}[i] = \frac{1-(1-e^{-\beta})\prod_{j\leq k-1}\gamma_{PD}^{(t-1)}[s_j]}{2-(1-e^{-\beta})\prod_{j\leq k-1}\gamma_{PD}^{(t-1)}[s_j]} \ .$$

Note that this reduces the complexity of the computation in step 6 from $\Theta(k2^{k-1})$ to $\Theta(k)$.

For the computation of step 11, recall that the number of messages coming to

a random edge (x, c) in random k-SAT is given by a Poisson with mean rk. Each of those messages come from a clause in which x can appear with the same sign as the one in witch it appears in c with probability $1/2$. In other words, $l_u \sim \text{Poi}(rk/2)$ messages in the computation in 11 come from clauses that don't agree with c (with respect to x) and $l_s \sim \text{Poi}(rk/2)$ messages come from clauses that do agree with c. Thus,

$$\gamma_{PD}^{(t)}[i] = \frac{\prod_{j=1}^{l_u} \hat{\gamma}_{PD}^{(t)}[s_j] \cdot \prod_{j=1}^{l_s}(1 - \hat{\gamma}_{PD}^{(t)}[s_j])}{\prod_{j=1}^{l_u} \hat{\gamma}_{PD}^{(t)}[s_j] \cdot \prod_{j=1}^{l_s}(1 - \hat{\gamma}_{PD}^{(t)}[s_j]) + \prod_{j=1}^{l_s} \hat{\gamma}_{PD}^{(t)}[s_j] \cdot \prod_{j=1}^{l_u}(1 - \hat{\gamma}_{PD}^{(t)}[s_j])} \ ,$$

which also takes $\Theta(k)$ to compute.

Appendix C

The RS Replica Method

The RS replica method is another technique from statistical physics used to approximate the free entropy density. In this section we give a brief overview of the replica method with the objective of acquire a general idea of the method and can be skipped. We do not give the complete derivation of the free entropy density of any RCSP, rather in the next section we state the expression (without derivation) of the free entropy density approximation for random k- and $(2+p)$-SAT using this approach and compare it with the one obtained using the cavity method. Then we proceed to explain how Biroli et. al. [12] manipulated the expression to give a bound for the tri-critical point of $(2+p)$-SAT. The goal is either to mimic their approach using the expression obtained using the cavity method or to establish a inequality relationship between the two expressions.

The replica method is based on the idea that computing the expectation of the j-th integer moment of a complex random variable is easier than computing the

expectation of other functions of the random variable like its logarithm. Thus, the method uses equations that relate the expected value of the j-th moment to the expected value of the function that needs to be computed, for example, the following relation

$$\mathbb{E}[\log Z] = \lim_{j \to 0} \frac{\mathbb{E}[Z^j] - 1}{j}$$

is used to compute the expected value of the free entropy. Notice that the relation require us to treat n as real variable in order to take the limit. This is common of the method and it is one of the things that makes it non rigorous. Another relation frequently used to compute the expected value of the free entropy is

$$\mathbb{E}[\log Z] = \lim_{j \to 0} \frac{1}{j} \log(\mathbb{E}[Z^n]) \ .$$

C.1 Computing the j-th Moment of the Partition Function

The computation of the j-th moment will depend, of course, on the particular system under study, however there are some aspects of the computation that can be generalized. First note that Z^j can be written as an j-fold sum

$$\begin{aligned} Z^j &= \left(\sum_{\sigma} \exp(-\beta H(\sigma|I)) \right)^j \\ &= \sum_{\sigma_1} \cdots \sum_{\sigma_j} \exp\left(-\beta \sum_{i=1}^{j} H_I(\sigma_i) \right) \ . \end{aligned}$$

The above expression is interpreted as the partition function of a new system, where a configuration now consist of j assignments $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_j)$ with $\sigma_i \in D^n$ and energy

function $\boldsymbol{H}(\sigma) = \sum_{i=1}^{j} H_I(\sigma_i)$. Observe that this new system consist of j independent copies of the original one. Each of those copies is known as *replica*.

By using the independence between the constraints in $H_I(\cdot)$, the expected value of Z^j is

$$\begin{aligned}
\mathbb{E}\left[Z^j\right] &= \sum_{\sigma_1}\cdots\sum_{\sigma_j}\mathbb{E}\left[\exp\left(-\beta\sum_{i=1}^{j}H_I(\sigma_i)\right)\right] \\
&= \sum_{\sigma_1}\cdots\sum_{\sigma_j}\mathbb{E}\left[\exp\left(-\beta\sum_{t=1}^{m}\sum_{i=1}^{j}\theta_t(\sigma_i)\right)\right] \\
&= \sum_{\sigma_1}\cdots\sum_{\sigma_j}\left(\mathbb{E}\left[\exp\left(-\beta\sum_{i=1}^{j}\theta(\sigma_i)\right)\right]\right)^m \quad . \qquad \text{(C.1)}
\end{aligned}$$

The computation of the expectation in the above equation depends, for many RCSPs including random k-SAT, on the vector of replicas $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_j)$ only through the overlap function $Q : 2^{[j]} \to \{1, \ldots, n\}$ that counts the number of variables in which a subset of the replicas agrees. Thus, we can rewrite the j-fold sum in terms of all the possible overlap functions Q.

$$\mathbb{E}[Z^j] = \sum_{Q}\mathcal{N}_n(Q)\exp(rn\cdot f(Q,\beta)) \ ,$$

where $\mathcal{N}_N(Q)$ denotes the number of configurations with overlap function Q and $\exp(f(Q,\beta))$ denotes the expectation inside the parenthesis of Eq (C.1) for a given overlap function Q. The number of valid overlap functions is at most n^{2^j} whereas the number of possible number of configurations for the replicas is 2^{nj}, thus we expect the function $\mathcal{N}_n(Q)$ to satisfy a large deviation principle: $\mathcal{N}_n(Q) \doteq \exp(n\cdot s(Q))$. With these considerations,

the j-th moments is written as

$$\mathbb{E}[Z^j] = \sum_Q \exp(n(r \cdot f(Q, \beta) + s(Q))) \ .$$

The summation thus depends on an energy term $r \cdot f(Q, \beta)$ and an entropic term $s(Q)$ and the parameter β determines which one of them dominates: at high temperatures the entropic term dominates whereas at low temperatures the energetic term does. In the large limit $n \to \infty$ the previous summation is computed using the saddle point method i.e. we have to look for the overlap functions that maximize $g(Q|\beta) = r \cdot f(Q, \beta) + s(Q)$.

C.2 Maximizing $g(Q|\beta)$

The maximization (minimization) of $g(Q|\beta)$ is not an easy task, and physicists have used physical insights in order to come up with reasonable saddle points. First it is important to highlight that the function $g(Q|\beta)$ is symmetric under permutation of replicas i.e. $g(Q|\beta) = g(\pi(Q)|\beta)$, where $\pi(Q)$ denotes the overlap function Q resulting from a permutation $\pi(\cdot)$ of the replicas. This is due to the fact that the replicas are indistinguishable from each other, thus $g(Q|\beta)$ should not depend on the details of any specific replica.

The search for saddle points has been organized through families of overlap functions that satisfy different levels of symmetry. The first level consist on the family of overlap functions that are symmetric $Q = \pi(Q)$. If the saddle points are in this family then it is said that the system is replica symmetric (RS). The next level known as 1-step replica symmetry breaking (1RSB) divides the replicas into s different symmetric

groups, meaning that the overlap function corresponding to each group is symmetric. This argument can be generalized by subdividing the groups into subgroups (2RSB) and so on.

C.3 Computing the Free Entropy Density

Once a expression for $\mathbb{E}[Z^j]$ has been obtained. One treats j as an integer and takes limits in the definition of the free entropy density:

$$f \equiv \lim_{n\to\infty} \frac{1}{n}\mathbb{E}[\log Z] = \lim_{n\to\infty}\lim_{j\to 0} \frac{1}{jn}\log(\mathbb{E}[Z^j]) \ .$$

Notice that the $n \to \infty$ limit is taken after taking the $j \to 0$ limit. However, in our calculation of the j-th moment we were implicitly assuming that we were in the $n \to \infty$ limit: we assumed a large deviation principle for $\mathcal{N}_n(Q)$ and we used the saddle point method. The interchange of the limits is not general and needs a mathematical proof for every particular system, unfortunately for most models there is no such proof and we have to include this to the set of steps that make the method non-rigorous.

C.3.1 The RS Replica Method for random k- and $(2+p)$ - SAT.

In section (C.4) we state the RS cavity method and the RS replica method expression of the free entropy density for random k-SAT. The goal is to show the main differences among them. It is believed that both methods are equivalent in the sense that both predict the same location of the phase transitions for random k-SAT and other RCSP, however, as far as we know, there is no rigorous proof of such statement.

In section (C.5) we give an overview of the analysis made by Biroli et.al. [12] to show that the tri-critical point for $(2+p)$-SAT is upper bounded by $2/3$ by assuming that the RS replica method is correct.

C.4 RS Replica Method vs RS Cavity Method

The RS replica method approximation for the free entropy density for random k-SAT is given by the following expression[1] [12]:

$$\begin{aligned} f_{RS_{rm}}(r|\gamma_{rm}) &= \int_{-\infty}^{\infty} \frac{dhdv}{2\pi} e^{\mathrm{i}vh} \cdot \Gamma_{ft}(v) \cdot [1 - \log \Gamma_{ft}(v)] \log[2\cosh(\beta h)] \\ &\quad + r \int_{-\infty}^{\infty} \prod_{l=1}^{k} dh_l \gamma_{rm}(h_l) \cdot \log \left[1 + (e^{-\beta} - 1) \prod_{l=1}^{k} \left(\frac{e^{-\beta h_l}}{2\cosh(\beta h_l)} \right) \right] \end{aligned} \quad \text{(C.2)}$$

where $\gamma_{rm}(\cdot)$ is a probability density function over $\mathbb{R}$ and $\Gamma_{ft}(v) = \int_{-\infty}^{\infty} dx \exp(-ixv)\gamma_{rm}(x)$ denotes the Fourier transform of $\gamma_{rm}(\cdot)$. Because equation (C.2) was obtained through the saddle point method, the correct density $\gamma_{rm}(\cdot)$ is the one that minimizes $f_{RS_{rm}}(r|\gamma_{rm})$ (i.e. the one that maximizes the free energy density).

On the other hand, the RS cavity computation for the free entropy density for random k-SAT derived using the methodology presented in Appendix A is given as follows: Let $v(x)$ denote the density function of a random variable over $\{+1, -1\}$, where the probability $\Pr[x = +1] = p_{+1}$ is itself chosen at random from a distribution $\gamma_{cm}(\cdot)$ with support on $[0, 1]$, and let

[1] The expression in Biroli's paper is the free energy density. To get the free entropy density we just have to multiply by $-\beta$.

$$f_r(0) = \mathbb{E}\left[\log\left(\sum_{x\in\{+1,-1\}}\prod_{j\leq s}\hat{v}_j(x)\right)\right] \ ,$$

where s is a poisson random variables with mean kr, and the $\hat{v}_j(x)$ are i.i.d random univariate function defined as

$$\hat{v}(x) = \sum_{y_1,\ldots,y_{k-1}} \exp\left(-\beta\theta(y_1,\ldots,y_{k-1},x)\right)\prod_{j=1}^{k-1} v_j(y_j) \ , \tag{C.3}$$

where $\theta(\cdot)$ is a random constraint-function and the functions $v_i(\cdot)$ are i.i.d. with the same distribution as $v(x)$. In the case of random k-SAT

$$\exp\left(-\beta\theta(x_1,\ldots,x_k)\right) = 1 + (e^{-\beta}-1)\cdot\prod_{j=1}^{k}\frac{1+J_jx_j}{2} \ ,$$

where the J's are uniform random variables over $\{+1,-1\}$. That is, the variables J determine the random signs of the literals in a random clause. Thus $\exp\left(-\beta\theta(x_1,\ldots,x_k)\right) = \exp(-\beta)$ if the clause is unsatisfied and $\exp\left(-\beta\theta(x_1,\ldots,x_k)\right) = 1$ otherwise. Using the previous definitions, the RS cavity expression is

$$f_{RS_{cm}}(r|\gamma_{cm}) = f_r(0) - r(k-1)\mathbb{E}\left[\log\left(v_c\right)\right] \ ,$$

where

$$v_c = \sum_{y_1,\ldots,y_k}\left[1+(e^{-\beta}-1)\cdot\prod_{i=1}^{k}\frac{1+J_iy_i}{2}\right]\prod_{j=1}^{k}v_j(y_j) \ .$$

The first thing to notice is that $\gamma_{cm}(\cdot)$ is a distribution over $[0,1]$ whereas $\gamma_{rm}(\cdot)$ is a distribution over the reals. Fortunately, this is just a difference in notation. It is possible to write the function $v(\cdot)$ in terms of a density over the reals as follows

$$v(x|h) = \frac{\exp(\beta x h)}{\exp(\beta h) + \exp(-\beta h)} = \frac{\exp(\beta x h)}{2\cosh(\beta h)} \ ,$$

where h is a random variable over $\mathbb{R}$ with distribution $\gamma(\cdot)$. The expression for $\mathbb{E}\left[\log\left(v_c\right)\right]$ using this parametrization is

$$\mathbb{E}\left[\log\left(v_c\right)\right] = \mathbb{E}\left[\log\left(\sum_{y_1,\ldots,y_k}\left[1 + (e^{-\beta} - 1)\cdot\prod_{i=1}^{k}\frac{1+J_i y_i}{2}\right]\prod_{i=1}^{k}\frac{\exp(\beta y_i h_i)}{2\cosh(\beta h_i)}\right)\right] \ .$$

Any of the 2^k realization of the J random variables occurs with probability 2^{-k} and for all of them

$$\mathbb{E}\left[\sum_{y_1,\ldots,y_k}\left[1 + (e^{-\beta} - 1)\cdot\prod_{i=1}^{k}\frac{1+J_i y_i}{2}\right]\prod_{i=1}^{k}\frac{\exp(\beta y_i h_i)}{2\cosh(\beta h_i)}\right] \ = \ \mathbb{E}\left[1 + (e^{-\beta} - 1)\prod_{i=1}^{k}\frac{\exp(\beta h_i)}{2\cosh(\beta h_i)}\right] \ ,$$

where we use the fact that $\exp(\beta x h)/2\cosh(\beta h)$ is a density and the product $\prod_{i=1}^{k}(1 + J_i y_i)/2$ is 1 for a single combination of the y's and 0 otherwise. Thus,

$$\mathbb{E}\left[\log\left(v_c\right)\right] = \mathbb{E}\left[\log\left(1 + (e^{-\beta} - 1)\prod_{i=1}^{k}\frac{\exp(\beta h_i)}{2\cosh(\beta h_i)}\right)\right] \ ,$$

which corresponds to the expression appearing in the second term of $f_{RS_{rm}}(r|\gamma_{rm})$ after the expectation with respect to the h's random variables has been written explicitly.

The term $f_r(0)$ is more problematic. The $\hat{v}(\cdot)$ functions expressed in terms of the new parametrization have the following form

$$\hat{v}(x) = \sum_{y_1,\ldots,y_{k-1}}\left[1 + (e^{-\beta} - 1)\cdot\frac{1+Jx}{2}\prod_{i=1}^{k-1}\frac{1+J_i y_i}{2}\right]\prod_{i=1}^{k-1}\frac{\exp(\beta y_i h_i)}{2\cosh(\beta h_i)} \ .$$

Again, for any realization of the J_i random variables we have

$$\hat{v}(x) = 1 + (e^{-\beta} - 1) \cdot \frac{1 + Jx}{2} \prod_{i=1}^{k-1} \frac{\exp(\beta h_i)}{2\cosh(\beta h_i)} \quad .$$

Therefore,

$$\begin{aligned} f_r(0) &= \mathbb{E}\left[\log\left(\sum_{x\in\{+1,-1\}} \prod_{j\leq s} \hat{v}_j(x)\right)\right] \\ &= \mathbb{E}\left[\log\left(\sum_{x\in\{+1,-1\}} \prod_{j\leq s} \left\{1 + (e^{-\beta} - 1) \cdot \frac{1 + J_j x}{2} \prod_{i=1}^{k-1} \frac{\exp(\beta h_{i,j})}{2\cosh(\beta h_{i,j})}\right\}\right)\right] \quad . \end{aligned}$$

It does not seem like the above expression can be transformed into something similar to the first term of $f_{RS_{rm}}(r|\gamma_{rm})$. Perhaps taking the expectation with respect to the poisson random variable s could give rise to the Fourier transform terms $\Gamma_{ft}(v)$ after the correct algebraic manipulations, unfortunately we were not able to find such transformation.

Also, note that we are taking the expectations with respect with the randomness of the model at the very end of the cavity computation, whereas this step was done at very early stages of the replica method.

C.5 RS Replica Method Computation of the Tri-Critical Point of $(2+p)$- SAT

In [12] the authors showed how to compute the tri-critical point of $(2+p)$-SAT assuming that equation (C.2) gives upper bounds on the free entropy density. In the next paragraphs we describe such computation.

The first step consist in parametrizing the density $\gamma(\cdot)$ as follows:

$$\gamma(h) = (1-B)\delta(h) + \frac{B}{\sqrt{\Delta}}\Phi(\frac{h}{\sqrt{\Delta}}) \ ,$$

where $B \in [0,1]$ represents the fraction of frozen variables in the model andΔ is the typical squared scale of (the field) h acting on frozen variables at $\beta \to \infty$. The function $\delta(\cdot)$ is the Dirac function that accounts for the fraction of free variables and Φ is an even probability distribution. After taking the$\beta \to \infty$ limit and using the above parametrization, the functional $f_{RS_{rm}}(\cdot)$ for $(2+p)$-SAT has the following form:

$$\begin{aligned} f_{RS_{rm}}(r,p,B,\Delta|\Phi) &= 2\sqrt{\triangle}(\frac{B}{\pi}\int_0^\infty \frac{dv}{v}\Phi'_{ft}(v)\ln[1-B+B\Phi_{ft}(v)] \\ &\quad +r\int_0^{1/(2\sqrt{\Delta})} dh\{(1-p)B^2[\Phi_{cc}(v)]^2 + pB^3[\Phi_{cc}(v)]^3\}) \end{aligned}$$

where

$$\begin{aligned} \Phi_{ft}(v) &= \int_{-\infty}^{\infty} dx \exp(-ixv)\Phi(x) \\ \Phi_{cc}(h) &= \int_h^\infty dx\Phi(x) \end{aligned}$$

are the Fourier transform and the complementary cumulative function of Φ respectively. Once a distribution Φ is chosen, the goal is to find the values of B andΔ that minimize $f_{RS_{rm}}(\cdot)$ for given values of r and p.

When $p = 0$ the problem reduces to random 2-SAT and the best upper bound for the satisfiability threshold that can be computed using this approach is obtained by solving for the value of r where $f_{RS_{rm}}(r, p=0, B, \Delta)$ changes sign. Note that when $B = 0$, $f_{RS_{rm}}(r, p=0, B, \Delta) = 0$ and since the fraction of frozen variables is continuos

at the transition and is zero for $r < r_2^*$, is possible to identify the transition by expanding the expression around $B = 0$ and $\Delta = 0$ to the leading order and neglecting irrelevant terms in Δ, thus getting

$$f_{RS_{rm}}(r, p, B, \Delta|\Phi) = 2\sqrt{\triangle}(B^2 f_{rs}^{(2)}(r) + B^3 f_{rs}^{(3)}(r)) \ ,$$

where

$$\begin{aligned} f_{rs}^{(2)}(r) &= \frac{1}{\pi}\int_0^\infty \frac{dv}{v}\Phi'_{ft}(v)[\Phi_{ft}(v) - 1] \\ &\quad -r\int_0^\infty dh[\Phi_{cc}(v)]^2 \end{aligned}$$

$$f_{rs}^{(3)}(r) = -\frac{1}{2\pi}\int_0^\infty \frac{dv}{v}\Phi'_{ft}(v)[\Phi_{ft}(v) - 1]^2 \ .$$

Observe that $f_{rs}^{(3)}(r)$ is always positive, thus, the minimum is located at $B = 0$ when $f_{rs}^{(2)}(r) \geq 0$ and at $B > 0$ if $f_{rs}^{(2)}(r) < 0$. Therefore the threshold r_2^* is determined by the condition $f_{rs}^{(2)}(r_2) = 0$. This condition gives $r_2^* = 1$ independently of the distribution Φ used.

When $p = 1$ the problem reduces to random 3-SAT. In the SAT phase $B = 0$, but the transition is not continuos so it is not possible to expand around to $B = 0$. In this case the threshold value is obtained by solving for the value of r where the minimum of $f_{RS_{rm}}(r, p = 1, B, \Delta)$ with $B > 0$, $\Delta > 0$ changes sign.

$$f_{RS_{rm}}(r, p = 1, B, \Delta) = 2\sqrt{\Delta}(B^3(s_{rs}(B) - re_{rs}(\Delta)) \ ,$$

where

$$s_{rs}(B) = \frac{1}{\pi B^2} \int_0^\infty \frac{dv}{v} \Phi'_{ft}(v) \ln[1 - B + B\Phi_{ft}(v)]$$

$$e_{rs}(\Delta) = \int_0^{1/(2\sqrt{\Delta})} dh[\Phi_{cc}(v)]^3 \ .$$

When $B > 0$, $\Delta > 0$, the minimum of $f_{RS_{rm}}(r, p = 1, B, \Delta)$ changes sign at

$$r = \frac{\min_B s_{rs}(B)}{\max_\Delta e_{rs}(\Delta)} = \frac{s_{rs}(B_C)}{e_{rs}(\Delta_C)} \ .$$

To analyze the tri-critical point expand $f_{RS_{rm}}(\cdot)$ in powers of B,

$$f_{RS_{rm}}(r, p, B, \Delta) = 2\sqrt{\Delta}(B^2 f_{rs}^{(2)}(r, p) + B^3 f_{rs}^{(3)}(r, p)) \ ,$$

where,

$$f_{rs}^{(2)}(r, p) = (1 - (1 - p)r) \int_0^\infty dh[\Phi_{cc}(v)]^2$$

$$\begin{aligned} f_{rs}^{(3)}(r, p) &= -\frac{1}{2\pi} \int_0^\infty \frac{dv}{v} \Phi'_{ft}(v)[\Phi_{ft}(v) - 1]^2 \\ &\quad -rp \int_0^\infty dh[\Phi_{cc}(v)]^3 \ . \end{aligned}$$

As long as $f_{rs}^{(3)}$ remains positive, the threshold is at $r_c = 1/(1 - p)$ as in the case of 2-SAT. Therefore, the tri-critical point p_0 can be found through the condition

$$f_{RS}^{(3)}(1/(1 - p_0), p_0) = 0 \ .$$

Note that the value of p_0 depends on the choice of the distribution $\Phi(\cdot)$. Let $\Phi^{(1)}$ and $\Phi^{(2)}$ with $p_0^{(1)} < p_0^{(2)}$. It is possible to show that this implies that the free entropy density using $\Phi^{(1)}$ is smaller than the one obtained using $\Phi^{(2)}$. Thus p_0 has to be minimized over the choice of possible distributions Φ.

Bibliography

[1] R. Abou-Chacra, DJ Thouless, and PW Anderson. A selfconsistent theory of localization. *Journal of Physics C: Solid State Physics*, 6(10):1734, 2002.

[2] D. Achioptas and G.B. Sorkin. Optimal myopic algorithms for random 3-SAT. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 590–600. IEEE, 2000.

[3] D. Achlioptas. Lower bounds for random 3-SAT via differential equations. *Theoretical Computer Science*, 265(1-2):159–185, 2001.

[4] D. Achlioptas, P. Beame, and M. Molloy. A sharp threshold in proof complexity. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 337–346. ACM New York, NY, USA, 2001.

[5] D. Achlioptas, P. Beame, and M. Molloy. A sharp threshold in proof complexity yields lower bounds for satisfiability search. *Journal of Computer and System Sciences*, 68(2):238–268, 2004.

[6] D. Achlioptas, L.M. Kirousis, E. Kranakis, and D. Krizanc. Rigorous results for random (2+ p)-SAT. *Theoretical Computer Science*, 265(1):109–129, 2001.

[7] D. Achlioptas and R. Menchaca-Mendez. Exponential lower bounds for DPLL algorithms on satisfiable random 3-cnf formulas. *Theory and Applications of Satisfiability Testing–SAT 2012*, pages 327–340, 2012.

[8] D. Achlioptas and R. Menchaca-Mendez. Unsatisfiability bounds for random CSPs from an energetic interpolation method. *Automata, Languages, and Programming*, pages 1–12, 2012.

[9] D. Achlioptas and Y. Peres. The threshold for random k-SAT is $2^k log 2 - O(k)$. *JOURNAL-AMERICAN MATHEMATICAL SOCIETY*, 17(4):947–974, 2004.

[10] Dimitris Achlioptas. Setting 2 variables at a time yields a new lower bound for random 3-sat. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 28–37. ACM, 2000.

[11] Mohsen Bayati, David Gamarnik, and Prasad Tetali. Combinatorial approach to the interpolation method and scaling limits in sparse random graphs. In *STOC'10*, pages 105–114, 2010.

[12] G. Biroli, R. Monasson, and M. Weigt. A variational description of the ground state structure in random satisfiability problems. *The European Physical Journal B-Condensed Matter and Complex Systems*, 14(3):551–568, 2000.

[13] B. Bollobás, C. Borgs, J.T. Chayes, J.H. Kim, and D.B. Wilson. The scaling

window of the 2-SAT transition. *Random Structures & Algorithms*, 18(3):201–256, 2001.

[14] M.T. Chao and J. Franco. Probabilistic analysis of two heuristics for the 3-satisfiability problem. *SIAM Journal on Computing*, 15(4):1106–1118, 1986.

[15] V. Chvátal and B. Reed. Mick gets some (the odds are on his side)[satisfiability]. In *Foundations of Computer Science, 1992. Proceedings., 33rd Annual Symposium on*, pages 620–627. IEEE, 1992.

[16] V. Chvatal and E. Szemeredi. Many hard examples for resolution. *Journal of the Association for Computing Machinery*, 35(4):759–768, 1988.

[17] S.A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[18] R. Courant and D. Hilbert. *The Calculus of Variations*. Wiley Online Library, 2007.

[19] J. Díaz, L. Kirousis, D. Mitsche, and X. Pérez-Giménez. On the satisfiability threshold of formulas with three literals per clause. *Theoretical Computer Science*, 410(30-32):2920–2934, 2009.

[20] R. Diestel. Graph theory. 2005. *Grad. Texts in Math*, 2005.

[21] O. Dubois and Y. Boufkhad. A general upper bound for the satisfiability threshold of random r-SAT formulae. *Journal of algorithms*, 24(2):395–420, 1997.

[22] W. Fernandez de la Vega. Random 2-SAT: results and problems. *Theoretical computer science*, 265(1):131–146, 2001.

[23] A. Flaxman. Algorithms for random 3-SAT. *Encyclopedia of Algorithms*, pages 742–744, 2008.

[24] S. Franz and M. Leone. Replica bounds for optimization problems and diluted spin systems. *Journal of Statistical Physics*, 111(3):535–564, 2003.

[25] S. Franz, M. Leone, and F.L. Toninelli. Replica bounds for diluted non-poissonian spin systems. *Journal of Physics A: Mathematical and General*, 36:10967, 2003.

[26] Ehud Friedgut. Sharp thresholds of graph properties, and the k-sat problem. *J. Amer. Math. Soc*, 12:1017–1054, 1998.

[27] A. Frieze and S. Suen. Analysis of Two Simple Heuristics on a Random Instance of k-SAT. *Journal of Algorithms*, 20(2):312–355, 1996.

[28] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. *Markov chain Monte Carlo in practice*. Chapman & Hall/CRC, 1996.

[29] A. Goerdt. A threshold for unsatisfiability. *Journal of Computer and System Sciences*, 53(3):469–486, 1996.

[30] R.L. Graham, D.E. Knuth, and O. Patashnik. *Concrete mathematics: a foundation for computer science*, volume 2. Addison-Wesley Reading, MA, 1994.

[31] F. Guerra and F.L. Toninelli. The thermodynamic limit in mean field spin glass models. *Communications in Mathematical Physics*, 230(1):71–79, 2002.

[32] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–562, 2006.

[33] A.C. Kaporis, L.M. Kirousis, and Y.C. Stamatiou. How to prove conditional randomness using the principle of deferred decisions. In *Special Volume on Computational Complexity and Statistical Physics. Santa Fe Institute, Studies in*, 2002.

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

[35] F. Krzakala, A. Montanari, F. Ricci-Tersenghi, G. Semerjian, and L. Zdeborova. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences*, 104(25):10318, 2007.

[36] F.R. Kschischang, B.J. Frey, and H.A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

[37] L.A. Levin. Universal sequential search problems. *Problemy Peredachi Informatsii*, 9(3):115–116, 1973.

[38] M Virasoro Marc Mézard, G Parisi. *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications*. World Scientific, 1986.

[39] Colin McDiarmid. Concentration, probabilistic methods for algorithmic discrete mathematics, 195–248. *Algorithms Combin*, 16, 1998.

[40] S. Mertens, M. Mézard, and R. Zecchina. Threshold values of random k-SAT from the cavity method. *Random Structures & Algorithms*, 28(3):340–373, 2006.

[41] M. Mézard and A. Montanari. Information, Physics and Computation, 2008.

[42] M. Mezard and A. Montanari. *Information, physics, and computation.* Oxford University Press, USA, 2009.

[43] R. Monasson and R. Zecchina. Tricritical points in random combinatorics: the SAT case. *Journal of Physics A: Mathematical and General*, 31:9209, 1998.

[44] Rémi Monasson and Riccardo Zecchina. Entropy of the K -satisfiability problem. *Phys. Rev. Lett.*, 76:3881–3885, May 1996.

[45] Rémi Monasson and Riccardo Zecchina. Statistical mechanics of the random k-satisfiability model. *Phys. Rev. E*, 56:1357–1370, Aug 1997.

[46] A. Montanari. Tight bounds for ldpc and ldgm codes under map decoding. *Information Theory, IEEE Transactions on*, 51(9):3221 –3246, sept. 2005.

[47] D. Panchenko and M. Talagrand. Bounds for diluted mean-fields spin glass models. *Probability Theory and Related Fields*, 130(3):319–336, 2004.

[48] C.P. Robert and G. Casella. *Monte Carlo statistical methods.* Springer Verlag, 2004.

[49] B. Selman, H. Kautz, B. Cohen, et al. Local search strategies for satisfiability testing. *Cliques, coloring, and satisfiability: Second DIMACS implementation challenge*, 26:521–532, 1993.

[50] B. Van Brunt. *The calculus of variations*, volume 1. Springer, 2004.

[51] M.J. Wainwright. *Graphical Models, Exponential Families, and Variational Inference.* Now Publishers, 2008.

[52] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003.