Survey Propagation: An Algorithm for Satisfiability

A. Braunstein,[1,3] M. Mézard,[2] R. Zecchina[3]
[1] *International School for Advanced Studies (SISSA), via Beirut 9, 34100 Trieste, Italy*
[2] *Laboratoire de Physique Théorique et Modèles Statistiques, CNRS and Université Paris Sud, Bâtiment 100, 91405 Orsay Cedex, France*
[3] *The Abdus Salam International Centre for Theoretical Physics (ICTP), Strada Costiera 11, 34100 Trieste, Italy; e-mail: zecchina@ictp.trieste.it*

Received 20 January 2003; revised 19 August 2004; accepted 14 October 2004
Published online 4 March 2005 in Wiley InterScience (www.interscience.wiley.com).
DOI 10.1002/rsa.20057

ABSTRACT: We study the satisfiability of randomly generated formulas formed by M clauses of exactly K literals over N Boolean variables. For a given value of N the problem is known to be most difficult when $\alpha = M/N$ is close to the experimental threshold α_c separating the region where almost all formulas are SAT from the region where all formulas are UNSAT. Recent results from a statistical physics analysis suggest that the difficulty is related to the existence of a clustering phenomenon of the solutions when α is close to (but smaller than) α_c. We introduce a new type of message passing algorithm which allows to find efficiently a satisfying assignment of the variables in this difficult region. This algorithm is iterative and composed of two main parts. The first is a message-passing procedure which generalizes the usual methods like Sum-Product or Belief Propagation: It passes messages that may be thought of as surveys over clusters of the ordinary messages. The second part uses the detailed probabilistic information obtained from the surveys in order to fix variables and simplify the problem. Eventually, the simplified problem that remains is solved by a conventional heuristic. Random Struct. Alg., 27, 201–226, 2005

1. INTRODUCTION

The satisfiability problem is the archetype of combinatorial optimization problems which are well known to be intractable in the worst case. However, experimental studies show that

Correspondence to: R. Zecchina

many instances of satisfiability are surprisingly easy, even for naive heuristic algorithms. In an attempt to get a better understanding of which instances are easy or hard to solve, a lot of efforts have focused in recent years on the "random K-sat" problem [1]. Instances of this problem are generated by considering N variables and $M = \alpha N$ clauses, where each clause contains exactly K distinct variables, and is picked up with uniform probability distribution from the set of $\binom{N}{K}2^K$ possible clauses. For a given value of α, the probability $P_N(\alpha)$ that a randomly generated instance is SAT is a decreasing function, with $P_N(0) = 1$ and $\lim_{\alpha\to\infty} P_N(\alpha = 0)$, which has been shown to approach, as N increases, a step function characteristic of a zero-one law [12], or a "phase transition." It is convenient to identify a crossover regime between "SAT" and "UNSAT" regimes using the value $\alpha_c(N)$ of the number of constraints per variable where $P_N(\alpha_c(N)) = 1/2$. From numerical simulations, $\alpha_c(N)$ is supposed to converge, in the large N limit, to a value around $\alpha_c \simeq 4.27$ [8, 10, 16, 26], but this convergence has not yet been established rigorously. Interestingly, the performance of algorithms is found to be much worse around this value of $\alpha = 4.27$: randomly generated instances with α near to the phase transition are particularly difficult to solve.

Rigorous lower and upper bounds have been found for this conjectured satisfiability threshold: it has been established that $\lim_{N\to\infty} P_N(\alpha) = 1$ for $\alpha < \alpha_{lb}$ and $\lim_{N\to\infty} P_N(\alpha) = 0$ for $\alpha > \alpha_{ub}$. The present best bounds for the case of the random 3-SAT problem (with $K = 3$) are $\alpha_{ub} = 4.506$ (from [9], using the first moment method) and $\alpha_{lb} = 3.42$ (from [15], using algorithmic analysis). Note also the interesting algorithm-independent upper bound found in [1, 28] using the second moment method, which becomes better for larger values of K.

Recently, some elaborate statistical physics methods have been brought to bear on the random satisfiability problem. These non-rigorous analytical calculations have put forward some interesting conjectures about what happens in the solution space of the problem as this threshold is approached [22, 24] (see also previous work in [2, 25]). They suggest the following overall picture, which should hold for a generic sample of the random satisfiability problem, in the limit $N \to \infty$, with α fixed.

1. There exists a SAT-UNSAT phase transition at a critical value α_c which can be computed by solving some (complicated) integral equation; for $K = 3$, one gets $\alpha_c \simeq 4.267$.
2. There exists a second threshold α_{clust} separating two phases which are both "SAT" (in each of them there exists a satisfying assignment with probability 1), but with very different geometric structures.
3. For $\alpha < \alpha_{clust}$, a generic problem has many solutions, which tend to form one giant "cluster"; the set of all satisfying assignments forms a connected cluster in which it is possible to find a path between two solutions that requires short steps only (each pair of consecutive assignments in the path are close together in Hamming distance). In this regime, local search algorithms and other simple heuristics can relatively easily find a solution. This region is called the "easy-SAT" region.
4. For $\alpha_{clust} < \alpha < \alpha_c$, there exists a "hard SAT" phase where the solution space breaks up into many smaller clusters. Solutions in separate clusters are generally far apart: It is not possible to transform a SAT assignment in one cluster into another one in a different cluster by changing only a finite number of variables. Because of this clustering effect, local search algorithms tend to have a very slow convergence when applied to large N instances.

So far, the analytic method used in the most recent statistical physics analysis, named the cavity method [21], is nonrigorous, and turning this type of approach into a rigorous theory is an open subject of current research [11, 35]. Note, however, that, in the simpler case of the random K-XOR-SAT, the validity of this statistical physics analysis can be confirmed by rigorous studies [6, 23], and the above clustering conjecture has been fully confirmed [3].

Interestingly, the statistical physics analysis suggests a new efficient heuristic algorithm for finding SAT assignments in the hard SAT phase, which has been put forward by two of us in [24].

The aim of this paper is to provide a detailed self-contained description of this algorithm, which does not rely on the statistical physics background. We shall limit the description to the regime where solutions exist, the so-called SAT phase; some modification of the algorithm allows us to address the optimization problem of minimizing the number of violated constraints in the UNSAT phase, but it will not be discussed here.

The basic building block of the algorithm, called survey propagation (SP), is a message passing procedure which resembles in some respect the iterative algorithm known as belief propagation (BP), but with some crucial differences which will be described. BP is a generic algorithm for computing marginal probability distributions in problems defined on factor graphs, which has been very useful in the context of error correcting codes [13] and Bayesian networks [32].

While in simple limits we are able to give some rigorous results together with an explicit comparison with the belief propagation procedures, in general there exists no rigorous proof of convergence of the algorithm. However, we provide clear numerical evidence of its performance over benchmarks problems which appear to be far larger than those which can be handled by present state-of-the-art algorithms.

The paper is organized as follows: Section 2 describes the satisfiability problem and its graphical representation in terms of a factor graph. Section 3 explains two message passing algorithms, namely warning propagation (WP) and belief propagation (BP). Both are exact for tree factor graphs. Even if they are typically unable to find a solution for random SAT in the "interesting" hard-SAT region, they are shown here because they are in some sense the basic building blocks of our survey propagation algorithm. Section 4 explains the survey propagation algorithm itself, a decimation procedure based on it, the "survey inspired decimation" (SID), and the numerical results. In Section 5 we give some heuristic arguments from statistical physics which may help the reader to understand where the SP algorithm comes from. Section 6 contains a few general comments.

2. THE SAT PROBLEM AND ITS FACTOR GRAPH REPRESENTATION

We consider a satisfiability problem consisting of N Boolean variables $\{x_i \in \{0,1\}\}$ (where $\{0,1\} \equiv \{F,T\}$), with $i \in \{1,\dots,N\}$, with M constraints. Each constraint is a clause, which is the logical OR of the variables or of their negations. A clause a is characterized by the set of variables $i_1,\dots,i_K$ which it contains, and the list of those which are negated, which can be characterized by a set of K numbers $J^a_{i_r} \in \{\pm 1\}$ as follows. The clause is written as

$$\left(z_{i_1} \vee \cdots \vee z_{i_r} \vee \cdots \vee z_{i_K}\right), \tag{1}$$

where $z_{i_r} = x_{i_r}$ if $J^a_{i_r} = -1$ and $z_{i_r} = \bar{x}_{i_r}$ if $J^a_{i_r} = 1$ (note that a positive literal is represented by a negative J). The problem is to find whether there exists an assignment of the $x_i \in \{0,1\}$

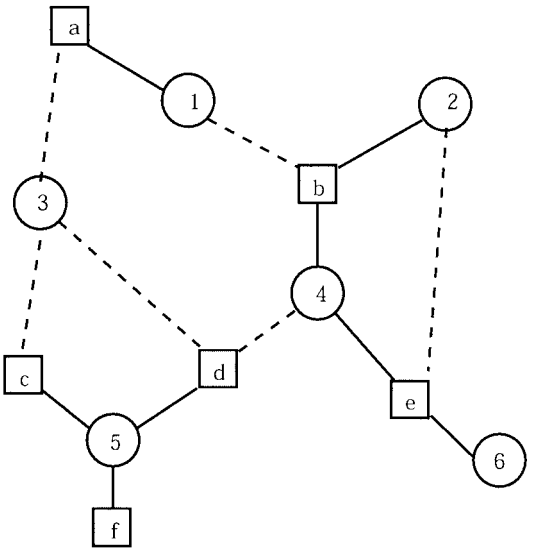

Fig. 1. An example of a factor graph with 6 variable nodes $i = 1, \ldots, 6$ and 6 function nodes a, b, c, d, e, f. The formula which is encoded is: $F = (x_1 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (\bar{x}_3 \vee x_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee x_5) \wedge (\bar{x}_2 \vee x_4 \vee x_6) \wedge (x_5)$.

which is such that all the M clauses are true. We define the total cost C of a configuration $\mathbf{x} = (x_1, \ldots, x_N)$ as the number of violated clauses.

In what follows we shall adopt the factor graph representation [18] of the SAT problem. This representation is convenient because it provides an easy graphical description to the message passing procedures which we shall develop. It also applies to a wide variety of different combinatorial problems, thereby providing a unified notation.

The SAT problem can be represented graphically as follows (see Fig. 1). Each of the N variables is associated with a vertex in the graph, called a "variable node" (circles in the graphical representation), and each of the M clauses is associated with another type of vertex in the graph, called a "function node" (squares in the graphical representation). A function node a is connected to a variable node i by an edge whenever the variable x_i (or its negation) appears in the clause a. In the graphical representation, we use a full line between a and i whenever the variable appearing in the clause is x_i (i.e., $J_i^a = -1$), a dashed line whenever the variable appearing in the clause is $\bar{x}_i$ (i.e., $J_i^a = 1$). Variable nodes compose the set X ($|X| = N$) and function nodes the set A ($|A| = M$).

In summary, each SAT problem can be described by a bipartite graph, $G = (X \cup A; E = X \times A)$, where E is the edge set, and by the set of "couplings" $\{J_a^i\}$ needed to define each function node. For the K-SAT problem where each clause contains K variables, the degree of all the function nodes is K.

Throughout this paper, the variable nodes indices are taken in $i, j, k, \ldots$, while the function nodes indices are taken in $a, b, c, \ldots$. For every variable node i, we denote by $V(i)$ the set of function nodes a to which it is connected by an edge, by $n_i = |V(i)|$ the degree of the node, by $V_+(i)$ the subset of $V(i)$ consisting of function nodes a where the variable appears un-negated (the edge (a, i) is a full line), and by $V_-(i)$ the complementary subset of $V(i)$ consisting of function nodes a where the variable appears negated (the edge (a, i) is a dashed line). $V(i) \backslash b$ denotes the set $V(i)$ without a node b. Similarly, for each function node a,

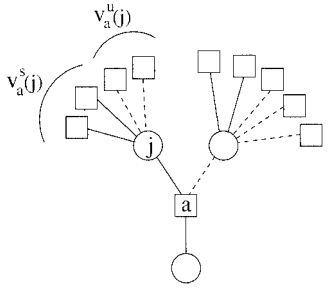

Fig. 2. A function node a of the factor graph with the $V_a^u(j)$ and $V_a^s(j)$ sets relative to node j.

we denote by $V(a) = V_+(a) \cup V_-(a)$ the set of neighboring variable nodes, decomposed according to the type of edge connecting a and i, and by n_a the degree. Given a function node a and a variable node j, connected by an edge, it is also convenient to define the two sets: $V_a^u(j)$ and $V_a^s(j)$, where the indices s and u respectively refer to the neighbors which tend to make variable j satisfy or unsatisfy the clause a, defined as (see Fig. 2):

$$\text{if } J_j^a = 1\text{: } V_a^u(j) = V_+(j);\ \ V_a^s(j) = V_-(j)\backslash a, \tag{2}$$

$$\text{if } J_j^a = -1\text{: } V_a^u(j) = V_-(j);\ \ V_a^s(j) = V_+(j)\backslash a. \tag{3}$$

The same kind of factor graph representation can be used for other constraint satisfaction problems, where each function node a defines an arbitrary function over the set $X_a \subset X$ of variable nodes to which is connected, and could also involve hidden variables.

3. THE MESSAGE PASSING SOLUTION OF SAT ON A TREE

In the special case in which the factor graph of a SAT problem is a tree (we shall call it a tree-problem), the satisfiability problem can be easily solved by many methods. Here we shall describe two message passing algorithms. The first one, called warning propagation (WP), determines whether a tree-problem is SAT or not; if it is SAT, WP finds one satisfying assignment. The second algorithm, called belief propagation (BP), computes the number of satisfying assignments, as well as the fraction of these assignments where a given variable is set to true. These algorithms are exact for tree-problems, but they can be used as heuristic in general problems, and we first give their general definition, which does not rely on the treelike structure of the factor graph.

3.1. Warning Propagation

The basic elementary message passed from one function node a to a variable i (connected by an edge) is a Boolean number $u_{a\to i} \in \{0, 1\}$ called a "warning."

The update rule is defined as follows. Given a function node a and one of its variables nodes i, the warning $u_{a\to i}$ is determined from the warnings $u_{b\to j}$ arriving on all the variables $j \in V(a)\backslash i$ according to

$$u_{a\to i} = \prod_{j\in V(a)\backslash i} \theta\left(-J_j^a\left(\sum_{b\in V(j)\backslash a} J_j^b u_{b\to j}\right)\right), \tag{4}$$

where $\theta(x) = 0$ if $x \leq 0$ and $\theta(x) = 1$ if $x > 0$. This update rule is used sequentially, resulting in the following algorithm:

WP algorithm
INPUT: the factor graph of a Boolean formula in conjunctive normal form; a maximal number of iterations t_{max}
OUTPUT: UN-CONVERGED if WP has not converged after t_{max} sweeps. If it has converged: the set of all warnings $u^*_{a\to i}$.

---

0. At time $t = 0$: For every edge $a \to i$ of the factor graph, randomly initialize the warnings $u_{a\to i}(t = 0) \in \{0, 1\}$, e.g., with probability 1/2.
1. For $t = 1$ to $t = t_{max}$:
   1.1. sweep the set of edges in a random order (obtained by a permutation which is randomly generated for each t with uniform distribution), and update sequentially the warnings on all the edges of the graph, generating the values $u_{a\to i}(t)$, using subroutine WP-UPDATE.
   1.2. If $u_{a\to i}(t) = u_{a\to i}(t-1)$ on all the edges, the iteration has converged and generated $u^*_{a\to i} = u_{a\to i}(t)$: go to 2.
2. If $t = t_{max}$ return UN-CONVERGED. If $t < t_{max}$ return the set of fixed point warnings $u^*_{a\to i} = u_{a\to i}(t)$

Subroutine WP-UPDATE($u_{a\to i}$)
INPUT: Set of all warnings arriving onto each variable node $j \in V(a)\backslash i$
OUTPUT: new value for the warning $u_{a\to i}$.

1. For every $j \in V(a)\backslash i$, compute the cavity field $h_{j\to a} = \left(\sum_{b\in V_+(j)\backslash a} u_{b\to j}\right) - \left(\sum_{b\in V_-(j)\backslash a} u_{b\to j}\right)$ (If $V(j)\backslash a$ is empty, then $h_{j\to a} = 0$).
2. Using these cavity fields $h_{j\to a}$, compute the warning $u_{a\to i} = \prod_{j\in V(a)\backslash i} \theta\left(h_{j\to a} J^a_j\right)$ (If $V(a)\backslash i$ is empty, then $u_{a\to i} = 1$).

The interpretation of the messages and the message-passing procedure is the following. A warning $u_{a\to i} = 1$ can be interpreted as a message sent from function node a, telling the variable i that it should adopt the correct value in order to satisfy clause a. This is decided by a according to the messages which it received from all the other variables j to which it is connected: if $\left(\sum_{b\in V(j)\backslash a} J^b_j u_{b\to j}\right) J^a_j < 0$, this means that the tendency for site j (in the absence of a) would be to take a value which does not satisfy clause a. If all neighbors $j \in V(a)\backslash i$ are in this situation, then a sends a warning to i. An example of the use of WP is shown in Fig. 3.

The warning propagation algorithm can be applied to any SAT problem. When it converges, this dynamics defines a fixed point, which is a set of warnings $u^*_{a\to i}$. These can be used to compute, for each variable i, the "local field" H_i and the "contradiction number" c_i

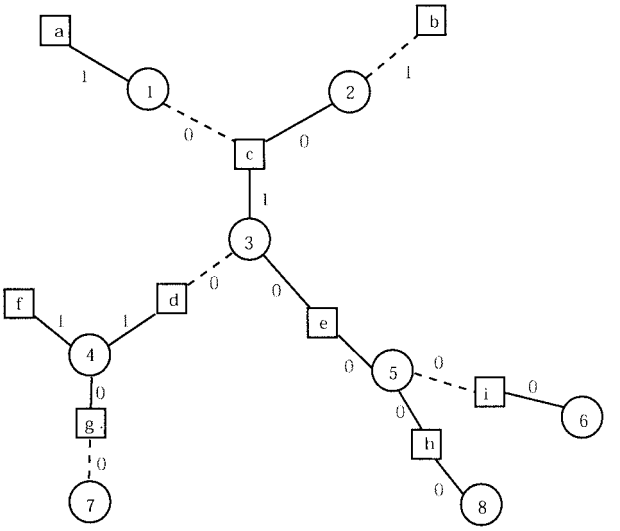

Fig. 3. An example of result obtained by the WP algorithm on a tree-problem with $N = 8$ variables and $M = 9$ clauses. The number on each edge of the graph is the value of the corresponding warning u^*. The local fields on the variable are thus: $1, -1, 1, 2, 0, 0, 0, 0$. The satisfying assignments are such that $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1$, $x_7 \in \{0, 1\}$, $(x_5, x_6, x_8) \in \{(1, 1, 1), (1, 1, 0), (0, 1, 1), (0, 0, 1)\}$. One can check that the variables with nonzero local field take the same value in all SAT assignments. In the WID algorithm, the variables $1, 2, 3, 4$ are fixed to $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1$; the remaining tree has only the clauses h and i remaining, and all the warnings are $u^* = 0$. The variable x_7 will be fixed arbitrarily. If one chooses for instance $x_5 = 1$, the remaining graph leaves x_8 unconstrained but imposes $x_6 = 1$, as can be checked by applying to it the BP algorithm.

which are two integers defined as

$$H_i = -\sum_{b \in V(i)} J_i^b u_{b \to i}^*, \tag{5}$$

$$c_i = 1 \quad \text{if} \quad \left(\sum_{b \in V_+(i)} u_{b \to i}^* \right) \left(\sum_{b \in V_-(i)} u_{b \to i}^* \right) > 0. \tag{6}$$

$$c_i = 0 \quad \text{otherwise.} \tag{7}$$

The local field H_i is an indication of the preferred state of the variable i: $x_i = 1$ if $H_i > 0$, $x_i = 0$ if $H_i < 0$. The contradiction number indicates whether the variable i has received conflicting messages.

The interest in WP largely comes from the fact that it gives the exact solution for tree-problems. This is summarized in the following simple theorem:

Theorem 1. *Consider an instance of the SAT problem with N variables for which the factor graph is a tree. Then the WP algorithm with $t_{max} = N$ converges to a unique set of fixed point warnings $u_{a \to i}^*$, independently on the initial warnings. If at least one of the*

corresponding contradiction numbers c_i is equal to 1, *the problem is UNSAT; otherwise it is SAT.*

Corollary. *In the case where the problem is SAT, the local fields H_i can be used to find an assignment of the variables satisfying all the clauses, using the following algorithm called "Warning Inspired Decimation" or WID:*

WID algorithm
INPUT: the factor graph of a Boolean formula in conjunctive normal form
OUTPUT: UN-CONVERGED, or status of the formula (SAT or UNSAT); If the formula is SAT: one assignment which satisfies all clauses.

---

1. While the number of unfixed variables is > 0, do:
   1.1. Run WP
   1.2. If WP does not converge, return UN-CONVERGED. Else compute the local fields H_i and the contradiction numbers c_i, using Eqs. (5), (7).
   1.3. If there is at least one contradiction number $c_i = 1$, return UNSAT. Else:
      1.3.1. If there is at least one local field $H_i \neq 0$: fix all variables with $H_i \neq 0$ ($H_i > 0 \Rightarrow x_i = 1$ and $H_i < 0 \Rightarrow x_i = 0$), and clean the graph, which means: {remove the clauses satisfied by this fixing, reduce the clauses that involve the fixed variable with opposite literal, update the number of unfixed variables}. GOTO label 1. Else:
      1.3.2. Choose one unfixed variable, fix it to an arbitrary value, clean the graph. GOTO label 1
2. return the set of assignments for all the variables.

Proof of Theorem 1 and of the corollary. The convergence of message passing procedures on tree graphs is a well known result (see, e.g., [18]). We give here an elementary proof of convergence for the specific case of WP, and then show how the results on H_i and c_i follow.

Call $\mathcal{E}$ the set of nodes. Define the leaves of the tree, as the nodes of degree 1. For any edge (a, i) connecting a function node a to a variable node i, define its level r as follows: Remove the edge (a, i) and consider the remaining subgraph containing a. This subgraph $\mathcal{T}_{a\to i}$ is a tree factor graph defining a new SAT problem. The level r is the maximal distance between a and all the leaves in the subgraph $\mathcal{T}_{a\to i}$ (the distance between two nodes of the graph is the number of edges of the shortest path connecting them). If an edge (a, i) has level $r = 0$ (which means that a is a leaf of the subgraph), $u_{a\to i}(t) = 1$ for all $t \geq 1$. If (a, i) has level $r = 1$, then $u_{a\to i}(t) = 0$ for all $t \geq 1$. From the iteration rule, a warning $u_{a\to i}$ at level r is fully determined from the knowledge of all the warnings $u_{b\to j}$ at levels $\leq r - 2$. Therefore, the warning $u_{a\to i}(t)$ at a level r is guaranteed to take a fixed value $u^*_{a\to i}$ for $t \geq 1 + r/2$.

Let us now turn to the study of local fields and contradiction numbers.

We first prove the following lemma.

Lemma. *If a warning $u^*_{a\to i} = 1$, the clause a is violated in the reduced SAT problem defined by the subgraph $\mathcal{T}_{a\to i}$.*

This is obviously true if the edge has level $r = 0$ or $r = 1$. Supposing that it holds for all levels $\leq r - 2$, one considers an edge (a, i) at level r, with $u^*_{a\to i} = 1$. From (4), this means that for all variable nodes $j \in V(a)\backslash i$, the node j receives at least one message from a neighboring factor node $b \in V(j)\backslash a$ with $u_{b\to j} = 1$. The edge (b,j) is at level $\leq r - 2$; therefore, the reduced problem on the graph $\mathcal{T}_{(b,j)}$ is UNSAT, and, therefore, the clause b imposes the value of the variable j to be 1 (True) if $J^b_j = -1$, or 0 (False) if $J^b_j = 1$: We shall say that clause b fixes the value of variable j. This is true for all $j \in V(a)\backslash i$, which means that the reduced problem on $\mathcal{T}_{a\to i}$ is UNSAT, or, equivalently, the clause a fixes the value of variable i.

Having shown that $u^*_{a\to i} = 1$ implies that clause a fixes the value of variable i, it is clear from (7) that a nonzero contradiction number c_i implies that the formula is UNSAT.

If all the c_i vanish, the formula is SAT. One can prove this for instance by showing that the WID algorithm generates a SAT assignment. The variables with $H_i \neq 0$ receive some nonzero $u^*_{a\to i} = 1$ and are fixed. One then "cleans" the graph, which means: Remove the clauses satisfied by this fixing, reduce the clauses that involve the fixed variable with opposite literal. By definition, this process has removed from the graph all the edges on which there was a nonzero warning. So, on the new graph, all the edges have $u^* = 0$. Following the step 2.2 of WID, one chooses randomly a variable i, fixes it to an arbitrary value x_i, and cleans the graph. The clauses a connected to i which are satisfied by the choice x_i are removed; the corresponding subgraphs are trees where all the edges have $u^* = 0$. A clause a connected to i which is not satisfied by the choice x_i may send some $u^* = 1$ messages (this happens if such a clause had degree 2 before fixing variable i). However, running WP on the corresponding subgraph $\mathcal{T}_{a\to i}$, the set of warnings cannot have a contradiction: A variable j in this subgraph can receive at most one $u^* = 1$ warning, coming from the unique path which connects j to a. Therefore, $c_j = 0$: one iteration of WID has generated a strictly smaller graph with no contradiction. By induction, it thus finds a SAT assignment. ∎

One should notice that the variables which are fixed at the first iteration of WID (those with nonzero H_i) are constrained to take the same value in all satisfying assignments.

3.2. Belief Propagation

While the WP algorithm is well adapted to finding a SAT assignment, the more complicated belief propagation (BP) algorithm is able to compute, for satisfiable problems with a tree factor graph, the total number of SAT assignments, and the fraction of SAT assignments where a given variable x_i is true.

We consider a satisfiable instance, and the probability space built by all SAT assignments taken with equal probability. We will also consider the space of SAT assignments with uniform probability for a graph built from the original one by removing a given clause a.

Calling a one of the clauses in which x_i appears, the basic ingredients of BP are the messages:

- $\mu_{a\to i}(x_i) \in [0, 1]$, the probability that clause a is satisfied in the space of SAT assignments of the graph without clause a, given the value of x_i.
- $\mu_{i\to a}(x_i) \in [0, 1]$, the probability that the variable with index i takes value x_i in the space of SAT assignments of the graph without clause a (this is again a typical "cavity" definition).

Notice that $\sum_{x_i \in \{0,1\}} \mu_{i\to a}(x_i) = 1$, while there is no such normalization for $\mu_{a\to i}(x_i)$. The BP equations are

$$\mu_{i\to a}(x_i) = C_{i\to a} \prod_{b\in V(i)\setminus a} \mu_{b\to i}(x_i), \tag{8}$$

$$\mu_{a\to i}(x_i) = \sum_{\{x_j (j\neq i)\}} f_a(X) \prod_{j\in V(a)\setminus i} \mu_{j\to a}(x_j), \tag{9}$$

where $C_{i\to a}$ is a normalization constant ensuring that $\mu_{i\to a}$ is a probability, the sum over $\{x_j (j \neq i)\}$ means a sum over all values of the variables $x_j \in \{0, 1\}$, for all j different from i, and $f_a(X)$ is a characteristic function taking value 1 if the configuration $X = \{x_i\}$ satisfies clause a, taking value 0 otherwise.

It is convenient to parametrize $\mu_{i\to a}(x_i)$ by introducing the number $\gamma_{i\to a} \in [0, 1]$ which is the probability that the variable x_i is in the state which violates clause a, in a problem where clause a would be absent. That is,

$$\mu_{i\to a}(x_i) = \gamma_{i\to a}\delta(x_i, \tfrac{1}{2}(1 + J_i^a)) + (1 - \gamma_{i\to a})\delta(x_i, \tfrac{1}{2}(1 - J_i^a)). \tag{10}$$

Let us denote by

$$\delta_{a\to i} \equiv \prod_{j\in V(a)\setminus i} \gamma_{j\to a} \tag{11}$$

the probability that all variables in clause a, except variable i, are in the state which violates the clause.

The BP algorithm amounts to an iterative update of the messages $\delta_{a\to i}$ according to the rule:

BP algorithm: INPUT: the factor graph of a Boolean formula in conjunctive normal form; a maximal number of iterations t_{max}; a requested precision ϵ.
OUTPUT: UN-CONVERGED if BP has not converged after t_{max} sweeps. If it has converged: the set of all messages $\delta^*_{a\to i}$.

---

0. At time $t = 0$: For every edge $a \to i$ of the factor graph, randomly initialize the messages $\delta_{a\to i}(t = 0) \in [0, 1]$
1. For $t = 1$ to $t = t_{max}$:
   1.1. Sweep the set of edges in a random order (obtained by a permutation which is randomly generated for each t with uniform distribution), and update sequentially the messages on all the edges of the graph, generating the values $\delta_{a\to i}(t)$, using subroutine BP-UPDATE.
   1.2. If $|\delta_{a\to i}(t) - \delta_{a\to i}(t - 1)| < \epsilon$ on all the edges, the iteration has converged and generated $\delta^*_{a\to i} = \delta_{a\to i}(t)$: go to 2.
2. If $t = t_{max}$, return UN-CONVERGED. If $t < t_{max}$, return the set of fixed point messages $\delta^*_{a\to i} = \delta_{a\to i}(t)$.

Subroutine BP-UPDATE($\delta_{a\to i}$)
INPUT: Set of all messages arriving onto each variable node $j \in V(a)\backslash i$
OUTPUT: new value for the message $\delta_{a\to i}$.

1. For every $j \in V(a)\backslash i$, compute the cavity field

$$\gamma_{j\to a} = \frac{P^u_{j\to a}}{P^u_{j\to a} + P^s_{j\to a}}, \tag{12}$$

where

$$P^u_{j\to a} = \prod_{b\in V^s_a(j)} (1 - \delta_{b\to j}) ,$$
$$P^s_{j\to a} = \prod_{b\in V^u_a(j)} (1 - \delta_{b\to j}) . \tag{13}$$

If an ensemble is empty, for instance $V^s_a(j) = \emptyset$, the corresponding $P^u_{j\to a}$ takes value 1 by definition.

2. Using these numbers $\gamma_{j\to a}$, compute the new message: $\delta_{a\to i} \equiv \prod_{j\in V(a)\backslash i} \gamma_{j\to a}$. If a factor node a is a leaf (unit clause) with a single neighbor i, the corresponding $\delta_{a\to i}$ takes value 1 by definition.

Note that this procedure corresponds simply to the application of the right-hand side of Eqs. (8), (9).

As WP, the BP algorithm is exact on trees (see, for instance, [18]). In fact it gives a more accurate results than WP since it allows to compute the exact probabilities μ_i (while WP identifies the variables which are fully constrained, and gives a zero local field on the other variables). A working example of BP is shown in Fig. 4. In this example and more in general for trees, BP also provides the exact number $\mathcal{N}$ of SAT assignments, as given by the following theorem:

Theorem 2. *Consider an instance of the SAT problem for which the factor graph is a tree, and there exist some SAT assignments. Then:*

a. The BP algorithm converges to a unique set of fixed point messages $\delta^*_{a\to i}$.
b. The values $\gamma^*_{i\to a}$ computed in the fixed point by the BP procedure parametrize the exact messages $\mu_{i\to a}$, i.e., Eq (10) holds for $\gamma_{i\to a} = \gamma^*_{i\to a}$.
c. The probability marginal μ_i that the variable $x_i = 1$ (the ratio between satisfying assigments in which $x_i = 1$ and the total number of satisfying assignments) is given by

$$\mu_i = \frac{\prod_{a\in V_-(i)}(1 - \delta^*_{a\to i})}{\prod_{a\in V_-(i)}(1 - \delta^*_{a\to i}) + \prod_{a\in V_+(i)}(1 - \delta^*_{a\to i})} . \tag{14}$$

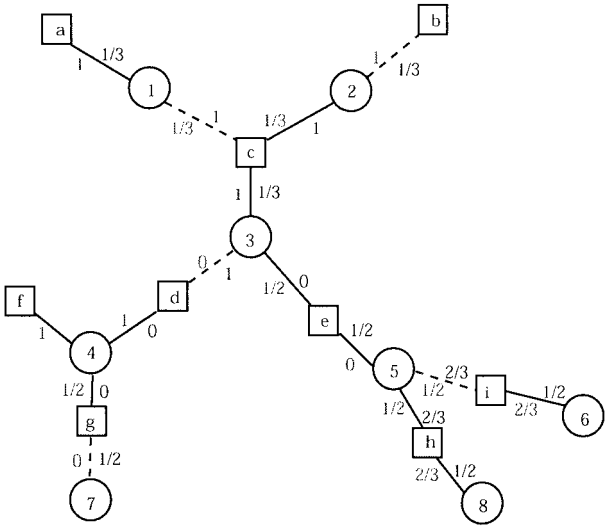

Fig. 4. An example of result obtained by the BP algorithm on the tree-problem with $N = 8$ variables and $M = 9$ clauses studied in Fig. 3.1. On each edge of the graph connecting a function node like c to a variable node like 2, appears the value of $\gamma_{2\to c}$ (in blue, on the right-hand side of the edge) and of $\delta_{c\to 2}$ (in red, on the left-hand side of the edge). Comparing to the WP result of Fig. 3.1, one sees that all the messages $\delta_{a\to i} = 1$, corresponding to strict warnings, are the same, while the messages $\delta_{a\to i} < 1$ are interpreted in WP as "no warning" (i.e., a null message). Using (14), the probability μ_i that each variable $x_i = 1$ are found equal to: $1, 0, 1, 1, 1/2, 3/4, 3/4, 1/2$. These are the exact results as can be checked by considering all satisfying assignments as in Fig. 3.1.

d. The number of SAT assignments is $\mathcal{N} = \exp(S)$, where the entropy S is given by

$$S = \sum_{a\in A} \ln\left[\prod_{i\in V(a)}\left(\prod_{b\in V_a^s(i)}(1-\delta^*_{b\to i}) + \prod_{b\in V_a^u(i)}(1-\delta^*_{b\to i})\right) - \prod_{i\in V(a)}\left(\prod_{b\in V_a^u(i)}(1-\delta^*_{b\to i})\right)\right] + \sum_{i\in X}(1-n_i)\ln\left[\prod_{b\in V_+(i)}(1-\delta^*_{b\to i}) + \prod_{b\in V_-(i)}(1-\delta^*_{b\to i})\right]. \tag{15}$$

Proof. Results a, b, and c are standard for BP (also called the "sum-product algorithm"); the reader can learn more about it in [18]. We just outline the main lines of their proofs for completeness.

a. The proof of convergence is simple, using the same strategy as the proof in Section 3: Messages at level 0 and 1 are fixed automatically, and a message at level r is fixed by the values of messages at lower levels.

b. In a tree factor graph, using the fact that by removing a clause all participating variables belong to different connected components, it is straightforward to verify that the BP equations (8), (9) are exact equations for the correct cavity marginals $\mu_{i\to a}$. Then the

statement in b is a consequence of the fact that the fixed point is unique. An alternative proof can be obtained by induction on the level of the edge (a, i).

c. Note that μ_i in (14) are computed with a formula similar to the one giving the BP equations (8), with the difference that one takes into account all neighbors of the site i. Precisely, using point b, $\mu_i = p_i(x_i = 1)$, where

$$p_i(x_i) = C_i \prod_{b \in V(i)} \mu^*_{b \to i}(x_i). \tag{16}$$

[C_i is a normalization constant ensuring that $p_i(0) + p_i(1) = 1$.] It is easy to check that $p_i(x_i)$ is the said probability marginal.

d. The slightly more involved result is the one concerning the entropy. We use the probability measure $P(X)$ on the space of all assignments which has uniform probability for all SAT assignments and zero probability for all the assignments which violate at least one clause:

$$P(X) = \frac{1}{\mathcal{N}} \prod_{a \in A} f_a(X). \tag{17}$$

From P one can define the following marginals:

- The "site marginal" $p_i(x_i)$ is the probability that variable i takes value $x_i \in \{0, 1\}$.
- The "clause marginal" $p_a(X_a)$ is the probability that the set of variables $x_i, i \in V(a)$, takes a given value, denoted by X_a (among the 2^{n_a} possible values).

For a tree factor graph, one easily shows by induction on the size of the graph that the full probability can be expressed in terms of the site and clause marginals as

$$P(X) = \prod_{a \in A} p_a(X_a) \prod_{i \in X} p_i(x_i)^{1-n_i}. \tag{18}$$

The entropy $S = \ln(\mathcal{N})$ is then obtained as

$$\begin{aligned} S = -\sum_X P(X) \ln P(X) = &-\sum_a \sum_{X_a} p_a(X_a) \ln[p_a(X_a)] \\ &- \sum_i (1 - n_i) \sum_{x_i} p_i(x_i) \ln[p_i(x_i)]. \end{aligned} \tag{19}$$

Let us now derive the expression of this quantity in terms of the messages used in BP. $p_i(x_i)$ has been found in (16). As for p_a, it is easy to see that it is equal to

$$p_a(X_a) = C_a f_a(X_a) \prod_{i \in V(a)} \mu_{i \to a}(x_i), \tag{20}$$

where C_a is a normalization constant. From (16) one gets after some reshuffling

$$\begin{aligned} \sum_i (n_i - 1) \sum_{x_i} p_i(x_i) \ln p_i(x_i) = &\sum_i (n_i - 1) \ln C_i + \sum_a \sum_{i \in V(a)} \sum_{X_a} p_a(X_a) \\ &\times \ln \left[\prod_{b \in V(i) \setminus a} \mu_{b \to i}(x_i) \right]; \end{aligned} \tag{21}$$

Using the BP equation (8), this gives

$$\sum_i (n_i - 1) \sum_{x_i} p_i(x_i) \ln p_i(x_i) = \sum_i (n_i - 1) \ln C_i + \sum_a \sum_{X_a} p_a(X_a) \ln \left[\prod_{i \in V(a)} \mu_{i \to a}(x_i) f_a(X_a) \right] - \sum_a \sum_{i \in V(a)} \ln C_{i \to a}, \tag{22}$$

where the term $f_a(X_a)$ inside the logarithm has been added, taking into account the fact that, as $f_a(X_a) \in \{0, 1\}$, one always has $p_a(X_a) \ln f_a(X_a) = 0$. Therefore,

$$S = -\sum_a \ln C_a + \sum_i (n_i - 1) \ln C_i - \sum_a \sum_{i \in V(a)} \ln C_{i \to a}. \tag{23}$$

In the notations of (13), one has

$$C_a = \frac{1}{1 - \prod_{i \in V(a)} \gamma_{i \to a}} = \frac{1}{1 - \prod_{i \in V(a)} P^u_{i \to a} / (P^u_{i \to a} + P^s_{i \to a})}, \tag{24}$$

$$C_{i \to a} = \frac{1}{P^u_{i \to a} + P^s_{i \to a}}, \tag{25}$$

and

$$C_i = \frac{1}{\prod_{b \in V_+(i)} (1 - \delta_{b \to i}) + \prod_{b \in V_-(i)} (1 - \delta_{b \to i})}. \tag{26}$$

Substitution into (23) gives the expression (15) for the entropy. ∎

4. SURVEY PROPAGATION

4.1. The Algorithm

The WP and BP algorithms have been shown to work for satisfiability problems where the factor graph is a tree. In more general cases where the factor graph has loops, they can be tried as heuristics, but there is no guarantee of convergence. In this section we present a new message passing algorithm, survey propagation (SP), which is also a heuristic, without any guarantee of convergence. It reduces to WP for tree-problems, but it turns out to be more efficient than WP or BP in experimental studies of random satisfiability problems. SP has been discovered using concepts developed in statistical physics under the name of "cavity method." Here we shall first present SP, then give some experimental results, and in the end expose the qualitative physical reasoning behind it.

A message of SP, called a survey, passed from one function node a to a variable i (connected by an edge) is a real number $\eta_{a \to i} \in [0, 1]$. The SP algorithm uses exactly the

same main procedure as BP (see Section 3.2), but instead of calling the BP-UPDATE, it uses a different update rule, SP-UPDATE, defined as:

Subroutine SP-UPDATE($\eta_{a\to i}$)
INPUT: Set of all messages arriving onto each variable node $j \in V(a)\backslash i$
OUTPUT: new value for the message $\eta_{a\to i}$.

---

1. For every $j \in V(a)\backslash i$, compute the three numbers:

$$\Pi^u_{j\to a} = \left[1 - \prod_{b\in V^u_a(j)} \left(1-\eta_{b\to j}\right)\right] \prod_{b\in V^s_a(j)} \left(1-\eta_{b\to j}\right),$$

$$\Pi^s_{j\to a} = \left[1 - \prod_{b\in V^s_a(j)} \left(1-\eta_{b\to j}\right)\right] \prod_{b\in V^u_a(j)} \left(1-\eta_{b\to j}\right),$$

$$\Pi^0_{j\to a} = \prod_{b\in V(j)\backslash a} \left(1-\eta_{b\to j}\right) \tag{27}$$

   if a set like $V^s_a(j)$ is empty, the corresponding product takes value 1 by definition.
2. Using these numbers, compute and return the new survey:

$$\eta_{a\to i} = \prod_{j\in V(a)\backslash i} \left[\frac{\Pi^u_{j\to a}}{\Pi^u_{j\to a} + \Pi^s_{j\to a} + \Pi^0_{j\to a}}\right]. \tag{28}$$

   If $V(a)\backslash i$ is empty, then $\eta_{a\to i} = 1$.

Qualitatively, the statistical physics interpretation of the survey $\eta_{a\to i}$ is a probability that a warning is sent from a to i (see Section 5 for details). Therefore, whenever the SP algorithm converges to a fixed-point set of messages $\eta^*_{a\to i}$, one can use it in a decimation procedure in order to find a satisfying assignment, if such an assignment exists. This procedure, called the survey inspired decimation (SID), is a generalization of the WID algorithm 3.1, defined by:

SID algorithm
INPUT: The factor graph of a Boolean formula in conjunctive normal form. A maximal number of iterations t_{max} and a precision ϵ used in SP
OUTPUT: One assignment which satisfies all clauses, or "SP UNCONVERGED," or "probably UNSAT"

---

0. Random initial condition for the surveys
1. Run SP. If SP does not converge, return 'SP UNCONVERGED' and stop (or restart, i.e. go to 0.). If SP converges, use the fixed-point surveys $\eta^*_{a\to i}$ in order to:
2. Decimate:
   2.1. If nontrivial surveys ($\{\eta \neq 0\}$) are found, then:

a. Evaluate, for each variable node i, the two "biases" $\{W_i^{(+)}, W_i^{(-)}\}$ defined by

$$W_i^{(+)} = \frac{\hat{\Pi}_i^+}{\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0}, \tag{29}$$

$$W_i^{(-)} = \frac{\hat{\Pi}_i^-}{\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0}, \tag{30}$$

where $\hat{\Pi}_i^+, \hat{\Pi}_i^-, \hat{\Pi}_i^0$ are defined by

$$\begin{aligned}
\hat{\Pi}_i^+ &= \left[1 - \prod_{a \in V_+(i)} \left(1 - \eta_{a\to i}^*\right)\right] \prod_{a \in V_-(i)} \left(1 - \eta_{a\to i}^*\right), \\
\hat{\Pi}_i^- &= \left[1 - \prod_{a \in V_-(i)} \left(1 - \eta_{a\to i}^*\right)\right] \prod_{a \in V_+(i)} \left(1 - \eta_{a\to i}^*\right), \\
\hat{\Pi}_i^0 &= \prod_{a \in V(i)} \left(1 - \eta_{a\to i}^*\right).
\end{aligned} \tag{31}$$

b. Fix the variable with the largest $|W_i^{(+)} - W_i^{(-)}|$ to the value $x_i = 1$ if $W_i^{(+)} > W_i^{(-)}$, to the value $x_i = 0$ if $W_i^{(+)} < W_i^{(-)}$. Clean the graph, which means: {remove the clauses satisfied by this fixing, reduce the clauses that involve the fixed variable with opposite literal, update the number of unfixed variables}.

2.2. If all surveys are trivial ($\{\eta = 0\}$), then output the simplified subformula and run on it a local search process (e.g., walksat [33, 34]).

4. If the problem is solved completely by unit clause propagation, then output "SAT" and stop. If no contradiction is found, then continue the decimation process on the smaller problem (go to 1.) Else (if a contradiction is reached) stop.

There exist several variants of this algorithm. In the code which is available at [37], for performance reasons we update simultaneously all η belonging to the same clause. The clauses to be updated are chosen in a random permutation order at each iteration step. The algorithm can also be randomized by fixing, instead of the most biased variables, one variable randomly chosen in the set of the x percent variables with the largest bias. This strategy allows to use some restart in the case where the algorithm has not found a solution. A fastest decimation can also be obtained by fixing in the step 2.1(b), instead of one variable, a fraction f of the N_t variables which have not yet been fixed (going back to 1 variable when $fN_t < 1$).

4.2. Experimental Study of the SP Algorithm

In order to get some concrete information on the behaviour of SP for large but finite N, we have experimented SP and SID on single instances of the random 3-SAT problem with many

variables, up to $N \sim 10^7$. In this section we summarize these (single machine) experiments and their results.

Instances of the 3-SAT problem were generated with the pseudo random number generator "Algorithm B" on page 32 of Knuth [17]. However, we found that results are stable with respect to changes in the random number generators. Formulas are generated by choosing k-tuples of variable indices at random (with no repetitions) and by negating variables with probability 0.5.

We first discuss the behavior of the SP algorithm itself. We have used a precision parameter $\epsilon = 10^{-3}$ (smaller values don't seem to increase performance significantly). Depending on the range of α, we have found the following behaviours, for large enough N:

- For $\alpha < \alpha_d \sim 3.9$, SP converges towards the set of trivial messages $\eta_{a\to i} = 0$, for all (a, i) edges. All variables are under-constrained.
- For $3.9 < \alpha < 4.3$, SP converges to a unique fixed-point set of nontrivial messages, independently from the initial conditions, where a large fraction of the messages $\eta_{a\to i}$ are in $]0, 1[$.

For "small' values of N, around $N = 1000$, one often finds some instances in which SP does not converge. But the probability of convergence, at a given $\alpha < 4.3$, increases with N. This is exemplified by the following quantitative measure of the performance of the SID algorithm (which uses SP). We have solved several instances of the random 3-SAT problem, for various values of α and N, using the SID algorithm in which we fix at each step the fraction fN_t of variables with largest $\left|W_i^{(+)} - W_i^{(-)}\right|$. Table 1 gives in each case the fraction of samples which are solved by SID, in a single run of decimation (without any restart). The algorithm fails when, either SP does not converge, or the simplified subformula found by SID is not solved by walksat. The performance of SID improves when N increases and when f decreases. Notice that for $N = 10^5$ we solve all the 50 randomly generated instances at $\alpha = 4.24$. For larger values of α the algorithm often fails. Unfortunately, in such cases it does not give any information on whether the instance is UNSAT. Some failures may be due to UNSAT instances, others are just real failures of the SID for SAT instances. Few experiments on even larger instances ($N = 10^6, 10^7$) have also been successfully run (the limiting factor being the available computer memory needed to store formulas).

As shown by the data on the average total number of SP iterations along the successful solution process in Table 1, the convergence time of the SP algorithm basically does not grow with N (a growth like $\log N$, which could be expected from the geometrical properties of the factor graph, is not excluded). Therefore, the process of computing all the SP messages $\eta^*_{a\to i}$ takes $\Theta(N)$, or maybe $\Theta(N \ln N)$, operations. If SID fixes at each step only one variable, it will thus converge in $\Theta(N^2 \log N)$ operations (the time taken by walksat to solve the simplified sub-formula seems to grow more slowly). When we fix a fraction of variables at a time, we get a further reduction of the cost to $O(N(\ln N)^2)$ (the second ln comes from sorting the biases).

A very basic yet complete version of the code which is intended to serve only for the study on random 3-SAT instances is available at the web site [37]. Generalization of the algorithm to other problems require some changes which are not implemented in the distributed code.

TABLE 1. Results Obtained by Solving with a Single Decimation Run of the SID Algorithm 50 Random Instances of 3-SAT with $N = 25000, 50000, 100000$ and $\alpha = 4.21, 4.22, 4.23, 4.24$

$N =$	2.5×10^4				5.0×10^4				1.0×10^5			
$f \backslash \alpha$	4.21	4.22	4.23	4.24	4.21	4.22	4.23	4.24	4.21	4.22	4.23	4.24
4%	86%	66%	28%	8%	98%	84%	52%	22%	100%	100%	72%	22%
2%	100%	86%	50%	22%	100%	98%	86%	48%			100%	68%
1%		94%	78%	32%		100%	94%	64%				88%
0.5%		98%	88%	50%			98%	66%				92%
0.25%		100%	90%	60%			100%	78%				92%
0.125%			94%	60%				84%				100%
$< t >$	1369	2428	4235	7843	1238	1751	3411	8607	1204	1557	2573	7461

SID was used by fixing variables in blocks fN_t, where N_t is the number of unfixed variables at time t, and various runs used different values of f taken in the geometric progression $f = 4\%, 2\%, 1\%, .5\%, .25\%, .125\%$, stopping if the formula was solved. For each value of (N, α, f), we give the fraction of the 50 instances which were solved (i.e. for which the algorithm found a SAT assignment). The maximal number of iterations in each call to SP was taken equal to $t_{max} = 10^3$ and the precision for convergence was taken equal to 10^{-3}. We call t the cumulative number of calls to SP during the full SID run. The last row gives $< t >$, the expectation value of t averaged over the successful runs.

5. HEURISTIC ARGUMENTS

Survey propagation has been invented using powerful concepts and methods developed in the statistical physics of disordered systems, notably the cavity method for diluted problems [21]. In this section we want to give some short background on these methods, in order to help the reader understand where SP comes from, and maybe develop similar algorithms in other contexts. Unfortunately, so far there is no rigorous derivation of the cavity method, so this whole section only contains heuristic arguments.

5.1. The Physical Picture Underlying the SP Construction: Clustering of Configurations

Let us start with a discussion of the validity of BP. As we saw in section 3, BP aims at computing the marginal probability distribution of a variable x_i, within the probability space built by all SAT assignments, each being given equal probability. The message $\mu_{a\to i}(x_i)$ used in BP can be computed exactly if one knows the joint probability distribution $P^{(a)}(X)$ of the variables in $X = \{x_j,\ j \in V(a)\backslash i\}$, in the graph where clause a is absent. Using the same notations as in (9), one has

$$\mu_{a\to i}(x_i) = \sum_{\{x_j (j\neq i)\}} f_a(X) P^{(a)}(X) \tag{32}$$

Comparing this Eq. (32) to the Eqs. (8), (9) of BP, one sees that BP uses an approximation, namely, the fact that the joint probability $P^{(a)}(X)$ factorizes: $P^{(a)}(X) \simeq \prod_{j\in V(a)\backslash i} \mu_{j\to a}(x_j)$. This amounts to assuming that the variables x_j, for $j \in V(a)\backslash i$, are uncorrelated in the absence of clause a. This assumption is obviously correct when the factor graph is a tree,

which confirms the validity of BP in that case. In a general problem, for such an assumption to hold, we need two conditions to be fulfilled:

- The variables $x_j, j \in V(a)\backslash i$ should be far from each other, in the factor graph where clause a is absent.
- There should be a single phase in the problem, or the probability measure should be reduced to one single pure phase.

The first condition is easily understood, and one can see that it is generically fulfilled when one considers a random satisfiability problem. In random K-sat with $M = \alpha N$ clauses, the factor graph is a random bipartite graph, where function nodes have degree K and variable nodes have fluctuating degrees, with a distribution which becomes, in the large N limit, a Poisson law of mean $K\alpha$. Locally such a graph is tree like. Taking a clause a at random, the minimal distance between two of the variables $x_j, j \in V(a)\backslash i$ is generically of order $\log N$.

The second condition is more subtle. In general a pure phase is defined in statistical physics as an extremal Gibbs measure [14]; however, the standard construction of Gibbs measures deals with infinite systems. Here we need to work with N variables where $N \gg 1$ but is finite, and the corresponding construction has not been worked out yet. For the random satisfiability problem, a heuristic description of a pure phase is a cluster of SAT assignments, defined as follows. Define the distance between two assignments $\{x_i\}$ and $\{y_i\}$ as the number of bits where they differ. Then build an auxiliary graph, in which the vertices are all the SAT assignments, and one places an edge between two vertices if and only if the distance the distance between the two corresponding SAT assignments is smaller than a number q. Each connected component in this auxiliary graph is a q-cluster. One is interested in the "clusters" obtained in large N-large q limit of the q-clusters, where the large N limit is taken first. Heuristic statistical physics arguments indicate that, in the random K-satisfiability problem, for $\alpha < \alpha_{clust}(K)$, there should exist one single such cluster of SAT assignments: This means that one can move inside the space of SAT assignment, from any assignment to another one, by a succession of moves involving each a number of flips of variables which is $\ll N$. In such a case the BP factorization approximation is expected to be correct. On the other hand, for $\alpha > \alpha_{clust}(K)$ the space of SAT assignment separates into many distant clusters, and the BP factorization does not hold globally, but it would hold if one could restrict the probability space to one given cluster α. Within such a restricted space, BP would converge to a set of messages $\mu^{\alpha}_{a\to i}(x_i)$ which depends on the cluster α.

In this situation, the cavity method uses a statistical approach. It considers all the clusters of SAT assignments, and attributes to each cluster a probability proportional to the number of configuration that it contains. Then one introduces, on each edge (a, i), a survey which gives the probability, when a cluster α is chosen randomly with this probability, that the message $\mu^{\alpha}_{a\to i}(x)$ is equal to a certain function $P(x)$.

This object is a probability of a probability and it is thus difficult to use in practical algorithms. For this reason, SP departs from the usual cavity method and uses a simpler object which is a survey of warnings, interpreted as follows: Consider one cluster α and an edge (a, i) of the factor graph. If, in every SAT assignments of the cluster α, all the variables $x_j, j \in V(a)\backslash i$ don't satisfy clause a, then a warning $u^{\alpha}_{a\to i} = 1$ is passed along the edge from a to i. The SP message along this edge is the survey of these warnings, when one picks up a cluster α at random: $\eta_{a\to i} = \sum_{\alpha} u^{\alpha}_{a\to i}/(\sum_{\alpha} 1)$. So basically the SP message gives the probability that there is a warning sent from a to i in a randomly chosen cluster. With

respect to the full-fledged cavity method, this is a much simplified object, which focuses onto the variables which are constrained.

The experimental results on random 3-satisfiability discussed in sect. 4.2 confirm the theoretical analysis of [22, 24] which indicate that all $\eta_{a\to i}$ vanish for $\alpha < \alpha_d \simeq 3.91$. This can be interpreted as the fact that, in this range of α, there are no "constrained clusters" (meaning clusters in which some of the variables are constrained). For $\alpha_d < \alpha < \alpha_c = 4.267$ the theory predicts the existence of nontrivial messages, meaning that there exist constrained clusters. This is the region where SP and SID are able to outperform existing algorithms. One should notice that the SID does not seem to converge up to the conjectured threshold $\alpha_c = 4.267$: Although it is very small, there remains a region in the "Hard SAT" phase, close to α_c where the simple version of SP/SID does not give the result. Recent work shows that this gap can be reduced by using some backtracking strategy in SID [31]. Whether it will be possible to close the gap with generalized versions of SP/SID, while keeping a typically polynomial running time, is an interesting open issue.

5.2. The "Don't-Care" State

In a given cluster α, a variable x_i can be thought of being in three possible states: either it is constrained equal to 0 (this means that $x_i = 0$ in all SAT assignments of the cluster α), or it is constrained equal to 1, or it is not constrained. In this last situation we attribute it the value $*$. Therefore, we can describe a cluster by the values of the N generalized variables $x_i^\alpha \in \{0, 1, *\}$, where $*$ will be denoted as the don't-care state. Such a description associates to each cluster a single point in $\{0, 1, *\}^N$. It discards a lot of information: $x_i^\alpha = *$ has lost all the information on the fraction of assignments in the cluster α where $x_i = 0$. But it gives a simplified description of the cluster and it focuses onto the constrained variables.

It is interesting to notice that the SP equations can be interpreted as BP equations in the presence of this extra don't-care state. This can be seen as follows. Borrowing the notations of the BP equations (8), (9) we denote by $\gamma_{i\to a} \in [0, 1]$ the probability that the variable x_i is in the state which violates clause a, in a problem where clause a would be absent, and by

$$\delta_{a\to i} = \prod_{j\in V(a)\backslash i} \gamma_{j\to a}, \tag{33}$$

the probability that all variables in clause a, except variable i, are in the state which violates the clause.

Let us compute $\mu_{i\to a}(x_i)$. This depends on the messages sent from the nodes $b \in \{V(i)\backslash a\}$ to variable i. The various possibilities for these messages are:

- No warning arriving from $b \in V_a^s(i)$, and no warning arriving from $b \in V_a^u(i)$. This happens with a probability

$$\Pi_{i\to a}^0 = \prod_{b\in V(i)\backslash a} (1 - \delta_{b\to i}) . \tag{34}$$

- No warning arriving from $b \in V_a^s(i)$, and at least one warning arriving from $b \in V_a^u(i)$. This happens with a probability

$$\Pi_{i\to a}^u = \left[1 - \prod_{b\in V_a^u(i)} (1 - \delta_{b\to i})\right] \prod_{b\in V_a^s(i)} (1 - \delta_{b\to i}) . \tag{35}$$

- No warning arriving from $b \in V_a^u(i)$, and at least one warning arriving from $b \in V_a^s(i)$. This happens with a probability

$$\Pi_{i\to a}^s = \left[1 - \prod_{b\in V_a^s(i)} (1-\delta_{b\to i})\right] \prod_{b\in V_a^u(i)} (1-\delta_{b\to i})\,. \tag{36}$$

- At least one warning arriving from $b \in V_a^u(i)$, and at least one warning arriving from $b \in V_a^s(i)$. This happens with a probability

$$\Pi_{i\to a}^C = \left[1 - \prod_{b\in V_a^s(i)} (1-\delta_{b\to i})\right] \left[1 - \prod_{b\in V_a^u(i)} (1-\delta_{b\to i})\right]. \tag{37}$$

As we work only with SAT configurations, the contradictory messages must be excluded. Therefore, the probability $\gamma_{i\to a} \in [0,1]$ that the variable x_i is in the state which violates clause a, given that there is no contradiction, is

$$\gamma_{i\to a} = \Pi_{i\to a}^u / (\Pi_{i\to a}^u + \Pi_{i\to a}^s + \Pi_{i\to a}^0). \tag{38}$$

The above equations in the enlarged space including the null message, given in (33)–(38) are identical to the SP Eqs. (28), (27), with the identification $\eta_{a\to i} = \delta_{a\to i}$.

5.3. Complexity

The above interpretation of SP using the don't-care state suggests a method to estimate the complexity, defined as the normalized logarithm of the number of constrained clusters of SAT assignments. As each constrained cluster of SAT assignments is associated with a point in $\{0,1,*\}^N$, the complexity should be given by the corresponding entropy, which can be estimated with the usual BP formula (15) in the presence of the don't-care state (see Ref. [5] for a rigorous derivation). The result, originally derived in [24] from a direct statistical physics analysis without the use of the don't-care state, is as follows.

The total complexity Σ can be decomposed into contributions associated with every function node and with every variable, and reads

$$\Sigma = \sum_{a=1}^{M} \Sigma_a - \sum_{i=1}^{N} (n_i - 1)\,\Sigma_i, \tag{39}$$

where

$$\Sigma_a = +\log\left[\prod_{j\in V(a)} \left(\Pi_{j\to a}^u + \Pi_{j\to a}^s + \Pi_{j\to a}^0\right) - \prod_{j\in V(a)} \Pi_{j\to a}^u\right], \tag{40}$$

$$\Sigma_i = +\log\left[\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0\right]. \tag{41}$$

In Fig. 5 we report the data for the complexity of random 3-SAT formulae of size $N = 10^6$ and α in the clustering range. This has been obtained with the following procedure. One generates first a "random" 3-SAT formula with $N = 10^6$ and $M = 4.27\,10^6$, using a pseudo-random number generator. The SP algorithm is run on this formula, and the complexity is

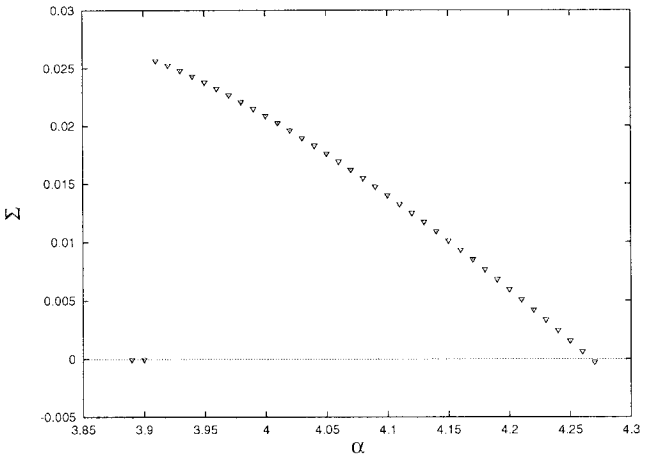

Fig. 5. Complexity per variable (Σ/N) of the satisfying clusters for a given sample of size $N = 10^6$. The complexity vanishes at α_c which is expected to be the critical threshold for the specific formula under study.

evaluated from (39). Then a new formula is generated from the previous one by eliminating 10^4 (pseudo-)randomly chosen clauses, and the algorithm is run again, etc. It turns out that, for $N = 10^6$, the resulting curve is very "reproducible," in the sense that the fluctuations of the curve from one random instance to the next are small (typically below 1%).

5.4. Interpretation of the SID Algorithm: Categories of Variables

Once SP has reached convergence, we can compute the total biases $\{W_i^{(\pm)}, W_i^{(0)}\}$. According to the previous discussion, these numbers should be interpreted as giving the fraction of constrained clusters where the variable x_i is respectively (frozen positive)/ (frozen negative)/ unconstrained. Having computed these weights, we may distinguish three reference types of variable nodes (of course all the intermediate cases will also be present): the **underconstrained** ones with $W_i^{(0)} \sim 1$, the **biased** ones with either $W_i^{(+)} \sim 1$ or $W_i^{(-)} \sim 1$, and the **balanced** ones with $W_i^{(+)} \simeq W_i^{(-)}$ and $W_i^{(0)}$ small.

Fixing a variable of each of these types produces different effects, consistently with the interpretation of the surveys. The following behaviors can be easily checked in numerical experiments:

Fixing a biased variable does not alter the structure of the clusters and the complexity changes smoothly (few constrained clusters are eliminated). This is the strategy used by the SID procedure.

Fixing an underconstrained variable has no effect on the complexity.

As expected, fixing a balanced variable produces a decrease very close to $\ln 2$ in the complexity.

5.5. A Summary of the Main Conjectures

The whole interpretation relies on the existence, in certain "hard SAT" regions of parameters (here in a window of α below α_c), of clusters of SAT assignments, which are very far apart

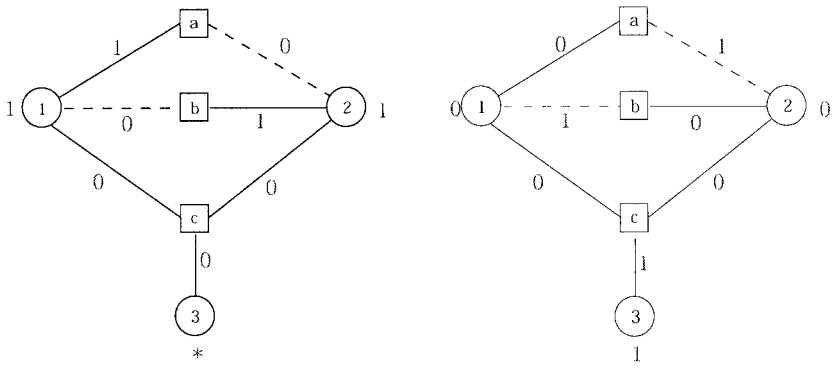

Fig. 6. A simple SAT problem with three variables, two 2-clauses, one 3-clause, and loops. Each figure gives one of the two possible solutions of warning propagation. The number on each edge (a, i) is the warning $u_{a\to i}$, the number next to each variable node is the corresponding generalized state of the variable in each of the two 'clusters'. The SP equations for this graph have an infinite number of solutions, with $\eta_{a\to 1} = \eta_{b\to 2} = x$, $\eta_{a\to 2} = \eta_{b\to 1} = y$, $\eta_{c\to 1} = \eta_{c\to 2} = 0$, and $\eta_{c\to 3} = (1-x)^2y^2/(1-xy)^2$.

(one cannot reach a cluster from another one unless one flips a finite fraction of the N variables). It would be very interesting to prove this statement.

When there exist such clusters, some of them may be "constrained clusters," in which some variables are constrained to take the same value for all the assignments of the cluster. The introduction of the don't-care state is an attempt at identifying all these clusters (in which each cluster is characterized by a single point in the enlarged assignment space including the don't-care state). SP, interpreted as BP in this enlarged assignment space, is an attempt at obtaining a statistical description of these constrained clusters.

Two conjectures arise naturally from the heuristic statistical physics approach and the numerical experiments on SP. They should hold for the random satisfiability problem, in the large N limit, in some hard SAT window of α just below α_c

- 1) With probability 1 (on the set of initial messages), SP converges to a unique set of fixed point messages.
- 2) These fixed point messages contain the correct information about the constrained clusters, and in particular the number of constrained clusters can be computed as in (39).

After the completion of this work, two groups have arrived to the conjecture that the "simple" clustering scenario described here should hold in the region $\alpha \in [4.15, \alpha_c \simeq 4.2667]$, and not hold outside this region [19, 27].

Note that for finite N, there are obvious counterexamples to these conjectures, as shown in Fig. 6.

6. COMMENTS AND PERSPECTIVES

When the solutions of a SAT (or more generally of a constrained satisfaction problem) tend to cluster into well-separated regions of the assignment space, it often becomes difficult to

find a global solution because (at least in all local methods), different parts of the problem tend to adopt locally optimal configurations corresponding to different clusters, and these cannot be merged to find a global solution. In SP we proceed in two steps. First we define some elementary messages which are warnings, characteristic of each cluster, then we use as the main message the surveys of these warnings. The warning that we used is a rather simplified object: It states which variables are constrained and which are not constrained. Because of this simplification the surveys can be handled easily, which makes the algorithm rather fast. As we saw in Section 5, one might also aim at a finer description of each cluster, where the elementary messages would give the probability that a variable is in a given state ($\in \{0, 1\}$), when considering the set of all SAT configurations inside this given cluster. In this case the survey will give the probability distribution of this probability distribution, when one chooses a cluster randomly. This is more cumbersome algorithmically, but there is no difficulty of principle in developing this extension. One could also think of generalizing further the messages (to probability distributions of probability distributions of probability distributions), if some problems have a structure of clusters within other clusters (this is known to happen for instance in spin glasses [20]), but the cost in terms of computer resources necessary to implement it might become prohibitive.

We have presented here a description of a new algorithmic strategy to handle the SAT problem. This is a rather general strategy which can also be applied in principle, to all Constraint Satisfaction Problems [24, 29]. At the moment the approach is very heuristic, although it is based on a rather detailed conjecture concerning the phase structure of random 3-SAT. The validity of a similar conjecture has been checked exactly in the random XOR-SAT problem. It would be interesting to study this algorithmic strategy in its own, independently from the random 3-SAT problem and the statistical physics study. One can expect progress on SP to be made in the future in various directions, among which: Rigorous results on convergence, different ways of using the information contained in the surveys, generalization of the algorithm [36] to deal with complex graphs which are not typical random graphs, use of the generalized SP with penalty to provide UNSAT certificates. The generalization of SP to generic Constraint Satisfaction Problems is discussed in a separate publication [4].

ACKNOWLEDGMENTS

We are indebted to G. Parisi and M. Weigt for very fruitful discussions. We also thank C. Borgs, J. Chayes, B. Hayes, S. Kirkpatrick, S. Mertens, O. Martin, P. Purdom, M. Safra, B. Selman, and J. Yedidia for stimulating exchanges. This work has been supported in part by the European Community's Human Potential Programme under Contract HPRN-CT-2002-00319, "STIPCO."

REFERENCES

[1] D. Achlioptas and Y. Perez, The threshold for random k-SAT is $2^k \log 2 - O(k)$ to appear in JAMS, Extended Abstract in STOC'03, 2003, pp. 223-231.

[2] G. Biroli, R. Monasson, and M. Weigt, European Phys J B 14 (2000), 551–568.

[3] A. Braunstein, M. Leone, F. Ricci-Tersenghi, and R. Zecchina. Complexity transitions in global algorithms for sparse linear systems over finite fields, J Phys A 35 (2002), 7559–7574.

[4] A. Braunstein, M. Mezard, M. Weigt, and R. Zecchina, Survey propagation for general constraint satisfaction problems, Volume on Computational Complexity and Statistical Physics, Oxford University Press, Santa Fe Institute Studies in the Sciences of Complexity, Santa Fe, NM, 2003, ArXiv: lanl/arXiv.org/cond-mat/0212451

[5] A. Braunstein and R. Zecchina, J Stat Mech Theory Experiment (JSTAT) (2004), p06007.

[6] S. Cocco, O. Dubois, J. Mandler, and R. Monasson, Rigorous decimation-based construction of ground pure states for spin glass models on random lattices, Phys Rev Lett 90 (2003), 047265.

[7] S. A. Cook and D. G. Mitchell, "Finding hard instances of the satisfiability problem: A survey, Satisfiability problem: Theory and applications, D.-Z. Du, J. Gu, and P. Pardalos (Editors), American Mathematical Society, Providence, RI, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 35. 1997.

[8] J. A. Crawford and L. D. Auton, Experimental results on the cross-over point in random 3-SAT, Artif Intell 81 (1996), 31–57.

[9] O. Dubois, Y. Boufkhad, and J. Mandler, Typical random 3-SAT formulae and the satisfiability threshold, Proc Eleventh Annu ACM-SIAM Symp Discrete Algorithms, 9–11 January 2000, San Francisco, CA, pp. 126–127.

[10] O. Dubois, R. Monasson, B. Selman, and R. Zecchina, (Editors), Phase transitions in combinatorial problems, Theoret Comput Sci 265 (2001).

[11] S. Franz and M. Leone, Replica bounds for optimization problems and diluted spin systems, J Stat Phys 111 (2003), 535.

[12] E. Friedgut, J Amer Math Soc 12 (1999), 1017–1054.

[13] R. G. Gallager, Low-density parity-check Codes, MIT Press, Cambridge, MA, 1963.

[14] H. O. Georgii, Gibbs measures and phase transitions, De Gruyter Studies in Mathematics Volume 9. de Gruyter, Berlin, 1988. (Russian edition: Mir, 1992).

[15] A. C. Kaporis, L. M. Kirousis, and E. G. Lalas, The probabilistic analysis of a greedy satisfiability algorithm, Proc 10th Annu European Symp Algorithms, Lecture Notes in Computer Science, Springer, London, 2002, pp. 574–585.

[16] S. Kirkpatrick and B. Selman, Critical behaviour in the satisfiability of random Boolean expressions, Science 264 (1994), 1297–1301.

[17] D. E. Knuth, The art of computer programming, Volume I: fundamental algorithms, Addison-Wesley, New York, 1968.

[18] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, Factor graphs and the sum-product algorithm, IEEE Trans Inform Theory 47 (2002), 498.

[19] S. Mertens, M. Mezard, and R. Zecchina, Threshold values for random K-SAT from the cavity method, preprint, 2003, http://arXiv.org/cs.CC/0309020

[20] M. Mézard, G. Parisi, and M. A. Virasoro, Spin glass theory and beyond, World Scientific, Singapore, 1987.

[21] M. Mézard and G. Parisi, The Bethe lattice spin glass revisited, Eur Phys J B 20 (2001), 217–233; The cavity method at zero temperature, J Stat Phys 111–125. (2003).

[22] M. Mézard, G. Parisi, and R. Zecchina, Science 297, (2002), 812–815. (Sciencexpress published online 27 June 2002; 10.1126/science.1073287)

[23] M. Mézard, F. Ricci-Tersenghi, and R. Zecchina, Alternative solutions to diluted p-spin models and XOR-SAT problems, J Stat Phys 111 (2003), 505-533.

[24] M. Mézard and R. Zecchina, Random 3-SAT: from an analytic solution to a new efficient algorithm, Phys Rev E 66 (2002), 056126.

[25] R. Monasson and R. Zecchina, Phys Rev E 56 (1997) 1357, Phys Rev Lett 76 (1996), 3881.

[26] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman and L. Troyansky, Nature (London) 400 (1999), 133–137.

[27] A. Montanari, G. Parisi, and F. Ricci-Tersenghi, Instability of one-step replica-symmetry-broken phase in satistiability problems, J Phys A 37 (2004), 2073.

[28] C. Moore and D. Achlioptas, Random k-SAT: Two moments suffice to cross a sharp threshold, preprint, extended abstract in FOCS'02, 2002, pp. 779-788.

[29] R. Mulet, A. Pagnani, M. Weigt, and R. Zecchina, Coloring random graphs, Phys Rev Lett 89 (2002), 268701.

[30] G. Parisi, On the survey-propagation equations for the random K-satisfiability problem, preprint, 2002, http://arXiv.org/cs.CC/0212009

[31] G. Parisi, A backtrack survey propagation algorithm for K-satisfiability, preprint, 2003, http://arXiv.org/cond-mat/0308510

[32] J. Pearl, Probabilistic reasoning in intelligent systems, 2nd edition, Morgan Kaufmann, San Francisco, CA, 1988.

[33] Satisfiability Library, www.satlib.org/

[34] B. Selman, H. Kautz, and B. Cohen, Local search strategies for satisfiability testing, Proc DIMACS, 1993, p. 661.

[35] M. Talagrand, Rigorous low temperature results for the p-spin mean field spin glass model, Probab Theory Related Fields 117, (2000), 303–360.

[36] J. S. Yedidia, W. T. Freeman and Y. Weiss, "Generalized belief propagation," Advances in neural information processing systems 13, T. K. Leen, T. G. Dietterich, and V. Tresp (Editors), 689–695. MIT Press, Cambridge, MA, 2001.

[37] R. Zecchina, ICTP, Trieste, Italy, www.ictp.trieste.it/˜zecchina/SP