# 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, Bât. 100, 91405 Orsay* CEDEX, *France*
[1] *The Abdus Salam International Centre for Theoretical Physics (ICTP), Str. Costiera 11, 34100 Trieste, Italy*

(Dated: 12th August 2005)

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 with $\alpha = M/N$ close to the experimental threshold $\alpha_c$ 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 $\alpha$ is close to (but smaller than) $\alpha_c$. We introduce a new type of message passing algorithm which allows to find efficiently a satisfiable assignment of the variables in the 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 are 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.

## I. INTRODUCTION

Combinatorial problems such as satisfiability are well known to be intractable in the worst case, yet many instances are surprisingly easy, even for naive heuristic algorithms. Work done over the past 10 years has shown that the hard instances of random K-SAT are concentrated in a narrow band near the critical value of the number of constraints per variable at which the problem becomes unsatisfiable [1, 2, 3, 4, 5].

More recently it has been possible to reach some understanding of what happens in the solution space of the problem as this threshold is approached[6, 7]. Well below the threshold, a generic problem has many solutions, which tend to form one giant cluster; the set of all satisfying assignments form 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). Hence greedy algorithms and other simple heuristics can readily identify a solution by a local search process.

Close to the critical threshold, however, the solution space breaks up into many smaller clusters. Solutions in separate clusters are generally far apart. What's worse, clusters that correspond to partial solutions -which satisfy some but not all of the constraints- are exponentially more numerous than the clusters of complete solutions and act as traps for local search algorithms.

We report work on an algorithm called survey propagation that efficiently finds solutions to many problem instances in this hard region, including instances too large for any earlier method. For example, for random 3-SAT we are able to solve instances close to the threshold, at a ratio of constraints per variable equal to 4.2, up to sizes of order $10^7$ variables; the computational time in this regime is found experimentally to scale roughly as $N \ln N$.

The method is a message passing procedure which resembles in some respects the iterative algorithm known as belief propagation[8], but with some crucial differences. The messages sent along the graph underlying the combinatorial problem are surveys of some elementary warning messages – probability distributions parametrized in a simple way – which describe how the single Boolean variables are expected to fluctuate from cluster to cluster. Once the iterative equations for such probability distributions have reached convergence, it is possible to identify the Boolean variables which can be safely fixed and to simplify the problem accordingly. The whole procedure is then repeated on the subproblem until a solution is found.

The method shows its power in the hard region of parameters where clustering occurs. This clustering phenomenon is particularly difficult to study in random systems, but recent progress in statistical physics has put forward some efficient heuristic methods like the cavity method[9], which allow for its quantitative analysis in a variety of cases. Turning this type of approach into a rigorous theory is an open subject of current research[10, 11]. In the case of highly symmetric combinatorial problems such as random K-XOR-SAT, the validity of the statistical physics analysis can be confirmed by rigorous studies [12, 13]. Moreover, for such problems there exist polynomial algorithms which have been shown [14] to undergo a dramatic change in memory requirement when clustering sets in.

The application of the cavity method to the random 3-SAT problem predicts a phase diagram with an intermediate phase exhibiting the clustering phenomenon[6, 7], and this theoretical analysis has suggested the development of the survey propagation algorithm put forward by Mézard and Zecchina in ref. [7]. 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 to address the optimization problem of minimizing the number of violated constraints in the UNSAT phase, but it will not be discussed here.

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: Sect. II describes the satisfiability problem and its graphical representation in terms of a factor graph. Sect. III explains a simple message procedure which propagates some warnings along the graph. This is an exact algorithm for tree factor graphs. In general, it is not a very good algorithm, but it is shown here because these warnings are the basic building blocks of our survey propagation algorithm. In sect. IV, we discuss the influence of loops and the clustering of solutions found from the statistical physics analysis of the random 3-SAT problem. Sect. V explains the survey propagation algorithm itself, and sect. VI shows how it can be used in a decimation procedure to get some satisfiable assignments in difficult problems. In sect.VII we give a general discussion of the difference between the survey propagation and the usual belief propagation algorithm. Sect. VIII contains a few comments.

The reader should not confuse the three different message passing procedures which are discussed in this paper: a warning propagation algorithm (WP), in which elementary $0,1$ messages are passed between nodes, in sect. III, our survey propagation (SP) in sect. V and the standard belief propagation (BP) in sect. VII.

## II. THE SAT PROBLEM AND ITS FACTOR GRAPH REPRESENTATION

We consider a satisfiability problem consisting of $N$ Boolean variables $\{x_i \in \{0,1\}\}$ ($\{0,1\} \equiv \{F,T\}$), with $i \in \{1,...,N\}$, with $M$ constraints. In the SAT problem 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 $i_1,...,i_K$ 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

$$(z_{i_1} \vee ... \vee z_{i_r} \vee ... \vee z_{i_K}) \tag{1}$$

where $z_{i_r} = x_{i_r}$ if $J^a_{i_r} = -1$ and $z_{i_r} = \bar{x}_{i_r}$ if $J^a_{i_r} = 1$. The problem is to find whether there exists an assignment of the $x_i \in \{0,1\}$ which is such that all the $M$ clauses are true. We define the total cost $C$ of a configuration $\mathbf{x} = (x_1,...,x_N)$ as the number of violated clauses.

In what follows we shall adopt the factor graph representation [15] 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 to a vertex in the graph, called a "variable node" (circles in the graphical representation), and each of the $M$ clauses is associated to 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. $J^a_i = -1$), a dashed line whenever the variable appearing in the clause is $\bar{x}_i$ (i.e. $J^a_i = 1$). 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" $\{J^i_a\}$ needed to define each function node. For the K-SAT problem where each clause contains $K$ variables, the connectivity of all the function nodes is $K$.

Throughout this paper, the variable nodes indices are taken in $i,j,k,...$, while the function nodes indices are taken in $a,b,c,...$. 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 connectivity 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). $V(i) \setminus b$ denotes the set V(i) without a node $b$. Similarly, for each function node $a$, 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 connectivity.

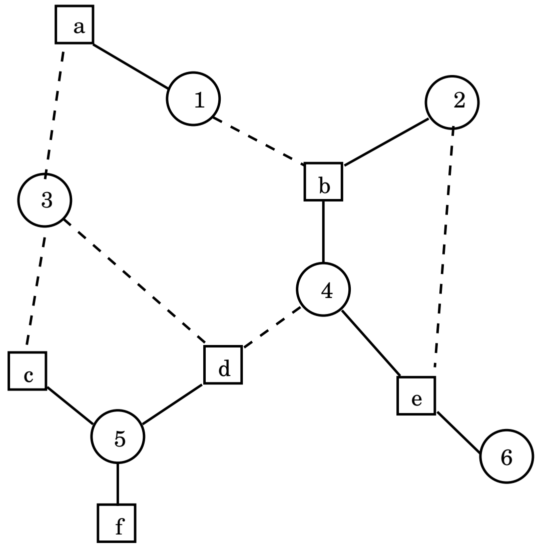


Figure 1: An example of a factor graph with 6 variable nodes $i = 1, .., 6$ and 6 function nodes $a, b, c, d, e, f$. The formula which is encoded is: $F = (x_1 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (\bar{x}_3 \vee x_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee x_5) \wedge (\bar{x}_2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

The same representation can be used for other combinatorial 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.

## III. 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, the problem can be easily solved by many methods. Here we shall describe a message passing solution. This solution uses elementary messages passed along the graph called cavity biases and cavity fields. It will be useful as a first step to develop the algorithm that we use for the general (non-tree) case, where we shall pass more elaborate messages which are interpreted as probability distributions of the elementary messages. Here we first define the messages, then give the rules to compute them, and finally show how this solves the SAT problem on a tree.

### A. Elementary messages: cavity biases and cavity fields

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 **cavity bias**.

The basic elementary message passed from one variable node $j$ to a function node $a$ (connected by an edge) is an integer number $h_{j\to a}$ called a **cavity field**.

In order to compute the cavity field $h_{j\to a}$, the variable $j$ considers the incoming cavity biases which it receives from all the function nodes $b$ to which it is connected, except $b = a$ (hence the name "cavity"), and performs the sum:

$$h_{j\to a} = \left( \sum_{b\in V_+(j)\setminus a} u_{b\to j} \right) - \left( \sum_{b\in V_-(j)\setminus a} u_{b\to j} \right) \tag{2}$$

If $j$ has no other neighbor than $a$, then $h_{j\to a} = 0$.

In order to compute the cavity bias $u_{a\to i}$, the function node $a$ considers the incoming cavity fields which it receives from all the variable nodes $j$ to which it is connected, except $j = i$. If there is at least one $j \in V(a) \setminus i$ such that $h_{j\to a} J_j^a \le 0$, then $u_{a\to i} = 0$, otherwise $u_{a\to i} = 1$. If $a$ has no other neighbor than $i$, then $u_{a\to i} = 1$. Therefore:

$$u_{a\to i} = \prod_{j\in V(a)\setminus i} \theta\left(h_{j\to a} J_j^a\right) \ , \tag{3}$$

where $\theta(x)$ is a step function taking values $\theta(x) = 0$ for $x \leq 0$ and $\theta(x) = 1$ for $x > 0$.

Given a function node $a$ and one of its variables nodes $i$ , then the cavity fields $h_{j\to a}$ $(j \neq i)$ can be viewed as inputs and $u_{a\to i}$ as the output. The interpretation of the messages and the message-passing procedure is the following. A cavity bias $u_{a\to i} = 1$ can be interpreted as a warning sent from function node $a$, telling the variable $i$ that it should adopt the correct value for satisfying the 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 $h_{j\to a} J_j^a > 0$, 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 $j \in V(a) \setminus i$ are in this situation, then $a$ sends a warning to $i$.

### B. Iteration dynamics: warning propagation

The above message passing procedure can be tried for any SAT problem. The typical iteration dynamics uses the cavity biases as basic messages. They are initialized to random values (for instance $u_{a\to i} = 0, 1$ with probability 1/2). Then at each iteration one sweeps the set of edges in a random order, and updates all the corresponding cavity biases.

If the process converges, this dynamics defines a fixed set of cavity biases $u^*_{a\to i}$. These can be used to compute, for each variable $i$, the "local field" $H_i$ and the "contradiction number" $c_i$ which are two integers defined as:

$$H_i = \sum_{b\in V_+(i)} u^*_{b\to i} - \sum_{b\in V_-(i)} u^*_{b\to i} \tag{4}$$

$$\begin{aligned} c_i &= 1 \quad \text{if} \quad \left(\sum_{b\in V_+(i)} u^*_{b\to i}\right)\left(\sum_{b\in V_-(i)} u^*_{b\to i}\right) > 0 & (5)\\ &= 0 \quad \text{otherwise.} & (6)\end{aligned}$$

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.

This warning passing (WP) procedure is a kind of adaptation to the SAT problem of the "Min-Sum" algorithm, which is itself a limiting case of the well known "Sum-Product" algorithm, widely used in various contexts for evaluating marginal probabilities of interacting discrete variables over graphs (see [15] for a review and a detailed list of references to previous works). A detailed comparison between our algorithms and the Sum-Product algorithm is contained in sect. VII. The WP algorithm can be summarized as:

**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 cavity biases` $u^*_{a\to i}$.

0. `At time` $t = 0$: `For every edge` $a \to i$ `of the factor graph, randomly initialize the cavity bias` $u_{a\to i}(t = 0) \in \{0, 1\}$
1. `For` $t = 1$ `to` $t = t_{max}$:
   - 1.1 `sweep the set of edges in a random order, and update sequentially the cavity biases on all the edges of the graph, generating the values` $u_{a\to i}(t)$, `using subroutine 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 cavity biases` $u^*_{a\to i} = u_{a\to i}(t)$

`Subroutine UPDATE(`$u_{a\to i}$`).`
`INPUT: Set of all cavity biases arriving onto each variable node` $j \in V(a) \setminus i$
`OUTPUT: new value for the cavity bias` $u_{a\to i}$.

1 `For every` $j \in V(a) \setminus i$, `compute the cavity field` $h_{j\to a} = \left(\sum_{b\in V_+(j)\setminus a} u_{b\to j}\right) - \left(\sum_{b\in V_-(j)\setminus a} u_{b\to j}\right)$

2 Using these cavity fields $h_{j\to a}$, compute the cavity bias $u_{a\to i} = \prod_{j\in V(a)\setminus i} \theta\left(h_{j\to a} J_j^a\right)$

An example of the use of WP is shown in fig. 2. The interest in WP largely comes from the fact that it gives the exact solution for problems where the factor graph is a tree. For the SAT case, this is summarized in the following simple theorem:

**THEOREM:**

*Consider an instance of the SAT problem for which the factor graph is a tree. Then the WP algorithm converges to a unique set of fixed point cavity biases $u^*_{a\to i}$, independently on the initial cavity biases. If at least one of the corresponding contradiction numbers $c_i$ is equal to 1, the problem is UNSAT, otherwise it is SAT.*

**Consequences:** 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. When $H_i \neq 0$, the variable $i$ is constrained. One should fix $x_i = 1$ when $H_i > 0$, and $x_i = 0$ when $H_i < 0$. A variable with $H_i = 0$ is under-constrained, which means that there exists satisfiable assignments where it is 1 and others where it is 0. In order to complete the determination of a satisfiable assignment, one can:
i) fix all constrained variables and eliminate all clauses which are satisfied by this choice and simplify the rest of the formula.
ii) choose randomly one under-constrained variable, fix it to one value $(0,1)$. Eliminate all clauses which are satisfied by this choice and simplify the rest of the formula.
iii) run again the WP algorithm on the new instance, and iterate the whole process.

This algorithm for determining a satisfiable assignment, exact for problems with a tree factor graph, which we call "Warning Inspired Decimation" or WID, is summarized as follows.

**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. (4,6).
2. If there is at least one contradiction number $c_i = 1$, return UNSAT. Else:
   - 2.1 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:
   - 2.2 Choose one unfixed variable, fix it to an arbitrary value, clean the graph. GOTO label 1

- return the set of assignments for all the variables.

**PROOF** of the theorem:

The convergence of WP on tree graphs is a well known result (see e.g.[15]). We give here an elementary proof of convergence for the specific case of SAT, 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 connectivity 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 $\mathcal{T}_{a-i}$ 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 $\mathcal{T}_{a-i}$ (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 cavity bias $u_{a\to i}$ at level $r$ is fully determined from the knowledge of all the cavity biases $u_{b\to j}$ at levels $\leq r-2$. Therefore the cavity bias $u_{a\to i}(t)$ 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**:

*If a cavity bias $u^*_{a\to i} = 1$, 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 (3), this means that for all variable nodes $j \in V(a)\setminus i$, the node $j$

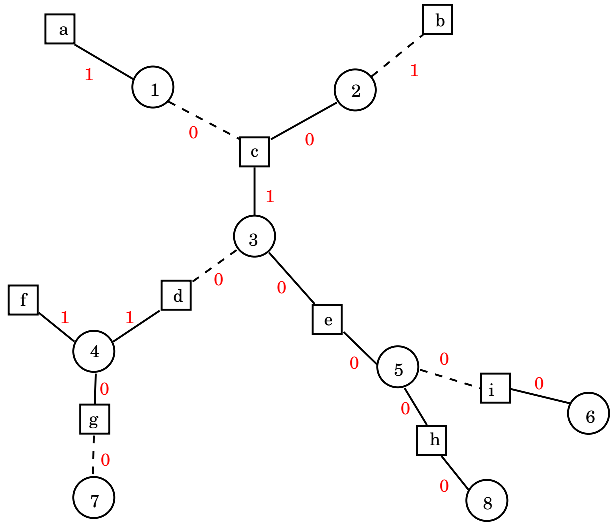


Figure 2: An example of result obtained by the WP algorithm on a tree problem with $N = 8$ variables and $M = 9$ clauses. The number on each edge of the graph is the value of the corresponding cavity bias $u^*$. The local fields on the variable are thus: $1, -1, 1, 2, 0, 0, 0, 0$. The satisfiable assignments 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 BID 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.

receives at least one message from a neighboring factor node $b \in V(j) \setminus a$ 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 $J_j^b = -1$, or 1 (False) if $J_j^b = 1$: we shall say that clause $b$ fixes the value of variable $j$. This is true for all $j \in V(a) \setminus i$, which means that the reduced problem on $\mathcal{T}_{a-i}$ 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 (6) 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 cavity bias. So on the new graph, all the edges have $u^* = 0$. Following the step 2.2 of BID, one chooses randomly a variable $i$, one 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 are not satisfied by the choice $x_i$ may send some $u^* = 1$ messages (this happens if such a clause had connectivity 2 before fixing variable $i$). However, running WP on the corresponding subgraph $\mathcal{T}_{a-i}$, the set of cavity biases can not 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 BID has generated a strictly smaller graph with no contradiction. By induction, it thus finds a SAT assignment.$\square$

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

## IV. LOOPS AND CLUSTERING OF SOLUTIONS: THE CASE OF RANDOM 3-SAT

One can try to use the WP algorithm of the previous section in the case where the factor graph contains loops. In some cases it may work as before, but very often the iteration does not converge. A stationary solution which is always present if there are no unit clause is the trivial one with all messages zero.

The case of randomly generated 3-SAT formulas is a good test-bed for checking the behavior of algorithms. We take $M = \alpha N$ and generate clauses by choosing triplets of variables at random with uniform probability. The $J$s are set to $\pm 1$ with equal probability. The associated factor graph has the following properties. For each triplet $i < j < k$ of variable nodes, a function node connecting them is present with a probability $6\alpha/N^2$. For large $N$, the connectivities $n(i)$ of variable nodes become independent identically distributed (iid) random variables with a Poisson distribution of mean $3\alpha$.

As discussed in refs. [6, 7], in the large $N$ limit there exist two critical values of $\alpha$. For $\alpha < \alpha_d \simeq 3.921$, the set of all satisfying assignments $S_M$ is connected, that is one can find a path in $S_M$ to go from any assignment to any other assignment requiring short steps only (in Hamming distance). No variables are constrained to take the same value in all satisfying assignments and a solution can be found by simple greedy algorithms[6, 7]. For $\alpha_d < \alpha < \alpha_c \simeq 4.267$, $S_M$ becomes divided into subsets which are far apart in Hamming distance and which are composed of the same number of solutions (up to sub-exponential corrections). We denote with $\mathcal{N}_{\text{cl}} \equiv \exp(\Sigma(\alpha))$ the number of such clusters and with $\mathcal{N}_{\text{int}} \equiv \exp(S_{\text{int}}(\alpha))$ the number of solutions within each clusters. In the language of statistical physics, the quantity $\Sigma$ is called the **complexity** and $S_{\text{int}}$ the internal **entropy**. Although there exists in this phase an exponentially large number of clusters, each containing an exponentially large number of solutions, it is very difficult to find one solution because of the proliferation of 'metastable' clusters. A metastable cluster is a cluster of assignments which all have the same fixed number $C$ of violated clauses, and such that one cannot find at a small Hamming distance of this cluster an assignment which violates strictly less than $C$ clauses.

The metastable clusters with $C > 0$ are exponentially (in $N$) more numerous than the satisfiable clusters. One can then expect that local search algorithms will generically get trapped in these much more numerous metastable clusters. We call the range $\alpha_d < \alpha < \alpha_c$ the **hard-sat** region. At $\alpha = \alpha_c$ the complexity vanishes ($\lim_{N\to\infty} \Sigma(\alpha_c)/N = 0$) and for $\alpha > \alpha_c$, the instances are almost always unsatisfiable.

On the algorithmic side, random 3-SAT instances generated just below the threshold $\alpha_c$ are used as benchmarks for ranking the efficiency of search algorithms. One should notice that the proliferation of metastable clusters in the hard SAT phase can be seen only for $N$ large enough. Indeed, it has been found for instance that at $\alpha = 4.2$, the complexity of the most numerous metastable states is typically $\Sigma(4.2) \sim .003N$. The smallness of this number implies that the effects of proliferation will be seen only for $N$ larger than 1000 or so.

The next section explains an algorithm which is well suited for finding SAT assignment in the hard-SAT phase.

## V. FROM WARNING PROPAGATION TO SURVEY PROPAGATION

From the analytical and numerical studies [6, 7] of random 3-SAT in the hard sat region, one learns the geometrical reasons for the failure of the simple (BP) message-passing procedure: in the low $\alpha$ regime it works correctly in the sense that it correctly predicts all messages zero. All variables are under-constrained (or, in the physics analogy, "unfrozen"). However, in the hard sat region $\alpha \in [3.921, 4.267]$ the only stationary solution which is found in practice continues to be the one in which all cavity biases are zero: the simple iteration rule has very small probability to find a set of cavity biases corresponding to a given cluster. The reason can be qualitatively understood as follows: while the local iteration rule in some region of the graph will tend to generate cavity-biases polarized in the direction of one cluster, other regions of the graph will get polarized in some other directions corresponding to different clusters. This basic incapability of the 'elementary message' passing equations to describe the clustering phenomenon is probably common to many hard combinatorial problems. A generalization of the elementary BP message passing protocol is necessary in order to deal with it.

Taking the clustering scenario as a working hypothesis, we can define the probability distribution of messages with respect to the cluster-to-cluster fluctuations. We may then study the evolution under the cavity steps of the full histograms of message and make use of the whole set of such $3M$ surveys to find satisfying assignments.

### A. Definition of surveys

We define the **survey** of the cavity-bias $u_{a\to i}$ as:

$$Q_{a\to i}(u) = \frac{1}{\mathcal{N}_{\text{cl}}} \sum_{\ell} \delta(u, u^{\ell}_{a\to i}) \tag{7}$$

where $\ell$ runs over all clusters of solutions and $u^{\ell}_{a\to i}$ is the cavity bias over the edge $a \to i$ in the cluster $\ell$. $\delta$ denotes the Kronecker delta function over discrete variables, $\delta(x, y) = 1$ if $x = y$ and zero otherwise.

Note that $u \in \{0, 1\}$, and therefore the cavity-bias-survey $Q_{a\to i}(u)$, which is a probability distribution, is parametrized by a single number. We shall write:

$$Q_{a\to i}(u) = (1 - \eta_{a\to i})\delta(u, 0) + \eta_{a\to i}\delta(u, 1) \tag{8}$$

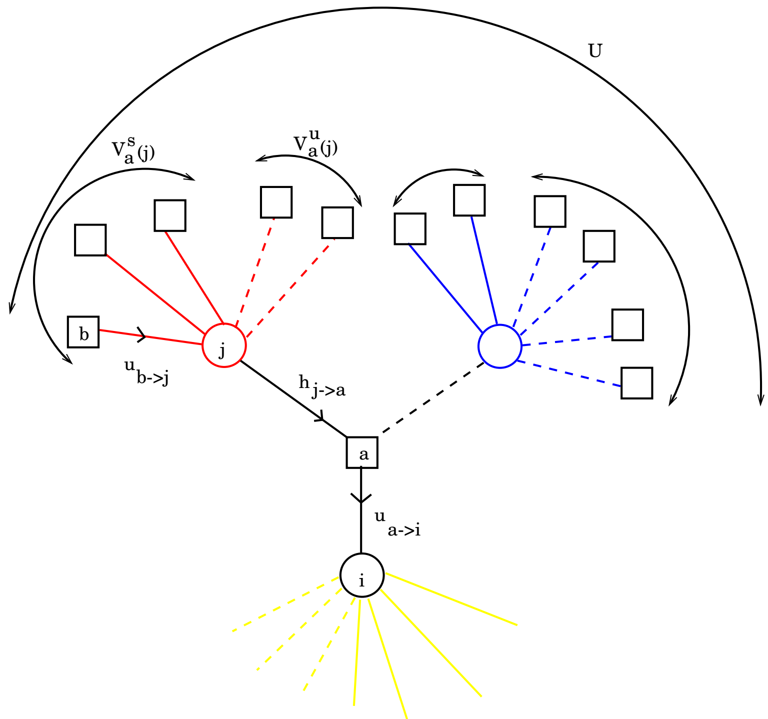


Figure 3: A function node $a$ and its neighborhood. The survey of the cavity-bias $u_{a\to i}$ can be computed from the knowledge of the joint probability distribution for all the cavity-biases in the set $U$, i.e. those coming onto all variable nodes $j$ neighbors of $a$ (except $j = i$).

In general for any set of cavity biases, say $u, v, w, ..$ we can define similarly the survey $Q(u, v, w, ..)$ as the joint probability that, in a randomly chosen cluster, these cavity biases take value $u, v, w, ...$

### B. Survey iteration: the general case

We want to write an iterative functional equation for the surveys. Consider an edge $a - i$ of the graph (see fig.3). The condition for the cavity bias $u^\ell_{a\to i}$ to be 1 has been written in (3). It depends on all the cavity biases $u^\ell_{b\to j}$, with $j \in V(a) \setminus i$ and, for each such $j$, $b \in V(j) \setminus a$ (see fig.3). We shall denote collectively by $U = \{u^\ell_{b\to j}\}$ the set of all these variables, and by $Q(U)$ their survey. From (3), the probability $\eta_{a\to i}$ that $u_{a\to i} = 1$ is equal to the probability that: $\forall j \in V(a) \setminus i: \; h_{j\to a} J^a_j > 0$.

Let us now study the probability $\rho_{j\to a}$ that the variable $J^a_j h_{j\to a}$ is strictly positive. For simplicity, we first suppose that $J^a_j = 1$. Then $\rho_{j\to a}$ is the probability that, among all cavity biases $u_{b\to j}, b \in V_+(j)$, at least one of them is a warning (with $u = 1$). Notice that, because $Q(U)$ is a survey over SAT assignments, the contradiction number of variable $j$ must be zero and therefore the cavity biases $u_{b\to j}$ with $b \in V_-(j) \setminus a$, will be automatically equal to zero. The result can be written as:

$$\rho_{j\to a} = \sum_U Q(U)\,\theta\left(\sum_{b\in V_+(j)\setminus a} u_{b\to j}\right) . \tag{9}$$

If $J^a_j = -1$, the roles of $V_+$ and $V_-$ in the above equation are exchanged. It is convenient to define the two sets: $V^u_a(j)$ and $V^s_a(j)$, 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.3):

$$\text{if} \quad J^a_j = 1 : \quad V^u_a(j) = V_+(j) \;\; ; \;\; V^s_a(j) = V_-(j) \setminus a \tag{10}$$

$$\text{if} \quad J^a_j = -1 : \quad V^u_a(j) = V_-(j) \;\; ; \;\; V^s_a(j) = V_+(j) \setminus a \tag{11}$$

The final result for $\eta_{a\to i}$ can then be written as:

$$\eta_{a\to i} = \sum_U Q\left(U\right) \prod_{j\in V(a)\setminus i} \theta\left(\sum_{b\in V_a^u(j)} u_{b\to j}\right) \ . \tag{12}$$

### C. Factorized form: propagation of cavity-bias-surveys

While the previous iteration equation (12) is exact, it is useless, because it relates the survey $Q_{a\to i}(u)$ to the joint survey $Q\left(U\right)$ of all cavity biases arriving onto the variable nodes neighbors of $a$ in the graph $\mathcal{T}_{a-i}$. This will itself depend on the joint probability of a larger number of cavity biases and so forth.

In order to have a closed set of equations for the $3M$ cavity-bias-surveys, we shall make the assumption that the messages that are injected into the function node are uncorrelated, that is that their joint probability distribution is factorized. This is exact on a tree, one can hope that it will be a good approximation for factor graphs with a locally tree-like structure, but in general it is an approximation that we cannot control easily, although we shall come back to this point later on.

Within the factorized approximation we write the propagation rule for the cavity bias surveys as:

$$\eta_{a\to i} = \prod_{j\in V(a)\setminus i} \rho_{j\to a} \ , \tag{13}$$

with

$$\rho_{j\to a} = C_{j\to a} \sum_{\{u_{b\to j}\}} \left[\prod_{b\in V(j)\setminus a} Q_{b\to j}(u_{b\to j})\right] \theta\left(\sum_{b\in V_a^u(j)} u_{b\to j}\right) \prod_{b\in V_a^s(j)} \delta\left(u_{b\to j},0\right) \ . \tag{14}$$

where $\sum_{\{u_{b\to j}\}}$ is a notation for the sum over all the cavity biases arriving onto all the variables $j\in V(a)\setminus i$ from function nodes $b$ different from $a$ (one could write more explicitly $\sum_{\{u_{b\to j}\}} = \left(\prod_{b\in V(j)\setminus a}\left[\sum_{u_{b\to j}\in\{0,1\}}\right]\right)$). With respect to the general iteration rule (12), we have used a factorized form for the $Q(U)$ probability, and we add two modifications.

**i)** We have explicitly written the constraint that all the cavity biases $u_{b\to j}, b\in V_a^s(j)$ are equal to zero. This constraint was automatic when the full distribution $Q(U)$ was used. In the factorized approximation, the constraint has to be implemented explicitly in order to cut out the contradictory messages: it is a necessary condition for the contradiction number on variable $j$ to vanish.

**ii)** Because of this extra constraint, a normalization constant $C_{j\to a}$ must be computed, as follows. The non contradictory messages can be divided into three classes. **Class u**: those with at least one $u_{b\to j}, b\in V_a^u(j)$ equal to 1 (these messages are the ones which contribute to $\rho$). **Class s**: those with at least one $u_{b\to j}, b\in V_a^s(j)$ equal to 1. **Class 0**: those with all $u_{b\to j}, b\in V_a^u(j)$ equal to 0. Let us compute the weight of each class, $\Pi_{j\to a}^{u,s,0}$:

$$\begin{aligned}
\Pi_{j\to a}^u &= \sum_{\{u_{b\to j}\}} \left[\prod_{b\in V(j)\setminus a} Q_{b\to j}(u_{b\to j})\right] \theta\left(\sum_{b\in V_a^u(j)} u_{b\to j}\right) \prod_{b\in V_a^s(j)} \delta\left(u_{b\to j},0\right)\\
\Pi_{j\to a}^s &= \sum_{\{u_{b\to j}\}} \left[\prod_{b\in V(j)\setminus a} Q_{b\to j}(u_{b\to j})\right] \theta\left(\sum_{b\in V_a^s(j)} u_{b\to j}\right) \prod_{b\in V_a^u(j)} \delta\left(u_{b\to j},0\right)\\
\Pi_{j\to a}^0 &= \sum_{\{u_{b\to j}\}} \left[\prod_{b\in V(j)\setminus a} Q_{b\to j}(u_{b\to j})\right] \left[\prod_{b\in V(j)\setminus a} \delta\left(u_{b\to j},0\right)\right] \ .
\end{aligned} \tag{15}$$

Then the normalization constant in (14) is:

$$C_{j\to a} = 1/(\Pi_{j\to a}^u + \Pi_{j\to a}^s + \Pi_{j\to a}^0) \ . \tag{16}$$

The equation (14) together with the normalization (16) provide a closed set of equations for the numbers $\eta_{a\to i}$ which characterize the cavity-bias-surveys (we remind that $\eta_{a\to i}$ is the probability that the cavity-bias $u_{a\to i}=1$).

### D. The survey propagation algorithm

For clarity and for algorithmic use we now write the survey propagation (SP) iteration equations explicitly in terms of the $\eta$ variables defined in (8):

$$\eta_{a\to i} = \prod_{j\in V(a)\setminus i}\left[\frac{\Pi^u_{j\to a}}{\Pi^u_{j\to a}+\Pi^s_{j\to a}+\Pi^0_{j\to a}}\right] , \tag{17}$$

where:

$$\begin{aligned}
\Pi^u_{j\to a} &= \left[1-\prod_{b\in V^u_a(j)}(1-\eta_{b\to j})\right]\prod_{b\in V^s_a(j)}(1-\eta_{b\to j}) \\
\Pi^s_{j\to a} &= \left[1-\prod_{b\in V^s_a(j)}(1-\eta_{b\to j})\right]\prod_{b\in V^u_a(j)}(1-\eta_{b\to j}) \\
\Pi^0_{j\to a} &= \prod_{b\in V(j)\setminus a}(1-\eta_{b\to j})
\end{aligned} \tag{18}$$

Note that a product like $\prod_{b\in V^s_a(j)}(1-\eta_{b\to j})$ gives the probability that no warning arrives on $j$ from the function nodes $b\in V^s_a(j)$; if the set $V^s_a(j)$ is empty, the product takes value 1 by definition. Also, if $V(a)\setminus i$ is empty, then $\eta_{a\to i}=1$. Finally, the SP algorithm can be summarized as follows.

**SP** `algorithm:`

`INPUT: the factor graph of a Boolean formula in conjunctive normal form; a maximal number of iterations` $t_{max}$`, a requested precision` $\epsilon$.

`OUTPUT: UN-CONVERGED if SP has not converged after` $t_{max}$ `sweeps. If it has converged: the set of all cavity-bias-surveys` $\eta^*_{a\to i}$.

0. `At time` $t=0$`: For every edge` $a\to i$ `of the factor graph, randomly initialize the cavity bias` $\eta_{a\to i}(t=0)\in[0,1]$
1. `For` $t=1$ `to` $t=t_{max}$:
   - 1.1 `sweep the set of edges in a random order, and update sequentially the cavity bias surveys on all the edges of the graph, generating the values` $\eta_{a\to i}(t)$`, using subroutine CBS-UPDATE.`
   - 1.2 `If` $|\eta_{a\to i}(t)-\eta_{a\to i}(t-1)|<\epsilon$ `on all the edges, the iteration has converged and generated` $\eta^*_{a\to i}=\eta_{a\to i}(t)$`: GOTO label 2.`
2. `If` $t=t_{max}$ `return UN-CONVERGED. If` $t<t_{max}$ `return the set of fixed point cavity bias surveys` $\eta^*_{a\to i}=\eta_{a\to i}(t)$

`Subroutine CBS-UPDATE(`$\eta_{a\to i}$`).`

`INPUT: Set of all cavity bias surveys arriving onto each variable node` $j\in V(a)\setminus i$

`OUTPUT: new value for the cavity bias survey` $\eta_{a\to i}$.

1. `For every` $j\in V(a)\setminus i$`, compute the values of` $\Pi^u_{j\to a},\Pi^s_{j\to a},\Pi^0_{j\to a}$ `using eq. (18).`
2. `Compute` $\eta_{a\to i}$ `using eq. (17).`

### E. Total bias and complexity

Suppose that SP has converged and gives a set of cavity bias surveys $\eta^*_{a\to i}$. This can be used to estimate the properties (the total bias) of each variable. The probability, when one picks up a cluster at random, that the variable $i$ is constrained ("frozen") to $x_i=1$ is given by its *positive bias*

$$W_i^{(+)} = \text{Prob}\left(\sum_{a\in V(i)} u_{a\to i}>0\right) . \tag{19}$$

Within the factorized approximation, this reads:

$$W_i^{(+)} = C_i \sum_{\{u_{a\to i}\}, a\in V(i)} \left[ \prod_{a\in V(i)} Q_{a\to i}(u_{a\to i}) \right] \theta \left( \sum_{a\in V_+(i)} u_{a\to i} \right) \prod_{a\in V_-(i)} \delta\left(u_{a\to i}, 0\right). \tag{20}$$

In order to write the bias in terms of the converged weights $\{\eta^*_{a\to i}\}$ given by SP it is convenient to introduce the following quantities:

$$\begin{aligned} \hat{\Pi}_i^+ &= \left[ 1 - \prod_{a\in V_+(i)} (1-\eta^*_{a\to i}) \right] \prod_{a\in V_-(i)} (1-\eta^*_{a\to i}) \\ \hat{\Pi}_i^- &= \left[ 1 - \prod_{a\in V_-(i)} (1-\eta^*_{a\to i}) \right] \prod_{a\in V_+(i)} (1-\eta^*_{a\to i}) \\ \hat{\Pi}_i^0 &= \prod_{a\in V(i)} (1-\eta^*_{a\to i}) \end{aligned} \tag{21}$$

The positive bias then reads:

$$W_i^{(+)} = \frac{\hat{\Pi}_i^+}{\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0}\,. \tag{22}$$

Similarly, the probability, when one picks up a cluster at random, that the variable $i$ is frozen to $x_i = 0$ is given by the negative bias

$$W_i^{(-)} = \frac{\hat{\Pi}_i^-}{\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0}\,. \tag{23}$$

The probability to pick up a cluster in which a variable is under-constrained or unfrozen reads

$$W_i^{(0)} = 1 - W_i^{(+)} - W_i^{(-)}. \tag{24}$$

We may also estimate the complexity, that is the normalized logarithm of the number of satisfying clusters. In this section we give the general result for the complexity obtained from the statistical physics analysis of ref. [7]; a more direct justification for this expression is given in sect. VII. The total complexity $\Sigma$ can be decomposed into contribution associated with every function node and with every variable, and reads:

$$\Sigma = \left( \sum_{a=1}^{M} \Sigma_a - \sum_{i=1}^{N} (n_i - 1)\Sigma_i \right) \tag{25}$$

where

$$\Sigma_a = +\log \left[ \prod_{j\in V(a)} \left( \Pi^u_{j\to a} + \Pi^s_{j\to a} + \Pi^0_{j\to a} \right) - \prod_{j\in V(a)} \Pi^u_{j\to a} \right] \tag{26}$$

$$\Sigma_i = +\log \left[ \hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0 \right] \tag{27}$$

In Fig. 4 we report the data for the complexity of a given random 3-SAT formula of size $N = 10^6$ and $\alpha$ in the clustering range. Note that we add clauses to the pre-existing formula and therefore the data for the complexity are continuous.

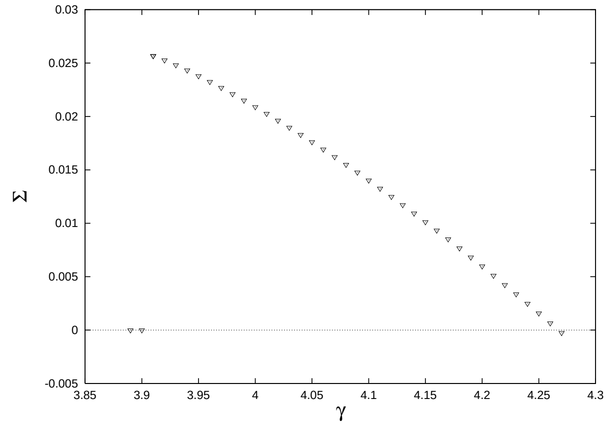


Figure 4: Complexity per variable ($\Sigma/N$) of the satisfying clusters for a given sample of size $N = 10^6$. The complexity vanishes at $\alpha_c$ which is expected to be the critical threshold for the specific formula under study.

### F. Discussion of survey propagation

When one runs the SP algorithm on a satisfiability problem with a tree factor-graph, one may encounter two situations: 1) if the problem is SAT, and the SP converges to values of $\eta_{a\to i} \in \{0, 1\}$. These values are exactly equal to the warnings found in the WP algorithm of section III: $\forall a, i :\ \eta_{a\to i} = u^*_{a\to i}$. This result is easily shown as follows: one first shows that $\eta_{a\to i} = u^*_{a\to i}$ for links $a - i$ at level 0 or 1. Then an inspection of all cases shows that the rule (17,18), when specialized to the cases where $\eta \in \{0, 1\}$, is equivalent to (3), provided there is no contradiction in the messages arriving onto function node $a$. 2) If the problem is UNSAT and there is a contradiction, either SP runs into trouble because (17,18) gives a 0/0 result, or the estimation of $W_i^\pm$ in (22,23) gives a 0/0 result (depending on whether the contradiction appears at the level of the cavity messages or only at the final stage: this last situation typically occurs when a variable belongs only to two clauses). The generalization of SP to unsat problems, i.e. to the MAX-K-SAT problem, has been discussed in [7] and we shall not explain it here. So SP does not bring any new result with respect to simple warning passing for a tree problem.

SP is a very heuristic algorithm, for which we have no proof of convergence, but we have found experimentally that it does converge in situations where the simple WP algorithm does not. When it converges, SP provides a full set of cavity bias surveys and an estimate for the complexity, or the number of SAT clusters, in the specific instance under study. Such notions have been defined in the large $N$ limit and what is the rigorous meaning of these quantities for finite $N$ in general is an open question. The validity of the factorization approximation is difficult to assess. Even the notion of cluster is not easy to define for finite $N$. Roughly speaking one can think that for large $N$, there might exist some effective 'finite $N$ clusters', such that the number of variables to flip in order to reach one cluster from another one is large, leading to a separation of scales in the number of variables involved between the intra-cluster moves and the inter-cluster moves. Such a situation would generally be difficult to handle for search algorithms, and this is where SP turns out to be quite effective.

In order to get some concrete understanding of these questions for large but finite $N$, we have experimented SP on single instances of the random 3-SAT problem with many variables, up to $N \sim 10^7$. The results are summarized as:

- **The under-constrained phase:** For $\alpha < \alpha_d$, SP converges toward the trivial solution $Q_{a\to i}(u) = \delta(u)$, for all $a \to i$ edges. All variables are under-constrained and there exist one single cluster[26].

- **The hard sat phase:** For $\alpha_d < \alpha < \alpha_c$, (where $\alpha_d \simeq 3.921$ and $\alpha_c \simeq 4.267$), that is in the hard sat region, SP converges to a unique non-trivial solution. The complexity is positive (exponential number of clusters). The surveys carry precise information concerning the space of solutions for one given sample. Working with $N$ not too large, $N \leq 10000$, some SAT configurations can be found efficiently by good algorithms like e.g. walksat-35 [18, 19]. We have used this to test the outcome of SP by the following method. For one given sample, we run SP, and we compute for each variable its 'magnetization' $m_i$ defined as $m_i = W_i^{(+)} - W_i^{(-)}$. We have generated with walksat-35 a large number $\mathcal{N}$ of satisfiable assignments, each new assignment being found from an independent random initial condition. In this set we measure, for each variable $i$, the number of assignments $\mathcal{N}_i^+$ where $x_i = 1$, and we define $m'_i = 2\mathcal{N}_i^+/\mathcal{N} - 1$. If we suppose that on average, a variable $i$ which is under-constrained in a cluster appears with equal probability as $x_i = 0$ or $x_i = 1$ in the SAT assignments of this cluster, then one can expect that $m_i = m'_i$. Indeed, extended numerical experiments (see ref. [7]) display a clear correlation between $m$ and $m'$, indicating that SP is able to predict correctly the fraction of SAT assignments where a variable $i$ has $x_i = 1$.

- **The UNSAT phase:** For $\alpha > \alpha_c = 4.267$ the version of SP explained above does not give useful information. For $\alpha_c < \alpha < \alpha_* \sim 4.36$ the equations converge and yet the complexity is negative. For $\alpha > \alpha_*$ the equations even cease to converge[20]. A more general formulation of SP, in which one penalizes contradictory messages instead of giving them a zero probability, can deal with such a regime and provide relevant information concerning the low-cost configuration, that is about the MAX-3-SAT problem (see [7]).

## VI. NEW SEARCH ALGORITHMS BASED ON SURVEY PROPAGATION

The previous section has shown how SP can give rather precise answers on the statistical properties of $S_M$ for a given instance of 3-SAT. The natural question to ask is how to use such information to find satisfying assignments.

If SP could predict with very high accuracy the value of the complexity of a given formula, it could also predict its satisfiability. Then one could proceed in finding a satisfying assignment just by the usual conversion of decision algorithms into a search algorithms. A variable is selected and fixed to one value. Then SP is used to evaluate the complexity of the subproblem of size $N-1$. If the subproblem is SAT ($\Sigma > 0$) then we keep the assignment, otherwise the opposite value of the binary variable is chosen. The process is repeated until all the variables have been exhausted (in at most $2N$ calls of SP). The above scheme, however, suffers from finite size effects and from the imprecision in the determination of the complexity, a fact which is particularly important close to $\alpha_c$. Moreover it does not take advantage of the complete information provided by the surveys, which is used in the procedure that we now describe.

### A. Categories of variables

Once SP has reached convergence, we can compute the total biases $\{W_i^{(\pm)}, W_i^{(0)}\}$ which tell the fraction of clusters where the variable $x_i$ is frozen positive/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 **under-constrained** ones with $W_i^{(0)} \sim 1$, the **biased** ones with $W_i^{(+)} \sim 1$ or $W_i^{(-)} \sim 1$ and the **balanced** ones with $W_i^{(+)} \simeq W_i^{(-)}$ and $W_i^{(0)}$ small.

Fixing these three types of variables produce different effects, consistently with the interpretation of the surveys. Fixing a biased variable does not alter the structure of the clusters and the complexity changes smoothly (few clusters are eliminated). Fixing an under-constrained variable affects only the internal entropy of the clusters (which we cannot evaluate). Interestingly enough, fixing a balanced variable has an enormous effect: it produces a decrease very close to $\ln 2$ in the complexity, indeed half of the clusters are eliminated by fixing one single balanced variable. Numerical experiments can be easily performed on samples of huge size [7].

### B. Decimation Search Algorithm

One strategy for using the information provided by SP is to fix as many variables as possible without eliminating too many clusters. At each **decimation** step the choice of the variable to fix is made by looking at the biases $W$s. Successively, the problem is simplified and SP is re-run over the smaller problem leading to a new set of $\eta$s and consequently to the new set of biases $W$s and to the complexity of the sub-problem. Eventually, either all variables have been fixed or (more likely) the remaining variables turn out to be under-constrained (i.e. $W_i^{(0)} = 1, \forall i$), in which case a simple search process can be run to find the complete satisfying assignment.

A straightforward implementation of the above ideas provide a simple algorithm, the survey inspired decimation (SID), that can be used to find solutions to random 3-SAT in the hard region $\alpha \in [\alpha_d, \alpha_c]$. We do not expect this implementation to be the most efficient one, in that no particular strategy has been worked out to optimize the decimation process.

**SID** `algorithm`

`INPUT: The factor graph of a Boolean formula in conjunctive normal form. A maximal number of iterations` $t_{max}$ `and a precision` $\epsilon$ `used in SP`

`OUTPUT: One assignment which satisfies all clauses, or 'SP UN-CONVERGED', or 'probably UNSAT'`

`0. Random initial condition for the surveys`

| | % solved | | | $N_{final}/N$ | | | total time (seconds) | | |
|---|---|---|---|---|---|---|---|---|---|
| $\gamma \downarrow$ | $f=4$ | $f=1$ | $f=4$ | $f=4$ | $f=1$ | $f=4$ | $f=4$ | $f=1$ | $f=4$ |
| 4.20 | 100% | | 100% | 0.48 | | 0.50 | 213.92 | | 2552.4 |
| 4.21 | 98% | 100% | 100% | 0.43 | 0.53 | 0.46 | 241.20 | 414.00 | 2882.2 |
| 4.22 | 82% | 100% | 100% | 0.37 | 0.47 | 0.38 | 296.20 | 485.00 | 3338.6 |
| $N \rightarrow$ | $10^5$ | | $10^6$ | $10^5$ | | $10^6$ | $10^5$ | | $10^6$ |

Figure 5: Results obtained by solving with a single decimation run of the SID algorithm 50 random instances of 3-SAT with $N = 10^5$ and 5 with $N = 10^6$ for each values of $\alpha$. All samples were first tried to solve by fixing variables in steps of $f = 4\%$ at a time, then the remaining (unsolved) samples where solved with $f = 1\%$. The maximal number of iteration was taken equal to $10^3$ and the precision for convergence was taken equal to $10^{-2}$. The table shows the fraction of instances which were solved by SID (first column), the fraction of variables which remained in the simplified instance when all surveys are trivial, and the average computer time requested for solving an instance in every case (on a 2.4 GHz PC).

1. `Run SP`
2. `Evaluate all the biases` $\{W_i^{(+)}, W_i^{(-)}, W_i^{(0)}\}$`, and the complexity` $\Sigma$`, using(21-27).`
3. `Simplify and Search:`
   - 3.1 **If non-trivial surveys ($\{\eta \neq 0\}$) are found**, `then fix the variable with the largest` $|W_i^{(+)} - W_i^{(-)}|$`) and simplify the problem.`
   - 3.2 **If all surveys are trivial ($\{\eta = 0\}$)** `, then output the simplified sub-formula and run on it a local search process (e.g. walksat).`
   - 3.3 **If SP does not converge**, `output ''probably unsat'' and stop (or restart, i.e.`**go to 0.** `)`
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`

The algorithm can also be randomized by fixing, instead of the most biased variables, one variable randomly chosen in the set of the XX percent variables with the largest bias (typically XX=5 works well). This strategy allows to use some restart in the case where the algorithm has not found a solution.

The overall idea underlying the search process is simple. At each time step a single variable is fixed according to the outcome of SP and the effect of such fixing is used to simplify the problem. The number of variables reduces from $N_t$ to $N_t - 1 - S_t$, where $S_t$ is the additional number of variables which become fixed due to simplification: satisfied clauses are eliminated, unsatisfied K clauses are transformed into (K-1) clauses. $K = 1$ clauses need to be satisfied and therefore their variables are properly fixed (unit clause propagation) leading to further variable elimination. The process is then repeated on the smaller sub-problem.

Starting from a 3-SAT instance, the algorithm will produce a sequence of mixed 2/3-SAT smaller and smaller sub problems. After fixing a certain fraction of the variables, the sub-formula turns out to be under-constrained (all the surveys are concentrated on $u_{a\to i} = 0$, i.e. $\{\eta_{a\to i} = 0\}$ ) and quickly solved by a local search method. The output of the local search joined with the set of fixed variables provides the overall satisfying assignment.

On top of restart, other correction strategies could also be implemented, e.g. backtrack. However, for large random 3-SAT instances the first run turns out to be almost always successful and no further error correction strategy is needed. On more realistic instances, where the topology of the factor graphs presents some non trivial structure, we expect correlations in the surveys and possibly errors which need to be cured. The SP equation can be generalized to deal with more complex graphs [7, 22] and yet on the practical side some general error correction strategy could become useful.

Extensive numerical experiments on random 3-SAT instances at $\alpha = 4.2 - 4.25$ with size up to $N = 10^7$ have shown a remarkable efficiency of the decimation algorithm. While the process of fixing a single variable takes some time ($O(N)$ operations), typically the first decimation run leads to the solution. The under-constrained subsystems produced by the decimation process turn out to be indeed very easy to solve. Even for the largest samples $N = O(10^7)$, the subsystems generated by SID (which typically involve between a third and half of the original variables) are solved in few seconds by walksat.

If we fix one variable at a time, the overall computational cost on hard random 3-SAT instances scales as $O(N^2)$. If we are not too close to the threshold, e.g. at $\alpha = 4.2$, we may even fix a fraction of variables at a time, leading to a further reduction of the cost to $O(N \ln N)$ (the 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 [21]. Generalization of the algorithm to other problems require some changes which are not yet implemented.

## VII. THE DIFFERENCE BETWEEN BELIEF PROPAGATION AND SURVEY PROPAGATION

The survey propagation equations described above display some similarities with the one found in the Sum-Product algorithm[15]. This 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 [17] and Bayesian networks [8], from which we borrow the name belief propagation (BP). This section explains the main difference between the two approaches.

### A. Belief propagation in the SAT problem

The belief propagation (BP) algorithm is a message passing procedure that generalises the warning passing procedure (WP) of sect.III B. It aims at computing the fraction of SAT assignments where a given variable $x_i$ is true. Let us consider the probability space built by all SAT assignments with equal probability. Calling $a$ one of the clauses in which $x_i$ appears, the basic ingredients of BP are the messages:

- $\mu_{a\to i}(x_i)$, the probability that clause $a$ is satisfied, given the value of the variable $x_i \in \{0,1\}$.
- $\mu_{i\to a}(x_i)$, the probability that the variable takes value $x_i$, when clause $a$ is absent (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 $\mu_{a\to i}(x_i)$.

These messages satisfy the following equations:

$$\mu_{i\to a}(x_i) = C_{i\to a} \prod_{b\in V(i)\setminus a} \mu_{b\to i}(x_i) \ , \tag{28}$$

$$\mu_{a\to i}(x_i) = \sum_{\{x_j (j\neq i)\}} f_a(X) \prod_{j\in V(a)\setminus i} \mu_{j\to a}(x_j) \ , \tag{29}$$

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 (writing for instance $\mu_{i\to a}(x_i) = \gamma_{i\to a}\delta(x_i,0) + (1-\gamma_{i\to a})\delta(x_i,1)$ in the case where $J_i^a = -1$).

Let us denote by

$$\delta_{a\to i} \equiv \prod_{j\in V(a)\setminus i} \gamma_{j\to a} \tag{30}$$

the probability that all variables in clause $a$, except variable $i$, are in the state which violates the clause. It is easy to see that the BP equations (28,29) can be written as:

$$\gamma_{j\to a} = \frac{P^u_{j\to a}}{P^u_{j\to a} + P^s_{j\to a}} \ , \tag{31}$$

where

$$\begin{aligned} P^u_{j\to a} &= \prod_{b\in V_a^s(j)} (1-\delta_{b\to j}) \ , \\ P^s_{j\to a} &= \prod_{b\in V_a^u(j)} (1-\delta_{b\to j}) \ . \end{aligned} \tag{32}$$

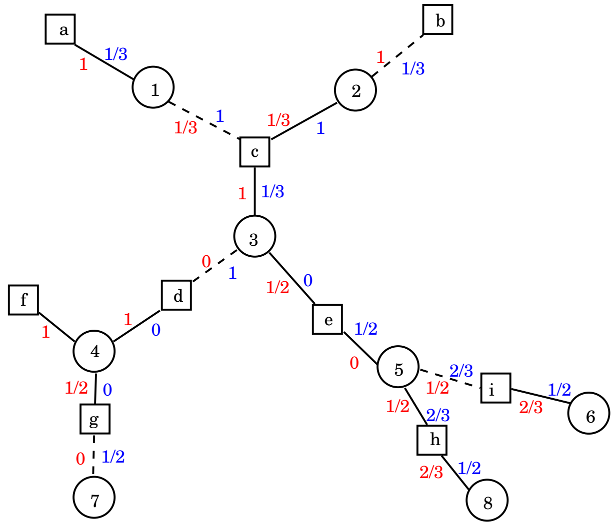


Figure 6: An example of result obtained by the BP algorithm on the tree problem with $N = 8$ variables and $M = 9$ clauses studied in fig.2. On each edge of the graph connecting a function node like $c$ to a variable node like 2, appears the value of $\gamma_{2toc}$ (in blue, on the right hand side of the edge) and of $\delta_{cto2}$ (in red, on the left hand side of the edge). Comparing to the WP result of fig.2, one sees that all the messages $\delta_{atoi} = 1$, corresponding to strict warnings, are the same, while the messages $\delta_{atoi} < 1$ are interpreted in WP as "no warning" (i.e. a null message). Using (33), the probability $\mu_i$ 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 satisfiable assignments as in fig.2.

If an ensemble is empty, for instance $V_a^s(j) = 0$, the corresponding $P_{j\to a}^u$ takes value 1 by definition. 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.

As WP, the BP algorithm is exact on a tree (see for instance [15]). In fact it gives a more accurate result since it allows to compute the exact probability $\mu_i$ (while WP identifies the variables which are fully constrained, and gives a zero local field on the other variables). An example of application is shown in fig.6. In this easy case of the tree, BP also provides the exact number N of SAT assignments, as given by the following theorem:

**THEOREM:** *Consider an instance of the SAT problem for which the factor graph is a tree, and there exist some SAT assignments. Then the BP algorithm converges to a unique set of fixed point messages $\delta^*_{a\to i}$. The probability $\mu_i$ that the variable $x_i = 1$ is given by:*

$$\mu_i = \frac{\prod_{a\in V_-(i)}(1-\delta^*_{a\to i})}{\prod_{a\in V_-(i)}(1-\delta^*_{a\to i}) + \prod_{a\in V_+(i)}(1-\delta^*_{a\to i})} \ . \tag{33}$$

*The number of SAT assignments can be written in terms of the entropy $S = \ln \mathcal{N}$, which is given by:*

$$\begin{aligned} S \ &= \ \sum_{a\in A} \ln\left[\prod_{i\in V(a)}\left(\prod_{b\in V_a^s(i)}(1-\delta^*_{b\to i}) + \prod_{b\in V_a^u(i)}(1-\delta^*_{b\to i})\right) - \prod_{i\in V(a)}\left(\prod_{b\in V_a^u(i)}(1-\delta^*_{b\to i})\right)\right] \\ &\quad + \sum_{i\in X}(1-n_i)\ln\left[\prod_{b\in V_+(i)}(1-\delta^*_{b\to i}) + \prod_{b\in V_-(i)}(1-\delta^*_{b\to i})\right] \end{aligned} \tag{34}$$

**PROOF:**

The proof of convergence is simple, using the same strategy as the proof in sect.III: 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.

The probability $\mu_i$ is computed from the same procedure as the one giving the BP equations (28,29), with the difference that one takes into account all neighbors of the site $i$.

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:

$$P(X) = \frac{1}{\mathcal{N}} \prod_{a\in A} f_a(X) \ . \tag{35}$$

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 the full probability can be expressed in terms of the site and clause marginals as:

$$P(X) = \prod_{a \in A} p_a(X_a) \prod_{i \in X} p_i(x_i)^{1-n_i} \ . \tag{36}$$

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

$$S = -\sum_X P(X) \ln P(X) = -\sum_a \sum_{X_a} p_a(X_a) \ln[p_a(X_a)] - \sum_i (1-n_i) \sum_{x_i} p_i(x_i) \ln[p_i(x_i)] \tag{37}$$

Let us now derive the expression of this quantity in terms of the messages used in BP. One has:

$$p_i(x_i) = c_i \prod_{b \in V(i)} \mu_{b \to i}(x_i) \tag{38}$$

and

$$p_a(X_a) = c_a f_a(X_a) \prod_{i \in V(a)} \mu_{i \to a}(x_i) \ , \tag{39}$$

where $c_i$ and $c_a$ are two normalization constants. From (38) one gets after some reshuffling:

$$\sum_i (n_i - 1) \sum_{x_i} p_i(x_i) \ln p_i(x_i) = \sum_i (n_i - 1) \ln c_i + \sum_a \sum_{i \in V(a)} \sum_{X_a} p_a(X_a) \ln \left[ \prod_{b \in V(i) \setminus a} \mu_{b \to i}(x_i) \right] \ ; \tag{40}$$

Using the BP equation (28), this gives:

$$\sum_i (n_i - 1) \sum_{x_i} p_i(x_i) \ln p_i(x_i) = \sum_i (n_i - 1) \ln c_i + \sum_a \sum_{X_a} p_a(X_a) \ln \left[ \prod_{i \in V(a)} \mu_{i \to a}(x_i) f_a(X_a) \right] - \sum_a \sum_{i \in V(a)} \ln C_{i \to a} \ , \tag{41}$$

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:

$$S = -\sum_a \ln c_a + \sum_i (n_i - 1) \ln c_i - \sum_a \sum_{i \in V(a)} \ln C_{i \to a} \tag{42}$$

In the notations of (31,32), one has

$$c_a = \frac{1}{1 - \prod_{i \in V(a)} \gamma_{i \to a}} = \frac{1}{1 - \prod_{i \in V(a)} P^u_{i \to a} / (P^u_{i \to a} + P^s_{i \to a})} \ , \tag{43}$$

$$C_{i \to a} = \frac{1}{P^u_{i \to a} + P^s_{i \to a}} \ , \tag{44}$$

and

$$c_i = \frac{1}{\prod_{b \in V_+(i)} (1 - \delta_{b \to i}) + \prod_{b \in V_-(i)} (1 - \delta_{b \to i})} \ . \tag{45}$$

Substitution into (42) gives the expression (34) for the entropy. $\square$

### B. The null message and the joker state

It is very instructive to compare the set of equations (28,29) of the BP algorithm to those of SP written in (17,18). The difference is that, in SP, one takes into account separately the case in which there is a null message onto a variable, meaning that it receives no warning at all. In this case the environment of the variable does not constrain it. The probability that variable $j$ is unconstrained, in the absence of clause $a$, is nothing but the probability $\Pi^0_{j\to a} = \prod_{b\in V(j)\setminus a}(1-\eta_{b\to j})$ defined in (18). If this probability is put equal to zero, i.e. if one does not take this possibility into account, one gets back the equations of BP. The originality of SP can thus be understood as the introduction of a new state of the variables, a 'joker state' as it has been called in [24], meaning that the variable is not constrained. This interpretation has been proposed recently for the coloring and satisfiability problems in [24, 25]. We expect that it can be used to 'derive' the SP algorithm as a belief propagation algorithm applied to this enlarged space of variables and messages.

Let us discuss explicitly the role played by the joker state in a BP like scheme. Each variable can now be in three states $0, 1, *$. The $*$ state is the joker state, meaning that the variable is not constrained.

Borrowing the notations of the BP equations (28, 29) 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

$$\delta_{a\to i} = \prod_{j\in V(a)\setminus i} \gamma_{j\to a} \tag{46}$$

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 $b \in \{V(i)\setminus a\}$ 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

$$\Pi^0_{i\to a} = \prod_{b\in V(i)\setminus a} (1-\delta_{b\to i}) \tag{47}$$

- 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

$$\Pi^u_{i\to a} = \left[1 - \prod_{b\in V_a^u(i)} (1-\delta_{b\to i})\right] \prod_{b\in V_a^s(i)} (1-\delta_{b\to i}) \tag{48}$$

- 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

$$\Pi^s_{i\to a} = \left[1 - \prod_{b\in V_a^s(i)} (1-\delta_{b\to i})\right] \prod_{b\in V_a^u(i)} (1-\delta_{b\to i}) \tag{49}$$

- 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

$$\Pi^c_{i\to a} = \left[1 - \prod_{b\in V_a^s(i)} (1-\delta_{b\to i})\right] \left[1 - \prod_{b\in V_a^u(i)} (1-\delta_{b\to i})\right] \tag{50}$$

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:

$$\gamma_{i\to a} = \Pi^u_{i\to a}/(\Pi^u_{i\to a} + \Pi^s_{i\to a} + \Pi^0_{i\to a}) \tag{51}$$

The above equations in the enlarged space including the null message, given in (46-51) are identical to the SP equations(17,18), with the identification $\eta_{a\to i} = \delta_{a\to i}$. If these equations converge, one can use the resulting messages to estimate the probabilities that the variable $x_i$ is in each of the states $0, 1, *$. The result is obviously given respectively by the quantities $W_i^{(-)}, W_i^{(+)}, W_i^{(0)}$ defined in (22,23,24).

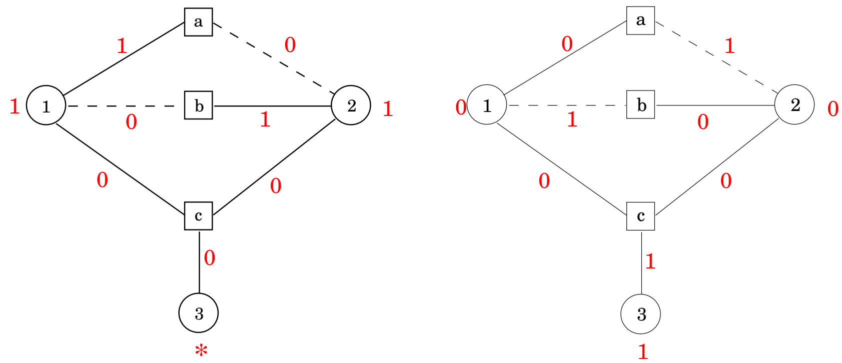


Figure 7: A simple SAT problem with three variables, two 2-clauses, one 3-clause, and loops. Each figure gives one of the two possible solutions of warning propagation. The number on each edge $a-i$ is the cavity bias $u_{a\to i}$, the number next to each variable node is the corresponding generalized 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 $\eta_{c\to 3} = (1-x)^2 y^2/(1-xy)^2$

### C. Clusters and the joker state

The introduction of the null message and the joker state thus gives some kind of a new interpretation to the whole SP algorithm: this algorithm allows to find the statistics of non-contradictory messages, providing assignments of the variables in the enlarged state space including the joker state. What is the meaning of these generalized assignments? They must be understood as non-contradictory solutions of the WP algorithm. On a tree, the WP algorithm gives one single set of warnings, and therefore there is a single generalized assignment (for instance, for the tree problem of fig. 2, the variables $x_1$ to $x_8$ are in the states $1, 0, 1, 1, *, *, *, *$). The BP generalization becomes useful only when there exist several possible generalized assignments, which is possible only in the presence of loops. In a loopy factor graph, it is possible that there will exist several different sets of warnings satisfying the WP equations, and different generalized assignments, as in the example of fig. 7. SP is an attempt at finding the statistics of these assignments. Experimentally, in graphs with loops, it turns out that SP is able to converge where BP does not [27]. Three questions arise naturally:

- 1) Does SP converge to a unique set of fixed point messages?
- 2) Does this fixed point message, or at least one of them, contain the correct information about all the generalized assignments?
- 3) What is the relationship between the set of generalized assignments and the possible clustering of solutions discussed in sect. IV?

The answer to questions 1) and 2) is in general no. The trivial example of fig. 7 is a counterexample. The answer to question 3) can be taken as a tautology: one can use the generalized assignments to define the clusters, each generalized assignment corresponding to one 'cluster'. For instance in the example of fig. 7, there exist two generalized assignments, $(1, 1, *)$ and $(0, 0, 1)$, corresponding to two 'clusters' of real SAT assignments: $C_1 = \{(1, 1, 0), (1, 1, 1)\}$ and $C_2 = \{(0, 0, 1)\}$. Furthermore, because WP converges to a unique solution on a tree, we know that two sets of warnings leading to two distinct generalized assignments must differ on a set of edges which has some loops. In the case of random 3-SAT problems where short loops are rare, this implies that distinct generalized assignments must typically differ on a large set of nodes (with a size which diverges when $N \to \infty$), which justifies in this case the identification of $\Sigma$ with the complexity of the problem.

Because message passing procedure are well controlled only on tree factor-graphs, it is difficult to make any general statement on SP in general, and it is easy to find simple cases in which it is not very useful, as the previous discussion has shown. However, one may notice that these cases are very easily solved by simple WP, and a set of WP messages is a special case of SP. The difficult case for WP is when different parts of the graph send messages corresponding to different generalized assignments. This will typically happen for a graph with some components which are not well connected, and in this case SP is supposed to perform better. Much more empirical and theoretical work is needed to establish the circumstances in which SP is really better, and to modify it to other cases.

## VIII. 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. 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 [23]), 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 Constrained Satisfaction Problems [7, 16]. At the moment the approach is very heuristic, although it is based on a rather detailed characterization of the phase structure of random 3-SAT, which can be checked exactly in the 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 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 [24].

### Acknowledgements

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, 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'.

---

[1] S.A. Cook, D.G. Mitchell, Finding Hard Instences of the Satisfiability Problem: A Survey, In: Satisfiability Problem: Theory and Applications. Du, Gu and Pardalos (Eds). DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 35, (1997)

[2] Dubois O. , Monasson R., Selman B. & Zecchina R. (Eds.), Phase Transitions in Combinatorial Problems, Theoret. Comp. Sci. **265** (2001).

[3] S. Kirkpatrick, B. Selman, Critical Behaviour in the satisfiability of random Boolean expressions, Science 264, 1297 (1994)

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

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

[6] M. Mézard, G. Parisi and R. Zecchina, Science 297, 812 (2002) (Sciencexpress published on-line 27-June-2002; 10.1126/science.1073287)

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

[8] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, 2nd ed. (San Francisco, MorganKaufmann,1988)

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

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

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

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

[13] S. Cocco, O. Dubois, J. Mandler, R. Monasson, *Rigorous decimation-based construction of ground pure states for spin glass models on random lattices*, Phys. Rev. Lett. 90, 047205 (2003)
[14] *Complexity transitions in global algorithms for sparse linear systems over finite fields*, J. Phys. A 35, 7559 (2002), A. Braunstein, M. Leone, F. Ricci-Tersenghi, R. Zecchina.
[15] F.R. Kschischang, B.J. Frey, H.-A. Loeliger, Factor Graphs and the Sum-Product Algorithm, *IEEE Trans. Infor. Theory* **47**, 498 (2002).
[16] *Coloring Random Graphs*, R. Mulet, A. Pagnani, M. Weigt, R. Zecchina, Phys. Rev. Lett. 89, 268701 (2002)
[17] R. G. Gallager, *Low-Density Parity-Check Codes* Cambridge, MA: MIT Pres,1963.
[18] B. Selman, H. Kautz, B. Cohen, Local search strategies for satisfiability testing, in: *Proceedings of DIMACS*, p. 661 (1993).
[19] Satisfiability Library: www.satlib.org/
[20] G. Parisi, in preparation.
[21] `www.ictp.trieste.it/~zecchina/SP`
[22] J.S. Yedidia, W.T. Freeman and Y. Weiss, Generalized Belief Propagation, in *Advances in Neural Information Processing Systems 13* eds. T.K. Leen, T.G. Dietterich, and V. Tresp, MIT Press 2001, pp. 689-695.
[23] Mézard, M., Parisi, G., & Virasoro, M.A. Spin Glass Theory and Beyond, World Scientific, Singapore (1987).
[24] *Survey propagation for general Contraint Satisfaction Problems*, A. Braunstein, M. Mezard, M. Weigt, R. Zecchina, Volume on Computational Complexity and Statistical Physics, Oxford University Press,Santa Fe Institute Studies in the Sciences of Complexity (2003), ArXiv: lanl/arXiv.org/cond-mat/0212451
[25] *On the survey-propagation equations for the random K-satisfiability problem*, G. Parisi, cs.CC/0212009.
[26] In statistical physics such a scenario is called *paramagnetic*
[27] In this respect, the case of the decoding algorithm in error correcting codes is different in that initial condition are always close to the solution.