Simplifying random satisfiability problems by removing frustrating interactions

A. Ramezanpour[*]
Institute for Advanced Studies in Basic Sciences, Zanjan 45195-1159, Iran

S. Moghimi-Araghi[†]
Department of Physics, Sharif University of Technology, P.O. Box 11365-9161, Tehran, Iran

(Received 6 June 2006; published 6 October 2006)

How can we remove some interactions (generate shorter clauses) in a constraint satisfaction problem (CSP) such that it still remains satisfiable? In this paper we study a modified survey propagation algorithm that enables us to address this question for a prototypical CSP, i.e., random K-satisfiability problem. The average number of removed interactions is controlled by a tuning parameter in the algorithm. If the original problem is satisfiable then we are able to construct satisfiable subproblems ranging from the original one to a minimal one with minimum possible number of interactions. The minimal satisfiable subproblems will directly provide the solutions of the original problem.

DOI: 10.1103/PhysRevE.74.041105 PACS number(s): 02.50.−r, 75.10.Nr, 64.60.Cn

I. INTRODUCTION

There are many combinatorial problems that can be represented as a constrained satisfaction problem (CSP) in which we are to satisfy a number of constrains defined over a set of discrete variables. An interesting example is the low-density parity-check code in information theory [1]. Here a code word consists of N variables $\in\{0,1\}$ that satisfy M parity-check constraints. Each constraint acts on a few variables and is satisfied if sum of the variables module 2 is zero. Another example is finding the fixed points of a random Boolean network [2]. Again we have N Boolean variables represented by the nodes of a directed network. The state of a node at a given time step is a logical function of the state of its incoming neighbors in the previous time step. Thus a fixed point of the problem is one that satisfies N constraints, one for each variable, where a constraint enforces the variable taking the outcome of the logical function.

From a physical point of view there exists a close relation between these problems with frustrated systems exhibiting glassy behavior, such as spin glasses [3]. The methods and concepts developed in the study of these systems enable us to obtain a better understanding of the above problems.

Random satisfiability problem is a typical CSP that allows us to study combinatorial CSP's in a simple framework. It is the first problem whose NP-completeness has been proven [4,5]. The problem is defined over N logical variables that are to satisfy M logical constraints or clauses. Each clause contains some randomly selected variables that can appear negated or as such with equal probability. The clause is satisfied if at least one of its components be TRUE. Here the interest is in the satisfiability of the problem and finding the solutions or ground-state configurations that result to the minimum number of violated clauses. For small number of clauses per variable $\alpha = M/N$, a typical instance of the problem is satisfiable (SAT), that is there is at least one configuration of the variables that satisfies all the clauses. On the other hand, for large α a typical instance of the problem is unsatisfiable (UNSAT) with probability one. We have a sharp transition at α_c that separates SAT and UNSAT phases of the problem [6].

The interaction pattern of clauses with variables makes a graph that is called the factor graph [7]. Notice that a larger number of interactions leads to more frustration and thus makes the problem harder both in checking its satisfiability and finding its solutions. Therefore, one way to make the problem easier is to reduce it to some smaller subproblems with smaller number of interactions. Then we would be able to utilize some local search algorithms (like Walksat and its generalizations [8]) to solve the smaller subproblem. However, for a given number of variables and clauses the chance to find a solution decreases as we remove the interactions from the factor graph. Moreover, the number of subproblems with a given number of interactions is exponentially large. These facts make the above reduction procedure inefficient unless we find a way around them.

Survey propagation algorithm is a powerful massage passing algorithm that helps us to check the satisfiability of the problem and find its solutions [9,10]. In Ref. [11] we showed that as long as we are in the SAT phase we can modify this algorithm to find the satisfiable spanning trees. Indeed the modified algorithm introduced in [11] enables us to remove some interactions from the problem such that the obtained subproblem is still satisfiable. There, we also showed that there is a correspondence between the set of solutions in the original problem and those of the satisfiable spanning trees.

In this paper we are going to investigate the modified algorithm in more details, by studying its performance for different classes of subproblems. There is a free parameter in the algorithm that allows us to control the number of interactions in the subproblems. In this way we can construct ensembles of satisfiable subproblems with different average number of interactions. The largest subproblem is the original problem and the smallest one is a subproblem in which each clause contains just one variable. The latter satisfiable subproblems, which we call minimal satisfiable subprob-

[*]Electronic address: ramezanpour@iasbs.ac.ir
[†]Electronic address: samanimi@sharif.edu

lems, result directly to the solutions of the original problem. We will show how the number of solutions (in replica symmetry breaking approximation) and the complexity (in one-step replica symmetric approximation) varies for different subproblems close to the SAT-UNSAT transition.

The paper is organized in this manner: First we define more precisely the random K-satisfiability problem and its known features. In Sec. III we briefly introduce belief and survey propagation algorithms that play an essential role in the remaining parts of the paper. Section IV has been divided into two subsections that deal with satisfiable subproblems. We start by some general arguments and then represent numerical results for different satisfiable subproblems. Finally Sec. V is devoted to our conclusion remarks.

II. RANDOM K-SATISFIABILITY PROBLEM

A random satisfiability problem is defined as follows: We take N logical variables $x_i \in \{0,1\}$. Then we construct a formula F of M clauses joined to each other by logical AND. Each clause contains a number of randomly selected logical variables. In the random K-SAT problem each clause has a fixed number of K variables. These variables, which join to each other by logical OR, are negated with probability $1/2$, otherwise appear as such. For example $F := (\bar{x}_2 \vee x_4) \wedge (\bar{x}_3 \vee x_2) \wedge (x_1 \vee x_3)$ is a 2-SAT formula with three clauses and four logical variables. A solution of F is a configuration of logical variables that satisfy all the clauses. The problem is satisfiable if there is at least one solution or satisfying configuration of the variables. Given an instance of the problem, then we are interested to know if it is satisfiable or not. A more difficult problem is to find the ground-state configurations which lead to the minimum number of violated clauses.

The relevant parameter that determines the satisfiability of F is $\alpha := = M/N$. In the thermodynamic limit ($N, M \to \infty$ and $\alpha \to$ const.) F is satisfied with probability one as long as $\alpha < \alpha_c$. Moreover, it has been found that for $\alpha_d < \alpha < \alpha_c$ the problem is in the Hard-SAT phase [9]. At α_d we have a dynamical phase transition associated with the break down of replica symmetry. Assuming one-step replica symmetry breaking, one obtains $\alpha_d \simeq 3.92$ and $\alpha_c \simeq 4.26$ for random 3-SAT problems [9]. Although this approximation seems to be exact near the SAT-UNSAT transition, it fails close to the dynamical transition where higher order replica symmetry breaking solutions are to be used [12,13].

A useful tool to study CSP's is the factor graph which is a bipartite graph of variable nodes and function nodes (clauses). The structure of this graph is completely determined by a $M \times N$ matrix with elements $J_{a,i} \in \{0,+1,-1\}$; $J_{a,i} = +1$ if clause a contains x_i, it is equal to -1 if $\bar{x}_i$ appears in a and otherwise $J_{a,i} = 0$. In a graph representation, we add an edge between function node a and variable node i if $J_{a,i} \neq 0$. The edges will be shown by a filled line if $J_{a,i} = +1$ and by a dashed line if $J_{a,i} = -1$.

We also define an energy (or cost function) for the problem which is the number of violated clauses for a given configuration of variables

$$E[\{s\}] \equiv \sum_{a=1}^{M} \prod_{j=1}^{K} \left(\frac{1 - J_{a,i_j^a} s_{i_j^a}}{2} \right). \tag{1}$$

Here we introduced spin variables $s_i = 2x_i - 1 \in \{-1,1\}$ and i_j^a is the index of jth variable in clause a. A solution of the problem is a configuration of zero energy and the ground states are those configuration having the minimum energy. Note that the presence of two variables in the same clause results to direct interactions between the corresponding spin variables.

III. BELIEF AND SURVEY PROPAGATION ALGORITHMS

In this section we give a brief description of some massage passing algorithms which help us to obtain some insights about the solution space of the problem. These algorithms have an iterative nature and can give information for single instances of the problem. For more details about the algorithms and their origin see [7,9,10].

A. Assuming replica symmetry; belief propagation

In the following we restrict ourselves in the SAT phase where there are some solutions that satisfy the problem. These solutions are represented by points in the N-dimensional configuration space of the variables. If the number of interactions is low enough we can assume a replica symmetric structure for the organization of the solutions in this space. It means that all the solutions lie in a single cluster (or pure state) in which any two solutions can be connected to each other by a path of finite steps (in the thermodynamic limit). Belief propagation algorithm enables us to find the solutions and their number (the cluster's size or entropy of the pure state) in this case.

Consider the set of solutions with $\mathcal{N}_s$ members. Each member is defined by N values for the variables $\{s_i^* \in \{-1,1\} | i = 1, \ldots, N\}$. We consider the probability space made by all the solutions with equal probability. Then we define the warnings $\eta_{a\to i}$ as the probability that all variables in clause a, except the x_i, are in a state that violates a. Assuming a treelike structure for the factor graph (i.e., ignoring the correlations between neighboring variables), $\eta_{a\to i}$ can be written as

$$\eta_{a\to i} = \prod_{j \in V(a)-i} P_a^u(j), \tag{2}$$

where $P_a^u(j)$ is the probability that variable j dose not satisfy clause a. We also denote by $V(a)$, the set of variables that belong to clause a and by $V(i)$, the set of clauses that variable i contributes in. In belief propagation algorithm $P_a^u(j)$ is given by [10]

$$P_a^u(j) = \frac{\Pi_{j\to a}^u}{\Pi_{j\to a}^s + \Pi_{j\to a}^u}, \tag{3}$$

where

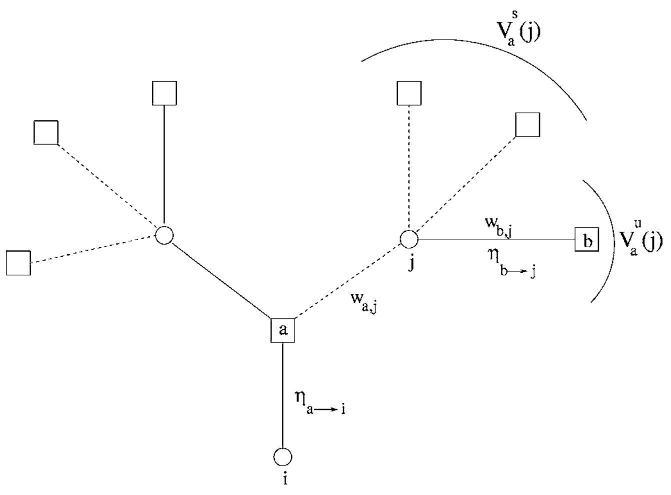

FIG. 1. Factor graph representation of the problem. Squares and circles denote function and variable nodes, respectively.

$$\Pi^u_{j\to a} = \prod_{b\in V_a^s(j)} (1-\eta_{b\to j}),$$

$$\Pi^s_{j\to a} = \prod_{b\in V_a^u(j)} (1-\eta_{b\to j}). \quad (4)$$

Here $V_a^s(j)$ denotes to the set of clauses in $V(j)-a$ that variable j appears in them as it appears in clause a, see Fig. 1. The remaining set of clauses are denoted by $V_a^u(j)$.

Starting from initial random values for η's, one can update them iteratively according to Eqs. (2)–(4). If the factor graph is spars enough and the problem is satisfiable then the iteration may converge with no contradictory warnings. Utilizing these warnings one can use the following relations to find the entropy of the pure state [10]

$$S = \ln \mathcal{N}_s = \sum_{a=1}^{M} S_a - \sum_{i=1}^{N} (k_i-1)S_i, \quad (5)$$

where

$$S_a = \ln\left[\prod_{j\in V(a)} (\Pi^s_{j\to a}+\Pi^u_{j\to a}) - \prod_{j\in V(a)} \Pi^u_{j\to a}\right],$$

$$S_i = \ln(\Pi_i^- + \Pi_i^+), \quad (6)$$

and

$$\Pi_i^- = \prod_{a\in V_+(i)} (1-\eta_{a\to i}),$$

$$\Pi_i^+ = \prod_{a\in V_-(i)} (1-\eta_{a\to i}). \quad (7)$$

In these equations $V_\pm(i)$ are the set of function nodes in $V(i)$ with $J_{a,i}=\pm 1$ and k_i is the number of clauses in $V(i)$.

It has been shown that the above algorithm gives exact results for treelike factor graphs [10].

B. Assuming one-step replica symmetry breaking: Survey propagation

When we have one-step replica symmetry breaking, the solutions organize in a number of well separated clusters with their own internal entropies. Suppose there are $\mathcal{N}_c$ of such clusters. In a coarse grained picture, we can assign a state $\{\sigma_i^* \in \{-1,0,1\} | i=1,\ldots,N\}$ to each cluster of the solution space. For a given cluster we set $\sigma_i^*=+1/-1$ if x_i has the same value $+1/-1$ in all the solutions belonging to the cluster. Otherwise, that is if x_i is not frozen and alternates between -1 and 1, $\sigma_i^*=0$. Again we can define a probability space in which all the clusters have the same probability. As before, $\eta_{a\to i}$ is the probability (in new space) that all variables in clause a, except x_i, are in states that violate clause a. Notice that we have to take into account the extra state $\sigma_i^*=0$, which is called the joker state, in the calculations. Generalizing the belief propagation relations one obtains [10]

$$\eta_{a\to i} = \prod_{j\in V(a)-i} P_a^u(j), \quad (8)$$

where

$$P_a^u(j) = \frac{\Pi^u_{j\to a}}{\Pi^s_{j\to a}+\Pi^0_{j\to a}+\Pi^u_{j\to a}}. \quad (9)$$

But now

$$\Pi^0_{j\to a} = \prod_{b\in V(j)-a} (1-\eta_{b\to j}),$$

$$\Pi^u_{j\to a} = \left(1-\prod_{b\in V_a^u(j)} (1-\eta_{b\to j})\right) \prod_{b\in V_a^s(j)} (1-\eta_{b\to j}),$$

$$\Pi^s_{j\to a} = \left(1-\prod_{b\in V_a^s(j)} (1-\eta_{b\to j})\right) \prod_{b\in V_a^u(j)} (1-\eta_{b\to j}). \quad (10)$$

The above equations can be solved iteratively for η's. As long as we are in the SAT phase, the above algorithm may converge with no contradictory warnings. Then the configurational entropy or complexity of the problem reads [10]

$$\Sigma = \ln \mathcal{N}_c = \sum_{a=1}^{M} \Sigma_a - \sum_{i=1}^{N} (k_i-1)\Sigma_i, \quad (11)$$

where

$$\Sigma_a = \ln\left(\prod_{j\in V(a)} (\Pi^s_{j\to a}+\Pi^0_{j\to a}+\Pi^u_{j\to a}) - \prod_{j\in V(a)} \Pi^u_{j\to a}\right),$$

$$\Sigma_i = \ln(\Pi_i^- + \Pi_i^0 + \Pi_i^+), \quad (12)$$

and

$$\Pi_i^- = \left(1-\prod_{a\in V_-(i)} (1-\eta_{a\to i})\right) \prod_{a\in V_+(i)} (1-\eta_{a\to i}),$$

$$\Pi_i^+ = \left(1-\prod_{a\in V_+(i)} (1-\eta_{a\to i})\right) \prod_{a\in V_-(i)} (1-\eta_{a\to i}),$$

$$\Pi_i^0 = \prod_{a\in V(i)} (1-\eta_{a\to i}). \quad (13)$$

To find a solution of the problem one can follow a simple survey inspired decimation algorithm that works with the biases a variable experiences [10]. Let us define W_i^+ as the probability that in a randomly selected cluster of solutions, variable i be frozen in state +1. Similarly we define W_i^- and W_i^0. Then, according to the above definitions we have

$$W_i^+ = \frac{\Pi_i^+}{\Pi_i^+ + \Pi_i^0 + \Pi_i^-},$$

$$W_i^- = \frac{\Pi_i^-}{\Pi_i^+ + \Pi_i^0 + \Pi_i^-},$$

$$W_i^0 = 1 - W_i^+ - W_i^-. \qquad (14)$$

After a single run of the survey propagation algorithm we would have the above probabilities. So, we could fix the state of the most biased variable (one that has the largest $|W^+ - W^-|$) to the favored value. In this way, the variable might satisfy some clauses that would be removed from the problem. Moreover, removing the assigned variable would result in some clauses with just a single variable. These degree-one clauses could easily be satisfied by assigning the appropriate value to the associated variables. One could again remove the satisfied clauses and fixed variables until no degree-one clause remains in the problem. Then we run the survey propagation algorithm on the simplified problem and repeat the above procedure. This process continues until we reach a simple problem with all warnings equal to zero. This problem can then be solved by a local search in the configuration space of the remained variables. The result would be a configuration of the variables that satisfies all the clauses.

IV. FINDING SATISFIABLE SUBPROBLEMS

Consider a satisfiable random K-SAT problem and the associated factor graph with N variable nodes, M function nodes, and KM edges. All the function nodes have the same degree $k_a = K$ and a variable node has degree k_i which, in the thermodynamic limit, follows a Poisson distribution of mean $K\alpha$. If $\{s_i^* | i = 1, \ldots, N\}$ is a solution of the problem, then any function node in the factor graph has at least one neighboring variable node that satisfies it. It means that for any solution we could remove some of the edges in the factor graph while the obtained subproblem is still satisfiable and $k_a \geqslant 1$ for all the function nodes. Obviously we could do this process until each function node had only one variable node, the one that should satisfy the corresponding clause. So, it is clear that for a satisfiable problem there exist many subproblems ranging from the original one, with $L = KM$ edges (or interactions), to a minimal one with $L = M$ edges in its factor graph. In general we define $G_x(M,N)$ as the ensemble of satisfiable subproblems defined by the parameter x. For example $G_L(M,N)$ is the ensemble of satisfiable subproblems with L edges.

An interesting point is the presence of a correspondence between the solutions of the original problem and the solutions of an ensemble of subproblems with L edges. Obviously any solution of the subproblems in $G_L(M,N)$ is also a solution of the original problem. Moreover, as described above, for any solution we can remove some of the edges until we obtain a subproblem of exactly L edges. In [11] we showed that this correspondence holds also for the set of spanning trees which is a subset of $G_{M+N-1}(M,N)$. These correspondence relations will allow us to construct the ensembles and to find the solutions of the original problem by solving a subproblem.

Notice that as the number of interactions in a problem decreases we have to pay less computational cost to solve it. In fact, treelike factor graphs can easily be solved by efficient local search algorithms. And if someone could give the ensemble of minimal subproblems, the whole set of solutions would be available. Now the main questions are the following: How can we construct these satisfiable subproblems and what can be said about the properties of these subproblems? We try to answer these questions by a simple modification of survey propagation algorithm, introduced in [11].

A. General arguments

For a given ensemble of subproblems $G_x(M,N)$ we would have $\mathcal{N}_x(M,N)$ members. Consider the ensemble G_x and the edge (a,i) in the factor graph. We define the weight $w_{a,i}$ as a measure of the edge's appearance frequency in the ensemble, that is,

$$w_{a,i} = \frac{1}{\mathcal{N}_x} \sum_{g \in G_x} y_{a,i}(g), \qquad (15)$$

where $y_{a,i}(g) = 1$ if the edge appears in g and otherwise $y_{a,i}(g) = 0$. Let $P_x(g)$ be a measure defined on the space of all subgraphs with equal probability for all subgraphs g that belong to G_x and zero otherwise. This probability can be written in terms of y's

$$P_x(g) = \frac{1}{\mathcal{N}_x} \sum_{g' \in G_x} \prod_{(a,i)} \delta_{y_{a,i}(g), y_{a,i}(g')}. \qquad (16)$$

It is then easy to show that $\Sigma_g P_L(g) = 1$ and $\ln \mathcal{N}_x = -\Sigma_g P_x(g) \ln P_x(g)$.

Suppose that we have obtained w's for the ensemble G_x from another way. As an estimate of $P_x(g)$ we write

$$P_x^e(g) = \prod_{(a,i)} \{y_{a,i}(g) w_{a,i} + [1 - y_{a,i}(g)](1 - w_{a,i})\}. \qquad (17)$$

Then we expect that

$$\ln \mathcal{N}_x^e = -\sum_g P_x^e(g) \ln P_x^e(g)$$
$$= -\sum_{(a,i)} [w_{a,i} \ln w_{a,i} + (1 - w_{a,i}) \ln(1 - w_{a,i})] \qquad (18)$$

gives a good estimate of $\ln \mathcal{N}_x$.

Suppose that we have obtained all the members in ensemble G_x. Assuming replica symmetry, we could run belief propagation on each member of the ensemble and obtain its entropy. Then we could define $\langle S_x \rangle$, the average of entropy taken over ensemble G_x. Similarly we could run survey propagation algorithm and define $\langle \Sigma_x \rangle$ as the average complexity of the subproblems in G_x. Actually we will not follow the above procedure and get around the difficult problem of finding all the ensemble members. Let us describe our procedure for the case of survey propagation algorithm. Generalization to the belief propagation algorithm would be straightforward.

To obtain w's in a given ensemble we make use of a self-consistent approach. We run survey propagation algorithm on the original factor graph but at the same time we take into account the fact that each edge has its own probability of appearing in the ensemble. Now the survey along edge (a,i) is updated according to the following rule:

$$\eta_{a\to i} = \prod_{j\in V(a)-i} [w_{a,j}P_a^u(j) + 1 - w_{a,j}], \tag{19}$$

where as before $P_a^u(j)$ is given by Eq. (9) with

$$\Pi^0_{j\to a} = \prod_{b\in V(j)-a} (1 - w_{b,j}\eta_{b\to j}),$$

$$\Pi^u_{j\to a} = \left(1 - \prod_{b\in V_a^u(j)} (1 - w_{b,j}\eta_{b\to j})\right) \prod_{b\in V_a^s(j)} (1 - w_{b,j}\eta_{b\to j}),$$

$$\Pi^s_{j\to a} = \left(1 - \prod_{b\in V_a^s(j)} (1 - w_{b,j}\eta_{b\to j})\right) \prod_{b\in V_a^u(j)} (1 - w_{b,j}\eta_{b\to j}). \tag{20}$$

An essential step here is the determination of w's in the given ensemble. Remember that an ensemble is a set of satisfiable subproblems which completely define the weights w's along the edges of the factor graph. Thus, if with a given set of w's we find a large warning sent from a to i, we expect a high probability for the presence of that edge in the ensemble. Here we make a crucial assumption and use the ansatz

$$w_{a,i} = [\eta_{a\to i}]^{\mu} \tag{21}$$

that incorporates the above fact. We take $\mu \geq 0$ as a free parameter and denote the resulted ensemble by G_μ. For a given μ we would have an ensemble of satisfiable subproblems with different number of edges. Because of the functional form of the above ansatz, the average number of edges in the ensemble decreases by increasing μ. Therefore, to obtain smaller satisfiable subproblems we will need to run the algorithm for larger values of μ.

Starting from initially random η's and w's we iterate the above equations until (i) the algorithm converges to some fixed point, (ii) or results to contradictory warnings, (iii) or dose not converge in a predefined limit for the number of iterations t_{max}. We think that as long as the original problem is satisfiable, the algorithm will converge in a finite fraction of times that we run it.

If the algorithm converges then we can utilize our definition for w's and construct satisfiable subproblems. To construct a subproblem in G_μ we go through all the edges and select them with probabilities $w_{a,i}$'s. We hope that such a subproblem be satisfiable with a considerable probability. Moreover, it is reasonable that we pay more computational cost to find smaller satisfiable subproblems which are closer to the solutions of the original problem.

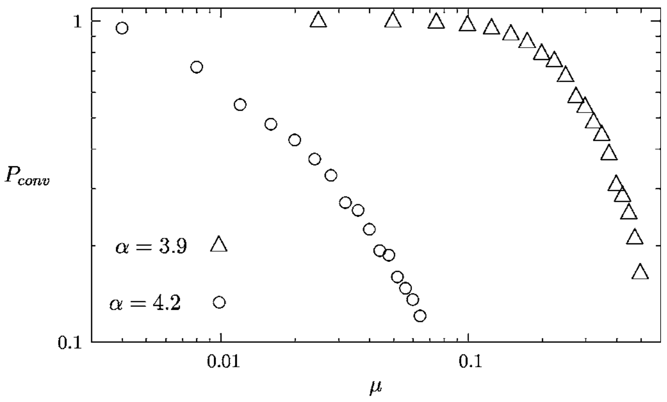

FIG. 2. Convergence probability for two values of α close to the SAT-UNSAT transition. Number of variables is $N=1000$ and statistical errors are of order 0.01.

B. Numerical results

In the following we will study some properties of satisfiable subproblems including the spanning trees of the original factor graph and the minimal subproblems.

We start with random initial values of $0 \leq \eta_{a,i}, w_{a,i} \leq 1$ for all the edges (a,i). Then, in each iteration of the algorithm we update η and w for all the edges according to Eqs. (19)–(21). The edges are selected sequentially in a random way. The algorithm converges if for all edges the differences between new and old values of η are less than ϵ. We bound the number of iterations from above to t_{max} and if the algorithm dose not converge within this number of iterations, we say it diverges. In the following we will work with $\epsilon = 0.001$ and $t_{max}=1000$. Moreover, we consider 3-SAT problems where each clause in the original problem has just $K=3$ variables.

Let us first study the convergence of the modified algorithm. To this end we repeat the algorithm for a number of times and define P_{conv} as the fraction of times in which the algorithm converges. In Fig. 2 we display P_{conv} for the modified survey propagation algorithm. It is observed that P_{conv}

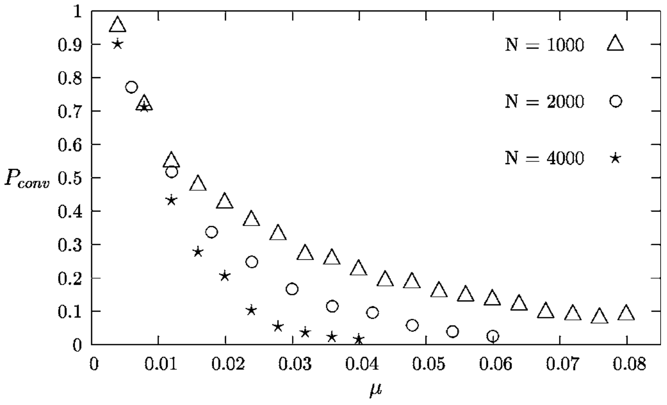

FIG. 3. Convergence probability for different problem sizes at α=4.2. Statistical errors are of order 0.01.

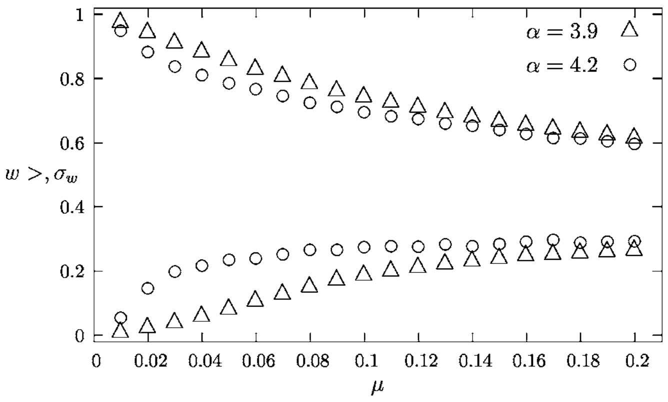

FIG. 4. The average weight and its standard deviation for $N = 1000$. Statistical errors are about the size of the points.

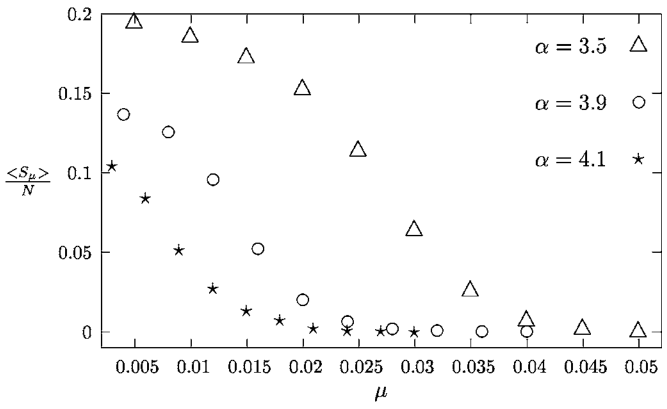

FIG. 6. Average entropy of a subproblem in G_μ for $N=1000$. Statistical errors are about the size of the points.

decreases by increasing μ. Moreover, P_{conv} diminishes more rapidly for larger α. It is reasonable because removal of the edges becomes harder as we get closer to the SAT-UNSAT transition. What happens if we increase the problem size? Figure 3 shows the finite size effects on convergence probability. These effects are significant due to the small problem sizes studied here. Moreover, as expected, the probability decreases more rapidly as N increases.

To see how the number of edges changes with μ we obtained the average weight of an edge, $\langle w \rangle$, and its standard deviation, σ_w, in converged cases. The average number of edges is given by $\langle L \rangle = 3M\langle w \rangle$. Figure 4 shows how these quantities behave with μ. We found that as μ becomes larger $\langle w \rangle$ decreases and finally (not shown in the figure) approaches to $1/3$, the minimum possible value to have a satisfiable subproblem when $K=3$.

Using our arguments in previous subsection we can obtain an estimate of the number of members in the ensemble G_μ, $\mathcal{N}^e_\mu$. In Fig. 5 we show how $\ln \mathcal{N}^e_\mu$ changes with μ. Here we have displayed the results just for small μ's where we are interested in. For larger μ's, $\mathcal{N}^e_\mu$ decreases to its value for $\langle L \rangle = M$.

As described in the previous section we can obtain the average entropy of a typical subproblem in G_μ by running belief propagation on it. The results have been displayed in Fig. 6. Similarly the average complexity of a subproblem is obtained by running survey propagation algorithm. Figure 7 shows this quantity for some values of α. As the figures show both $\langle S_\mu \rangle$ and $\langle \Sigma_\mu \rangle$ diminish with μ and α. Removing edges from the factor graph and approaching the SAT-UNSAT transition both decrease the number of solutions and complexity. Notice that for a fixed value of α we can define the threshold $\mu_c(\alpha)$ where the complexity vanishes. It is a decreasing function of α and we know already that $\mu_c(\alpha_c) = 0$.

1. Satisfiable spanning trees

Suppose that the algorithm converges and returns the weights w's for all the edges of the factor graph. It is not difficult to guess that maximum spanning trees have a larger probability to be a satisfiable spanning tree. A maximum spanning tree is a spanning tree of the factor graph with maximum weight $W = \sum_{(a,i)} w_{a,i}$. For a given μ and a converged case we can construct maximum spanning trees in the following way: We start from a randomly selected node in the original factor graph and find the maximum weight among the edges that connect it to the rest of the graph. Then we list the edges having a weight in the ϵ neighborhood of the maximum one and add randomly one of them to the new factor graph. If we repeat the addition of edges $N+M-1$ times we obtain a spanning tree factor graph which has the

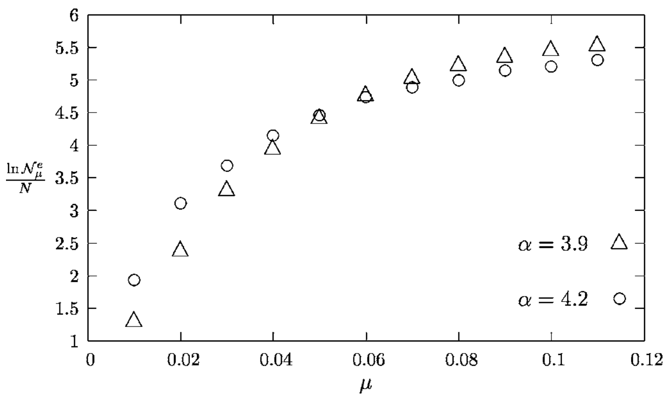

FIG. 5. Estimated value of the number of members in G_μ. The results are for $N=1000$ and statistical errors are of order 0.1.

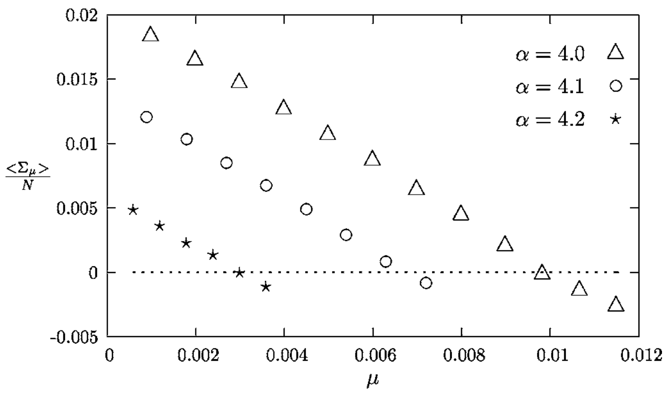

FIG. 7. Average complexity of a subproblem in G_μ for $N = 1000$. Statistical errors are about the size of the points.

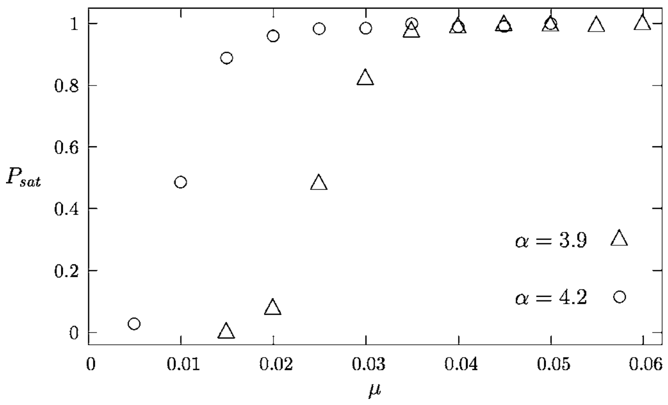

FIG. 8. Satisfiability probability of maximum spanning trees vs μ. The problem size is N=1000 and statistical errors are of order 0.01.

maximum weight on its edges. Notice that taking a nonzero interval to define the edges of maximum weight at each step, along with the randomness in choosing one of them, allows construction of a large number of maximum spanning trees.

We define P_{sat} as the probability that a maximum spanning tree be satisfiable if the algorithm converges. To find out the satisfiability of the subproblem we use a local search algorithm (focused metropolis search) introduced in [14]. Figure 8 displays this quantity vs μ for some values of α. The probability of finding a satisfiable spanning tree is considerable even for a very small μ and tends to unity as μ becomes large values. For instance, if α=4.2 then at μ =0.01 almost half the maximum spanning trees are satisfiable. For these parameters the fraction of converged cases is nearly 0.6 (see Fig. 2). Although the algorithm provides a simple way of constructing satisfiable spanning trees, in general finding them is not an easy task. For example, for a satisfiable problem with parameters (N=100, M=400, K=3), we found no satisfiable spanning tree among 10^7 randomly constructed ones.

Figure 9 shows the satisfiability of maximum spanning trees for some larger problem sizes at α=4.2. Hopefully, by increasing N the satisfiability probability increases for smaller values of μ and obtains its saturation value more rapidly. We hope that this behavior of P_{sat} compensate the

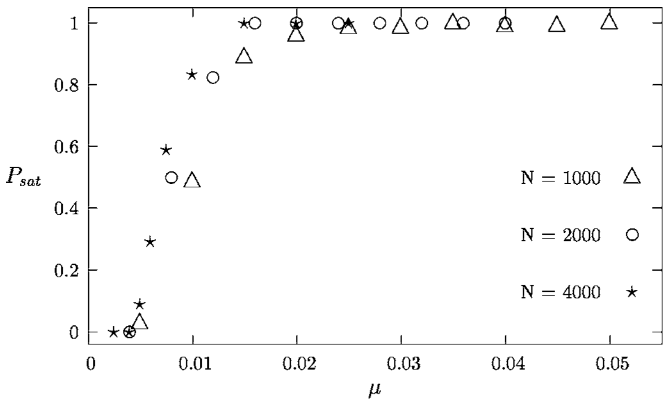

FIG. 9. Satisfiability probability of maximum spanning trees for a few problem sizes. Statistical errors are of order 0.01.

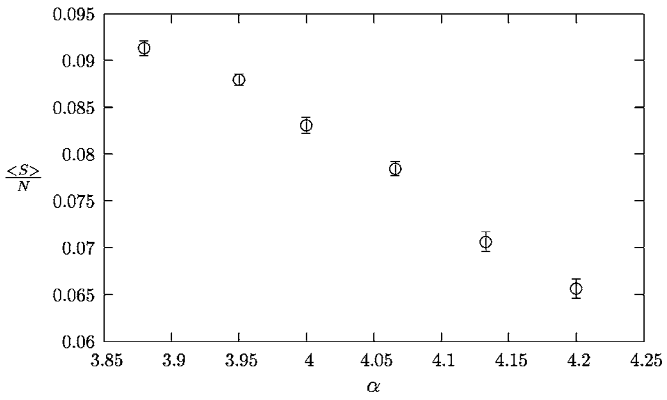

FIG. 10. Entropy of typical satisfiable spanning trees for N =1000 and μ=0.04.

decrease in P_{conv} for larger problem sizes. A look at Figs. 3 and 9 shows that for N=4000, α=4.2 and at μ=0.01 we have $P_{conv} \approx 0.5$ and $P_{sat} \approx 0.7$. It means that of 100 runs we can extract on average 35 satisfiable spanning trees.

Having a satisfiable spanning tree, we can find its solutions (which are also the solutions of the original problem) by any local search algorithm. This, besides the other methods, provides another way of finding the solutions of the original problem. In Fig. 10 we obtained the entropy of typical satisfiable spanning trees by running belief propagation on them. As the figure shows, this entropy decreases linearly with α.

It will be interesting to compare the structural properties of satisfiable spanning trees with those of randomly constructed ones. To this end we obtained the degree distribution of variable and function nodes in the corresponding spanning trees. In Fig. 11 we compare the degree distributions of variables. For function nodes we found no significant difference between the two kinds of spanning trees. However, the degree distribution of variable nodes is slightly broader for the satisfiable spanning trees. There are more low and high degree nodes in these spanning trees.

Another feature of satisfiable spanning trees is their low diameter compared to the random ones; take the node having maximum degree as the center of the spanning tree. The

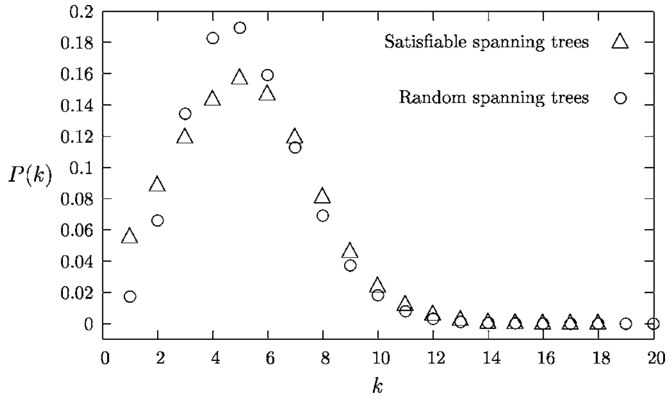

FIG. 11. Degree distribution of variable nodes in satisfiable and random spanning trees. The parameters are N=1000, α=4.2, and μ=0.025. Statistical errors are about the size of the points.

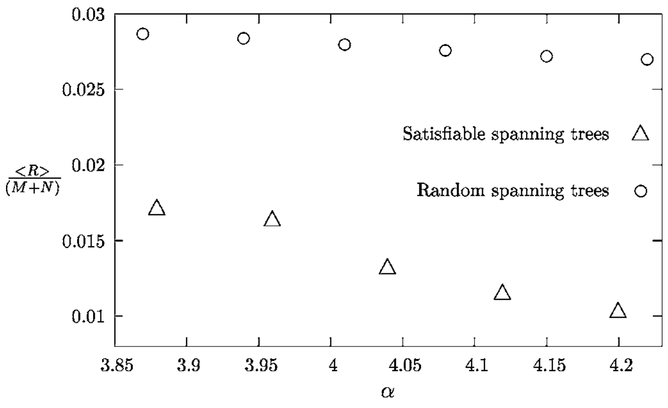

FIG. 12. Average diameter of satisfiable and random spanning trees for $N=1000$ and $\mu=0.05$.

distance of a node from the center is defined as the number of links in the shortest path connecting the center to the node. We define the largest distance from the center as the network's diameter. The diameters of the two kinds of spanning trees have been compared in Fig. 12. Satisfiable spanning trees have a diameter which is almost half the diameter of the random spanning trees.

2. Minimal satisfiable subproblems

A minimal subproblem has the minimum possible number of edges $L=M$ where each function node is connected to at most one variable node. Having such a subproblem it is easy to check its satisfiability. The solutions of a minimal satisfiable subproblem will also be the solutions of the original problem. Moreover, for any solution of the original problem there is at least one minimal satisfiable subproblem. The total number of minimal subproblems is K^M that makes the exhaustive search for satisfiable ones an intractable task.

Suppose that the algorithm for a given μ has been converged and returned the weights w's for all the edges. Among the edges emanating from function node a we choose the one with maximum weight. If there are more than one edge of maximum weight then we select one of them randomly. Notice that we treat all the edges in the ϵ neighborhood of the maximum weight in the same manner. For all the function nodes we make the above choice to construct a minimal subproblem. Then we check the satisfiability of the subproblem and repeat the process for a large number of minimal subproblems obtained from the converged runs of the algorithm. Again P_{sat} is defined as the probability of finding a minimal satisfiable subproblem. This quantity has been displayed in Fig. 13. We observe that even for very small μ, P_{sat} is close to 1. When the parameters are $N=1000$, $M=4200$, $K=3$, this happens at $\mu\approx 0.05$. According to Fig. 2, at these parameters we have to run the algorithm on average 15 times to find a converged case. In Fig. 14 we compare P_{sat} for two different problem sizes. As the figure shows there is no significant difference between the two results.

Having a minimal satisfiable subproblem we will be able to find the solutions directly. Any variable node that has at least one emanating edge is frozen in the obtained set of solutions. In Fig. 15 we have showed the fraction of free variables vs α. Notice that $1-\gamma$ is the fraction of frozen variables and $2^{N\gamma}$ gives the number of solutions in a typical satisfiable subproblem. As expected, the number of free variables decreases as we move closer to the SAT-UNSAT transition. Finally we look at the degree distribution of the variable nodes in the minimal satisfiable subproblems. In Fig. 16 we compare the degree distribution with that of random

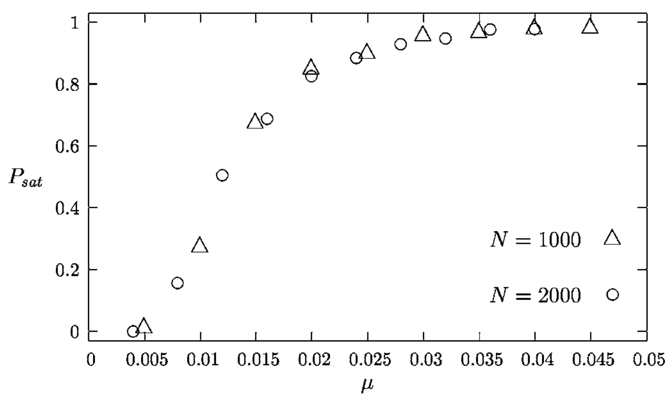

FIG. 14. Satisfiability probability of minimal subproblems at $\alpha=4.2$. Statistical errors are of order 0.01.

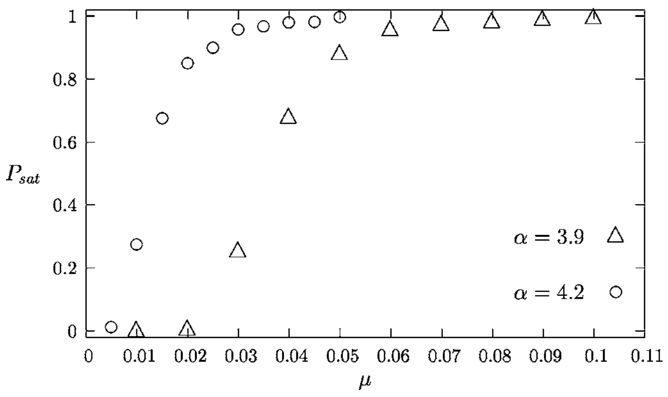

FIG. 13. Satisfiability probability of a minimal subproblem vs μ. Number of variables is $N=1000$ and statistical errors are of order 0.01.

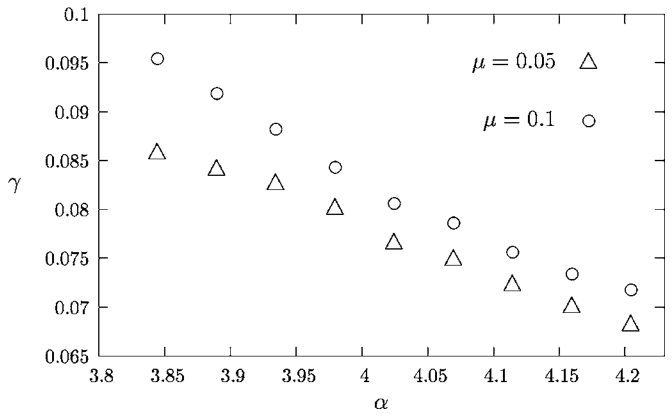

FIG. 15. Fraction of the free variables in the minimal satisfiable subproblems. Number of variables is $N=1000$ and statistical errors are of order 0.001.

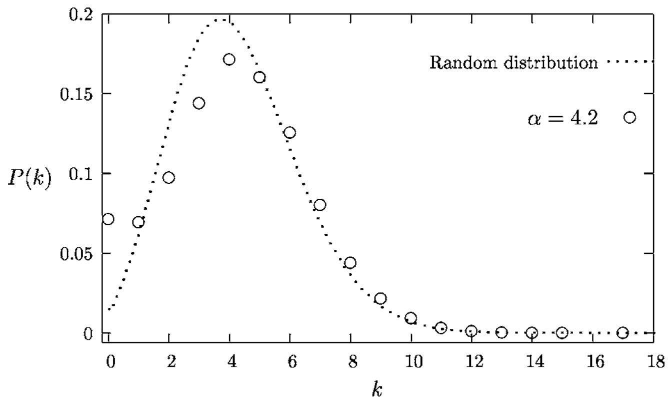

FIG. 16. Comparing degree distribution of the variable nodes in satisfiable and random minimal subproblems. Number of variable nodes is $N=1000$ and statistical errors are of order 0.01.

minimal subproblems where the emanating edges from the function nodes are distributed randomly between the variables. We observe that the real distribution is broader than the random one. Low and high degree nodes have more contribution in the minimal satisfiable subproblems. We encountered the same phenomenon in Fig. 11 that compares the degree distribution of satisfiable spanning trees with that of random ones.

V. CONCLUSION

In summary we showed that there is a way to reduce a random K-satisfiability problem to some simpler subproblems whose solutions are also the solutions of the original problem. To achieve this we modified the known message passing algorithms by assigning some weights to the edges of the factor graph. These weights, that are determined by a self-consistent procedure, lead us to an estimate of the number of satisfiable subproblems. This quantity is interesting to study because we think that it gives a measure of the hardness of the original problem. Indeed, we expect a harder problem to have a smaller number of equivalent subproblems that are easier to solve and at the same time provide the original problem's solutions. The introduced weights also helped us to construct satisfiable subproblems. Finding satisfiable subproblems allowed us to compute the expected value of their entropy and complexity. Studying the behavior of these quantities in the (α,μ) space could in turn reflect the richness of the solution space. We defined $\mu_c(\alpha)$ as the largest value of μ which results in a positive complexity for the associated subproblems. The behavior of μ_c with α near the SAT-UNSAT transition provides another signature of the transition.

As a special case we also constructed the satisfiable spanning trees. Satisfiable spanning trees are interesting because they are the closest treelike structures to the original problem. It is where the belief propagation algorithm turns out to be exact. To search for the relation between the satisfiability and the structure we compared the structural properties of satisfiable spanning trees with those of random spanning trees. We found significant differences in the diameter and the degree distribution of the variables between the two kinds of spanning trees.

As another special case we constructed the minimal satisfiable subproblems and studied some interesting features of their factor graph. Again we focused on the structural properties of minimal satisfiable subproblems. Constructing these subproblems one could easily obtain their solutions which satisfy the original problem too. It would be interesting to compare the efficiency of this method in finding the solutions with those of known algorithms. We also found the fraction of free variables in the minimal satisfiable subproblems. This quantity gives the entropy of minimal satisfiable subproblems and so the number of solutions we could extract from such subproblems.

The modified algorithm studied in this paper can be used, besides the present algorithms, to find the solutions of a constrained satisfaction problem in the SAT phase. Moreover, it provides a way to find the satisfiable subproblems which is not an easy task. Comparing satisfiable subproblems with equivalent random ones might provide some insights about the nature of satisfiable problems and their solutions. It is interesting to find out the relationship between the satisfiability and the structure of a factor graph.

Due to the computational limitations, the results have been restricted to small problem sizes of order 10^3. In this paper we tried to show how increasing the problem size influences the results.

[1] R. G. Gallager, *Low Density Parity-Check Codes* (MIT Press, Cambridge, 1963).
[2] L. Correale, M. Leone, A. Pagnani, M. Weigt, and R. Zecchina, Phys. Rev. Lett. **96**, 018101 (2006).
[3] M. Mezard, G. Parisi, and M. A. Virasoro, *Spin Glass Theory and Beyond* (World Scientific, Singapore, 1987).
[4] S. Cook, *Proceedings 3rd Annual ACM Symposium on Theory of Computing* (Association of Computing Machines, New York, 1971), p. 151.
[5] C. H. Papadimitriou, *Computational Complexity* (Addison-Wesley, Reading, MA, 1994).
[6] S. Kirkpatrick and B. Selman, Science **264**, 1297 (1994).
[7] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, IEEE Trans. Inf. Theory **42**, 498 (2001).
[8] *Local Search for Combinatorial Optimization*, edited by E. Aarts and J. K. Lenstra (Wiley, New York, 1997).
[9] M. Mezard and R. Zecchina, Phys. Rev. E **66**, 056126 (2002).
[10] A. Braunstein, M. Mezard, and R. Zecchina, Random Struct. Algorithms **27**, 201 (2005).
[11] A. Ramezanpour and S. Moghimi-Araghi, J. Phys. A **39**, 4901 (2006).
[12] A. Montanari, G. Parisi, and F. Ricci-Tersenghi, J. Phys. A **37**, 2073 (2004).
[13] S. Mertens, M. Mezard, and R. Zecchina, Random Struct. Algorithms **28**, 340 (2005).
[14] S. Seitz, M. Alava, and P. Orponen, J. Stat. Mech.: Theory Exp. (2005) P06006.