Journal of Machine Learning Research 16 (2015) 1249-1274 Submitted 4/13; Revised 9/14; Published 7/15

Perturbed Message Passing for Constraint Satisfaction Problems

Siamak Ravanbakhsh MRAVANBA@UALBERTA.CA
Russell Greiner RGREINER@UALBERTA.CA
Department of Computing Science
University of Alberta
Edmonton, AB T6E 2E8, CA

Editor: Alexander Ihler

Abstract

We introduce an efficient message passing scheme for solving Constraint Satisfaction Problems (CSPs), which uses stochastic perturbation of Belief Propagation (BP) and Survey Propagation (SP) messages to bypass decimation and directly produce a single satisfying assignment. Our first CSP solver, called *Perturbed Belief Propagation*, smoothly interpolates two well-known inference procedures; it starts as BP and ends as a Gibbs sampler, which produces a single sample from the set of solutions. Moreover we apply a similar perturbation scheme to SP to produce another CSP solver, *Perturbed Survey Propagation*. Experimental results on random and real-world CSPs show that Perturbed BP is often more successful and at the same time tens to hundreds of times more efficient than standard BP guided decimation. Perturbed BP also compares favorably with state-of-the-art SP-guided decimation, which has a computational complexity that generally scales exponentially worse than our method (w.r.t. the cardinality of variable domains and constraints). Furthermore, our experiments with random satisfiability and coloring problems demonstrate that Perturbed SP can outperform SP-guided decimation, making it the best incomplete random CSP-solver in difficult regimes.

Keywords: constraint satisfaction problem, message passing, belief propagation, survey propagation, Gibbs sampling, decimation

1. Introduction

Probabilistic Graphical Models (PGMs) provide a common ground for recent convergence of themes in computer science (artificial neural networks), statistical physics of disordered systems (spin-glasses) and information theory (error correcting codes). In particular, message passing methods have been successfully applied to obtain state-of-the-art solvers for Constraint Satisfaction Problems (Mézard et al., 2002)

The PGM formulation of a CSP defines a uniform distribution over the set of solutions, where each unsatisfying assignment has a zero probability. In this framework, solving a CSP amounts to producing a sample from this distribution. To this end, usually an inference procedure estimates the marginal probabilities, which suggests an assignment to a subset of the most biased variables. This process of sequentially fixing a subset of variables, called *decimation*, is repeated until all variables are fixed to produce a solution. Due to inaccuracy of the marginal estimates, this procedure gives an incomplete solver (Kautz et al., 2009), in the sense that the procedure's failure is not a certificate of unsatisfiability. An alternative approach is to use message passing to guide a search procedure that can back-track if a dead-end is reached (*e.g.*, Kask et al., 2004; Parisi, 2003). Here using a branch and bound technique and relying on exact solvers, one may also determine when a CSP is unsatisfiable.

The most common inference procedure for this purpose is Belief Propagation (Pearl, 1988). However, due to geometric properties of the set of solutions (Krzakala et al., 2007) as well as the complications from the decimation procedure (Coja-Oghlan, 2011; Kroc et al., 2009), BP-guided decimation fails on difficult instances. The study of the change in the geometry of solutions has lead to Survey Propagation (Braunstein et al., 2002) which is a powerful message passing procedure that is slower than BP (per iteration) but typically remains convergent, even in many situations when BP fails to converge.

Using decimation, or other search schemes that are guided by message passing, usually requires estimating marginals or partition functions, which is harder than producing a single solution (Valiant, 1979). This paper introduces a message passing scheme to eliminate this requirement, therefore also avoiding the complications of applying decimation. Our alternative has advantage over both BP- and SP-guided decimation when applied to solve CSPs. Here we consider BP and Gibbs Sampling (GS) updates as operators—Φ and Ψ respectively—on a set of messages. We then consider inference procedures that are convex combination (*i.e.*, $\gamma\Psi + (1-\gamma)\Phi$) of these two operators. Our CSP solver, Perturbed BP, starts at $\gamma = 0$ and ends at $\gamma = 1$, smoothly changing from BP to GS, and finally producing a sample from the set of solutions. This change amounts to stochastic biasing the BP messages towards the current estimate of marginals, where this random bias increases in each iteration. This procedure is often much more efficient than BP-guided decimation (BP-dec) and sometimes succeeds where BP-dec fails. Our results on random CSPs (rCSPs) show that Perturbed BP is competitive with SP-guided decimation (SP-dec) in solving difficult random instances.

Since SP can be interpreted as BP applied to an "auxiliary" PGM (Braunstein et al., 2005), we can apply the same perturbation scheme to SP, which we call Perturbed SP. Note that this system, also, does not perform decimation and directly produce a solution (without using local search). Our experiments show that Perturbed SP is often more successful than both SP-dec and Perturbed BP in finding satisfying assignments.

Stochastic variations of BP have been previously proposed to perform inference in graphical models (*e.g.*, Ihler and Mcallester, 2009; Noorshams and Wainwright, 2013). However, to our knowledge, Perturbed BP is the first method to directly combine GS and BP updates.

In the following, Section 1.1 introduces PGM formulation of CSP using factor-graph notation. Section 1.2 reviews the BP equations and decimation procedure, then Section 1.3 casts GS as a message update procedure. Section 2 introduces Perturbed BP as a combination of GS and BP. Section 2.1 compares BP-dec and Perturbed BP on benchmark CSP instances, showing that our method is often several folds faster and more successful in solving CSPs. Section 3 overviews the geometric properties of the set of solutions of rCSPs, then reviews first order Replica Symmetry Breaking Postulate and the resulting SP equations for CSP. Section 3.2 introduces Perturbed SP and Section 3.3 presents our experimental results for random satisfiability and random coloring instances close to the unsatisfiability threshold. Finally, Section 3.4 further discusses the behavior of decimation and perturbed BP in the light of a geometric picture of the set of solutions and the experimental results.

1.1 Factor Graph Representation of CSP

Let $x = (x_1, x_2, \ldots, x_N)$ be a tuple of N discrete variables $x_i \in \mathcal{X}_i$, where each $\mathcal{X}_i$ is the domain of x_i. Let $I \subseteq \mathcal{N} = \{1, 2, \ldots, N\}$ denote a subset of variable indices and $x_I = \{x_i \,|\, i \in I\}$ be the (sub)tuple of variables in x indexed by the subset I. Each constraint $C_I(x_I) : \left(\prod_{i \in I} \mathcal{X}_i\right) \to \{0, 1\}$ maps an assignment to 1 *iff* that assignment satisfies that constraint. Then the normalized product of all constraints defines a uniform distribution over solutions

$$\mu(x) \quad \triangleq \quad \frac{1}{Z} \prod_I C_I(x_I) \tag{1}$$

where the partition function $Z = \sum_{\mathcal{X}} \prod_I C_I(x_I)$ is equal to the number of solutions.[1] Notice that $\mu(x)$ is non-zero *iff* all of the constraints are satisfied—that is x is a solution. With slight abuse of notation we will use probability density and probability distribution interchangeably.

Example 1 (q-COL:) *Here, $x_i \in \mathcal{X}_i = \{1, \ldots, q\}$ is a q-ary variable for each $i \in \mathcal{N}$, and we have M constraints; each constraint $C_{i,j}(x_i, x_j) = 1 - \delta(x_i, x_j)$ depends only on two variables and is satisfied iff the two variables have different values (colors). Here $\delta(x, x')$ is equal to 1 if $x = x'$ and 0 otherwise.*

This model can be conveniently represented as a bipartite graph, known as a *factor graph* (Kschischang et al., 2001), which includes two sets of nodes: variable nodes x_i, and constraint (or factor) nodes C_I. A variable node i (note that we will often identify a variable "x_i" with its index "i") is connected to a constraint node I if and only if $i \in I$. We will use ∂ to denote the neighbors of a variable or constraint node in the factor graph—that is $\partial I = \{i \mid i \in I\}$ (which is the set I) and $\partial i = \{I \mid i \in I\}$. Finally we use Δi to denote the Markov blanket of node x_i ($\Delta i = \{j \in \partial I \mid I \in \partial i, j \neq i\}$).

The marginal of $\mu(\cdot)$ for variable x_i is defined as

$$\mu(x_i) \triangleq \sum_{\mathcal{X}_{\mathcal{N} \setminus i}} \mu(x)$$

where the summation above is over all variables but x_i. Below, we use $\widehat{\mu}(x_i)$ to denote an *estimate* of this marginal. Finally, we use $\mathcal{S}$ to denote the (possibly empty) set of solutions $\mathcal{S} = \{x \in \mathcal{X} \mid \mu(x) \neq 0\}$.

Example 2 (κ-SAT:) *All variables are binary ($\mathcal{X}_i = \{True, False\}$) and each clause (constraint C_I) depends on $\kappa = |\partial I|$ variables. A clause evaluates to 0 only for a single assignment out of 2^κ possible assignment of variables (Garey and Johnson, 1979).*

Consider the following 3-SAT problem over 3 variables with 5 clauses

$$\begin{aligned} SAT(x) = &\underbrace{(\neg x_1 \vee \neg x_2 \vee x_3)}_{C_1} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3)}_{C_2} \wedge \underbrace{(x_1 \vee \neg x_2 \vee x_3)}_{C_3} \\ &\wedge \underbrace{(\neg x_1 \vee x_2 \vee \neg x_3)}_{C_4} \wedge \underbrace{(x_1 \vee \neg x_2 \vee \neg x_3)}_{C_5}. \end{aligned} \tag{2}$$

1. For Equation (1) to remain valid when the CSP is unsatisfiable, we define $\frac{0}{0} \triangleq 0$.

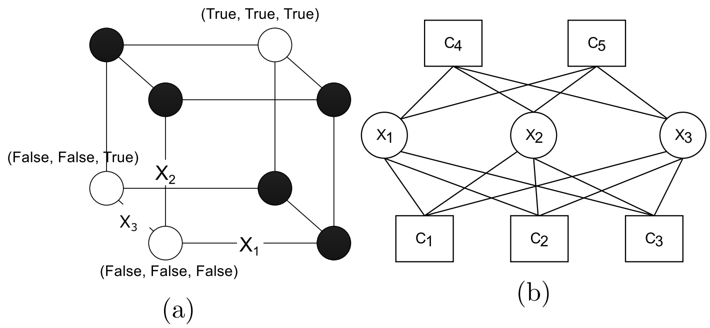


Figure 1: (a) The set of all possible assignments to 3 variables. The solutions to the 3-SAT problem of Equation (2) are in white circles. (b) The factor-graph corresponding to the 3-SAT problem of Equation (2). Here each factor prohibits a single assignment.

The constraint corresponding to the first clause takes the value 1*, except for* $x = \{True, True, False\}$*, in which case it is equal to* 0*. The set of solutions for this problem is given by* $\mathcal{S} = \big\{(True, True, True),\ (False, False, False),\ (False, False, True)\big\}$*. Figure 1 shows the solutions as well as the corresponding factor graph.*[2]

1.2 Belief Propagation-guided Decimation

Belief Propagation (Pearl, 1988) is a recursive update procedure that sends a sequence of messages from variables to constraints ($\nu_{i\to I}$) and vice-versa ($\nu_{I\to i}$)

$$\nu_{i\to I}(x_i) \ \propto \prod_{J\in\partial i\setminus I} \nu_{J\to i}(x_i) \tag{3}$$

$$\nu_{I\to i}(x_i) \ \propto \sum_{x_{I\setminus i}\in\mathcal{X}_{\partial I\setminus i}} C_I(x_I) \prod_{j\in\partial I\setminus i} \nu_{j\to I}(x_j) \tag{4}$$

where $J \in \partial i \setminus I$ refers to all the factors connected to variable x_i, except for factor C_I. Similarly the summation in Equation (4) is over $\mathcal{X}_{\partial I\setminus i}$, means we are summing out all x_j that are connected to C_I (*i.e.*, x_j *s.t.* $j \in I \setminus i$) except for x_i.

The messages are typically initialized to a uniform or a random distribution. This recursive update of messages is usually performed until convergence—*i.e.*, until the maximum change in the value of all messages, from one iteration to the next, is negligible (*i.e.*, below some small ϵ). At any point during the updates, the estimated marginal probabilities are given by

$$\widehat{\mu}(x_i) \ \propto \prod_{J\in\partial i} \nu_{J\to i}(x_i). \tag{5}$$

2. In this simple case, we could combine all the constraints into a single constraint over 3 variables and simplify the factor graph. However, in general SAT, this cost-saving simplification is often not possible.

In a factor graph without loops, each BP message summarizes the effect of the (sub-tree that resides on the) sender-side on the receiving side.

Example 3 *Applying BP to the 3-SAT problem of Equation* (2) *takes 20 iterations to converge (i.e., for the maximum change in the marginals to be below* $\epsilon = 10^{-9}$*). Here the message,* $\nu_{C_1\to 1}(x_1)$*, from* $C1$ *to* x_1 *is*

$$\nu_{C_1\to 1}(x_1) \quad \propto \quad \sum_{x_{2,3}} C_1(x_{1,2,3})\, \nu_{2\to C_1}(x_2)\, \nu_{3\to C_1}(x_3)$$

and similarly, the message in the opposite direction, $\nu_{1\to C_1}(x_1)$*, is defined as*

$$\nu_{1\to C_1}(x_1) \quad \propto \quad \nu_{C_2\to 1}(x_1)\, \nu_{C_3\to 1}(x_1)\, \nu_{C_4\to 1}(x_1)\, \nu_{C_5\to 1}(x_1).$$

Here BP gives us the following approximate marginals: $\widehat{\mu}(x_1 = True) = \widehat{\mu}(x_2 = True) = .319$ *and* $\widehat{\mu}(x_3 = True) = .522$*. From the set of solutions, we know that the correct marginals are* $\widehat{\mu}(x_1 = True) = \widehat{\mu}(x_2 = True) = 1/3$ *and* $\widehat{\mu}(x_3 = True) = 2/3$*. The error of BP is caused by influential loops in the factor-graph of Figure 1(b). Here the error is rather small; it can be arbitrarily large in some instances or BP may not converge at all.*

The time complexity of BP updates of Equation (3) and Equation (4), for each of the messages exchanged between i and I, are $\mathcal{O}(|\partial i|\; |\mathcal{X}_i|)$ and $\mathcal{O}(|\mathcal{X}_I|)$ respectively. We may reduce the time complexity of BP by synchronously updating all the messages $\nu_{i\to I}\ \forall I \in \partial i$ that leave node i. For this, we first calculate the beliefs $\widehat{\mu}(x_i)$ using Equation (5) and produce each $\nu_{i\to I}$ using

$$\nu_{i\to I}(x_i) \;\propto\; \frac{\widehat{\mu}(x_i)}{\nu_{I\to i}(x_i)}. \tag{6}$$

Note than we can substitute Equation (4) into Equation (3) and Equation (5) and only keep variable-to-factor messages. After this substitution, BP can be viewed as a fixed-point iteration procedure that repeatedly applies the operator $\Phi(\{\nu_{i\to I}\}) \triangleq \{\Phi_{i\to I}(\{\nu_{j\to J}\}_{j\in\Delta i, J\in\partial i\setminus I})\}_{i,I\in\partial i}$ to the set of messages in hope of reaching a fixed point—that is

$$\nu_{i\to I}(x_i) \;\propto\; \prod_{J\in\partial i\setminus I} \sum_{\mathcal{X}_{\partial J\setminus i}} C_J(x_J) \prod_{j\in\partial J\setminus i} \nu_{j\to J}(x_j) \;\triangleq\; \Phi_{i\to I}(\{\nu_{j\to J}\}_{j\in\Delta i, J\in\partial i\setminus I})(x_i) \tag{7}$$

and therefore Equation (5) becomes

$$\widehat{\mu}(x_i) \quad \propto \quad \prod_{I\in\partial i} \sum_{\mathcal{X}_{\partial I\setminus i}} C_I(x_I) \prod_{j\in\partial I\setminus i} \nu_{j\to I}(x_j) \tag{8}$$

where $\Phi_{i\to I}$ denotes individual message update operators. We let operator $\Phi(.)$ denote the set of these $\Phi_{i\to I}$ operators.

1.2.1 Decimation

The decimation procedure can employ BP (or SP) to solve a CSP. We refer to the corresponding method as BP-dec (or SP-dec). After running the inference procedure and obtaining $\widehat{\mu}(x_i)$, $\forall i$, the decimation procedure uses a heuristic approach to select the most biased variables (or just a random subset) and fixes these variables to their most biased values (or a random $\widehat{x}_i \sim \widehat{\mu}(x_i)$). If it selects a fraction ρ of remaining variables to fix after each convergence, this multiplies an additional $\log_{\frac{1}{\rho}}(N)$ to the linear (in N) cost[3] for each iteration of BP (or SP). The following algorithm 1 summarizes BP-dec with a particular scheduling of updates:

input : factor-graph of a CSP
output: a satisfying assignment x^* if an assignment was found. UNSATISFIED otherwise

1 initialize the messages
2 $\widetilde{\mathcal{N}} \leftarrow \mathcal{N}$ (set of all variable indices)
3 **while** $\widetilde{\mathcal{N}}$ *is not empty* **do** `// decimation loop`
4
5   **repeat** `// BP loop`
6
7   **foreach** $i \in \widetilde{\mathcal{N}}$ **do**
8   calculate messages $\{\nu_{I\to i}\}_{I\in\partial i}$ using Equation (4)
9   **if** $\{\nu_{I\to i}\}_{I\in\partial i}$ *are contradictory* **then**
        return : UNSATISFIED
10   calculate marginal $\widehat{\mu}(x_i)$ using Equation (5)
11   calculate messages $\{\nu_{i\to I}\}_{I\in\partial i}$ using Equation (3) or Equation (6)
12   **until** *convergence*
13   select $\mathcal{B} \subseteq \widetilde{\mathcal{N}}$ using $\{\widehat{\mu}(x_i)\}_{i\in\widetilde{\mathcal{N}}}$
14   fix $x^*_j \leftarrow \arg_{x_j} \max \widehat{\mu}(x_j) \quad \forall j \in \mathcal{B}$
15   reduce the constraints $\{C_I\}_{I\in\partial j}$ for every $j \in \mathcal{B}$

return : $x^* = (x^*_1, \ldots, x^*_N)$

Algorithm 1: Belief Propagation-guided Decimation (BP-dec)

The condition of line 9 is satisfied iff the product of incoming messages to node i is 0 for all $x_i \in \mathcal{X}_i$. This means that neighboring constraints have strict disagreement about the value of x_i and the decimation has found a contradiction. This contradiction can happen because, either (I) there is no solution for the reduced problem even if the original problem had a solution, or (II) the reduced problem has a solution but the BP messages are inaccurate.

Example 4 *To apply BP-dec to previous example, we first calculate BP marginals, as shown in the example above. Here $\widehat{\mu}(x_1)$ and $\widehat{\mu}(x_2)$ have the highest bias. By fixing the*

3. Assuming the number of edges in the factor graph are in the order of N. In general, using synchronous update of Equation (6) and assuming a constant factor cardinality, $|\partial I|$, the cost of each iteration is $\mathcal{O}(E)$, where E is the number of edges in the factor-graph.

value of x_1 *to* $False$*, the SAT problem of Equation* (2) *collapses to*

$$SAT(x_{\{2,3\}})|_{x_1=False} \;=\; (\neg x_2 \vee x_3) \wedge (\neg x_2 \vee \neg x_3).$$

BP-dec applies BP again to this reduced problem, which give $\widehat{\mu}(x_2 = True) = .14$ *(note here that* $\mu(x_2 = True) = 0$*) and* $\widehat{\mu}(x_3 = True) = 1/2$*. By fixing* x_2 *to* $False$*, another round of decimation yields a solution* $x^* = (False, False, True)$*.*

1.3 Gibbs Sampling as Message Update

Gibbs Sampling (GS) is a Markov Chain Monte Carlo (MCMC) inference procedure (Andrieu et al., 2003) that can produce a set of samples $\widehat{x}[1], \ldots, \widehat{x}[L]$ from a given PGM. We can then recover the marginal probabilities, as empirical expectations

$$\widehat{\mu}^L(x_i) \quad \propto \quad \frac{1}{L}\sum_{n=1}^{L} \delta(\widehat{x}[n]_i, x_i). \tag{9}$$

Our algorithm only considers a single particle $\widehat{x} = \widehat{x}[1]$. GS starts from a random initial state $\widehat{x}^{(t=0)}$ and at each time-step t, updates each $\widehat{x}_i$ by sampling from:

$$\widehat{x}_i^{(t)} \;\sim\; \mu(x_i) \quad \propto \quad \prod_{I \in \partial i} C_I(x_i, \widehat{x}_{\partial I \setminus i}^{(t-1)}) \tag{10}$$

If the Markov chain satisfies certain basic properties (Robert and Casella, 2005), $x_i^{(\infty)}$ is guaranteed to be an unbiased sample from $\mu(x_i)$ and therefore our marginal estimate, $\widehat{\mu}^L(x_i)$, becomes exact as $L \to \infty$.

In order to interpolate between BP and GS, we establish a correspondence between a particle in GS and a set of variable-to-factor messages—*i.e.*, $\widehat{x} \Leftrightarrow \{\nu_{i\to I}(.)\}_{i, I \in \partial i}$. Here all the messages leaving variable x_i are equal to a δ-function defined based on $\widehat{x}_i$

$$\nu_{i\to I}(x_i) \;=\; \delta(x_i, \widehat{x}_i) \quad \forall I \in \partial i.$$

We define the random GS operator $\Psi = \{\Psi_i\}_i$ and rewrite the GS update of Equation (10) as

$$\nu_{i\to I}(x_i) \quad \triangleq \quad \Psi_i(\{\nu_{j\to J}(x_j)\}_{j \in \Delta i, J \in \partial i})(x_i) \quad = \quad \delta(\widehat{x}_i, x_i) \tag{11}$$

where $\widehat{x}_i$ is sampled from

$$\begin{aligned} \widehat{x}_i \;\sim\; \widehat{\mu}(x_i) \quad &\propto \quad \prod_{J \in \partial i} C_I(x_i, \widehat{x}_{\partial I \setminus i}) \\ &\propto \quad \prod_{I \in \partial i} \sum_{\mathcal{X}_{\partial I \setminus i}} C_I(x_I) \prod_{j \in \partial I \setminus i} \nu_{j\to I}(x_j). \end{aligned} \tag{12}$$

Note that Equation (12) is identical to BP estimate of the marginal Equation (8). This equality is a consequence of the way we have defined messages in the GS update and allows us to combine BP and GS updates in the following section.

2. Perturbed Belief Propagation

Here we introduce an alternative to decimation that does not require repeated application of inference. The idea is to use a linear combination of BP and GS operators (Equation 7 and Equation 11) to update the messages

$$\Gamma(\{\nu_{i\to I}\}) \triangleq \gamma\ \Psi(\{\nu_{i\to I}\}) + (1-\gamma)\Phi(\{\nu_{i\to I}\}).$$

The Perturbed BP operator $\Gamma = \{\Gamma_{i\to I}\}_{i,I\in\partial i}$ updates each message by calculating the outgoing message according to BP and GS operators and linearly combines them to get the final message. During T iterations of Perturbed BP, the parameter γ is gradually and linearly changed from 0 towards 1. Algorithm 2 below summarizes this procedure.

input : factor graph of a CSP, number of iterations T
output: a satisfying assignment x^* if an assignment was found. UNSATISFIED otherwise

1 initialize the messages
2 $\gamma \leftarrow 0$
3 $\widetilde{\mathcal{N}} \leftarrow \mathcal{N}$ (set of all variable indices)
4 **for** $t = 1$ *to* T **do**
5   **foreach** *variable* x_i **do**
6   calculate $\nu_{I\to i}$ using Equation (4) $\forall I \in \partial i$
7   **if** $\{\nu_{I\to i}\}_{I\in\partial i}$ *are contradictory* **then**
              return : UNSATISFIED
8   calculate marginals $\widehat{\mu}(x_i)$ using Equation (12)
9   calculate BP messages $\nu_{i\to I}$ using Equation (3) or Equation (6) $\forall I \in \partial i$.
10   sample $\widehat{x}_i \sim \widehat{\mu}(x_i)$
11   combine BP and Gibbs sampling messages:
           $\nu_{i\to I} \ \leftarrow\ \gamma\ \nu_{i\to I} \ +\ (1-\gamma)\ \delta(x_i, \widehat{x}_i)$
12     $\gamma \leftarrow \gamma + \frac{1}{T-1}$
return : $x^* = \{x_1^*, \ldots, x_N^*\}$

Algorithm 2: Perturbed Belief Propagation

In step 7, if the product of incoming messages is 0 for all $x_i \in \mathcal{X}_i$ for some i, different neighboring constraints have strict disagreement about x_i; therefore this run of Perturbed BP will not be able to satisfy this CSP. Since the procedure is inherently stochastic, if the CSP is satisfiable, re-application of the same procedure to the problem may avoid this specific contradiction.

2.1 Experimental Results on Benchmark CSP

This section compares the performance of BP-dec and Perturbed BP on benchmark CSPs. We considered CSP instances from XCSP repository (Roussel and Lecoutre, 2009; Lecoutre, 2013), without global constraints or complex domains.[4]

4. All instances with intensive constraints (*i.e.*, functional form) were converted into extensive format for explicit representation using dense factors. We further removed instances containing constraints with

We used a convergence threshold of $\epsilon = .001$ for BP and terminated if the threshold was not reached after $T = 10 \times 2^{10} = 10,240$ iterations. To perform decimation, we sort the variables according to their bias and fix ρ fraction of the most biased variables in each iteration of decimation. This fraction, ρ, was initially set to 100%, and it was divided by 2 each time BP-dec failed on the same instance. BP-dec was repeatedly applied using the reduced ρ, at most 10 times, unless a solution was reached (that is $\rho = .1\%$ at final attempt).

For Perturbed BP, we set $T = 10$ at the starting attempt, which was increased by a factor of 2 in case of failure. This was repeated at most 10 times, which means Perturbed BP used $T = 10,240$ at its final attempt. Note that Perturbed BP at most uses the same number of iterations as the maximum iterations per single iteration of decimation in BP-dec.

Figure 2(a,b) compares the time and iterations of BP-dec and Perturbed BP for successful attempts where both methods satisfied an instance. The result for individual problem-sets is reported in the appendix.

Empirically, we found that Perturbed BP both solved (slightly) more instances than BP-dec (284 vs 253), and was (hundreds of times) more efficient: while Perturbed BP required only 133 iterations on average, BP-dec required an average of 41,284 iterations for successful instances.

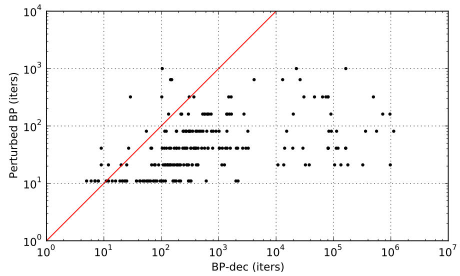
We also ran BP-dec on all the benchmarks with maximum number of iterations set to $T = 1000$ and $T = 100$ iterations. This reduced the number of satisfied instances to 249 for $T = 1000$ and 247 for $T = 100$, but also reduced the average number of iterations to 1570 and 562 respectively, which are still several folds more expensive than Perturbed BP. Figure 2(c-f) compare the time and iterations used by BP-dec in these settings with that of Perturbed BP, when both methods found a satisfying assignment. See the appendix for a more detailed report on these results.

3. Critical Phenomena in Random CSPs

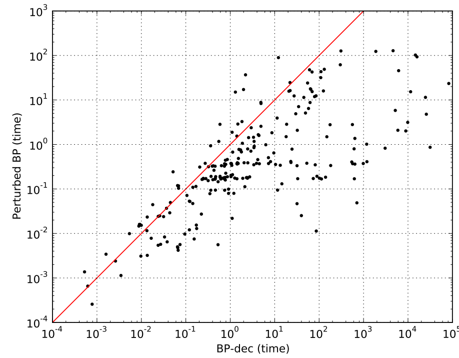
Random CSP (rCSP) instances have been extensively used in order to study the properties of combinatorial problems (Mitchell et al., 1992; Achioptas and Sorkin, 2000; Krzakala et al., 2007) as well as in analysis and design of algorithms (*e.g.*, Selman et al., 1994; Mézard et al., 2002). Random CSPs are closely related to spin-glasses in statistical physics (Kirkpatrick and Selman, 1994; Fu and Anderson, 1986). This connection follows from the fact that the Hamiltonian of these spin-glass systems resembles the objective functions in many combinatorial problems, which decompose to pairwise (or higher order) interactions, allowing for a graphical representation in the form of a PGM. Here message passing methods, such as belief propagation (BP) and survey propagation (SP), provide consistency conditions on locally tree-like neighborhoods of the graph.

The analogy between a physical system and computational problem extends to their critical behavior where computation relates to dynamics (Ricci-Tersenghi, 2010). In com-

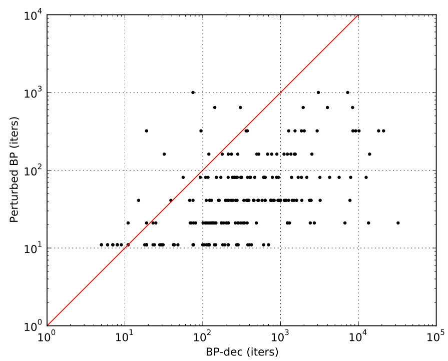
more than 10^6 entries in their tabular form. We also discarded instances that collectively had more than 10^8 entries in the dense tabular form of their constraints. Since our implementation represents all factors in a dense tabular form, we had to remove many instances because of their large factor size. We anticipate that Perturbed BP and BP-dec could probably solve many of these instances using a sparse representation.



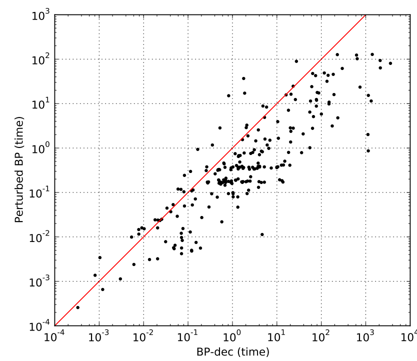
(a) iterations



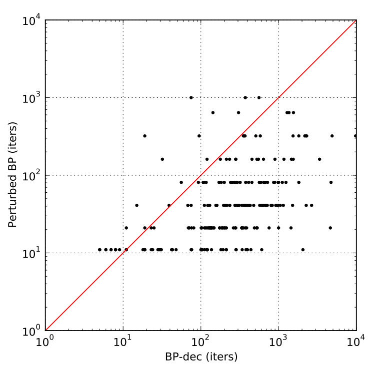
(b) time (seconds)



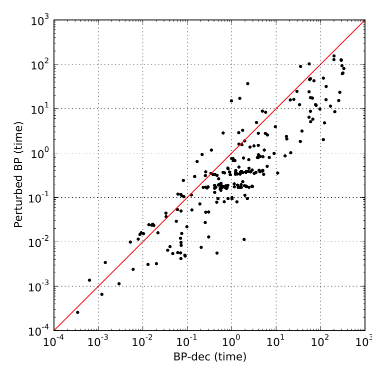
(c) iterations for $T = 1000$



(d) time (seconds) for $T = 1000$



(e) iterations for $T = 100$



(f) time (seconds) for $T = 100$

Figure 2: Comparison of time and number of iterations used by BP-dec and Perturbed BP in benchmark instances where both methods found satisfying assignments. (a,b) Maximum number of BP iterations per iteration of decimation is $T = 10240$, equal to the maximum iterations used by Perturbed BP. (c,d) Maximum number of iterations for BP in BP-dec is reduced to $T = 1000$. (e,f) Maximum number of iterations for BP in BP-dec is further reduced to $T = 100$.

puter science, this critical behavior is related to the time-complexity of algorithms employed to solve such problems, while in spin-glass theory this translates to dynamics of glassy state, and exponential relaxation times (Mézard et al., 1987). In fact, this connection has been used to attempt to prove the conjecture that $\mathcal{P}$ is not equal to $\mathcal{NP}$ (Deolalikar, 2010).

Studies of rCSP, as a critical phenomena, focus on the geometry of the solution space as a function of the problem's difficulty, where rigorous (*e.g.*, Achlioptas and Coja-Oghlan, 2008; Cocco et al., 2003) and non-rigorous (*e.g.*, cavity method of Mézard and Parisi, 2001, 2003) analyses have confirmed the same geometric picture.

When working with large random instances, a scalar α associated with a problem instance, a.k.a. control parameter—for example, the clause to variable ratio in SAT—can characterize that instance's difficulty (*i.e.*, larger control parameter corresponds to a more difficult instance) and in many situations it characterizes a sharp transition from satisfiability to unsatisfiability (Cheeseman et al., 1991).

Example 5 (Random κ-SAT) *Random κ-SAT instance with N variables and $M = \alpha N$ constraints are generated by selecting κ variables at random for each constraint. Each constraint is set to zero (i.e., unsatisfied) for a single random assignment (out of 2^κ). Here α is the control parameter.*

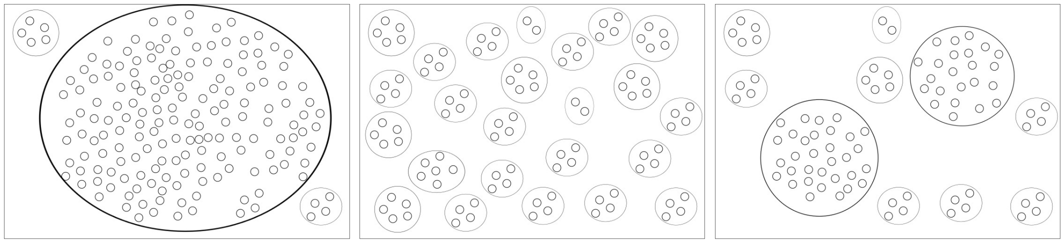
Example 6 (Random q-COL) *The control parameter for a random q-COL instances with N variables and M constraints is its average degree $\alpha = \frac{2M}{N}$. We consider Erdős-Rény random graphs and generate a random instance by sequentially selecting two distinct variables out of N at random to generate each of M edges. For large N, this is equivalent to selecting each possible factor with a fixed probability, which means the nodes have Poisson degree distribution $\mathbb{P}(|\partial i| = d) \propto e^{-\alpha}\alpha^d$.*

While there are tight bounds for some problems (*e.g.*, Achlioptas et al., 2005), finding the exact location of this transition for different CSPs is still an open problem. Besides transition to unsatisfiability, these analyses has revealed several other (phase) transitions (Krzakala et al., 2007). Figure 3(a)-(c) shows how the geometry of the set of solutions changes by increasing the control parameter.

Here we enumerate various phases of the problem for increasing values of the control parameter: **(a)** In the so-called *Replica Symmetric* (RS) phase, the symmetries of the set of solutions (a.k.a. ground states) reflect the trivial symmetries of problem w.r.t. variable domains. For example, for q-COL the set of solutions is symmetric w.r.t. swapping all red and blue assignment. In this regime, the set of solutions form a giant cluster (*i.e.*, a set of neighboring solutions), where two solutions are considered neighbors when their Hamming distance is one (Achlioptas and Coja-Oghlan, 2008) or non-divergent with number of variables (Mézard and Parisi, 2003). Local search methods (*e.g.*, Selman et al., 1994) and BP-dec can often efficiently solve random CSPs that belong to this phase.

(b) In *clustering* or *dynamical* transition (1dRSB[5]), the set of solutions decomposes into an exponential number of distant clusters. Here two clusters are distant if the Hamming distance between their respective members is divergent (*e.g.*, linear) in the number

5. 1dRSB stands for 1st order dynamical RSB. The term Replica Symmetry Breaking (RSB) originates from the technique—*i.e.*, Replica trick (Mézard et al. 1987)—that was first used to analyze this setting. According to RSB, the trivial symmetries of the problem do not characterize the clusters of solution.



(a) Replica Symmetric (b) clustering (c) condensation

Figure 3: A 2-dimensional schematic view of how the set of solutions of CSP varies as we increase the control parameter α from (a) replica symmetric phase to (b) clustering phase to (c) condensation phase. Here small circles represent solutions and the bigger circles represent clusters of solutions. Note that this view is very simplistic in many ways; for example, the total number of solutions and the size of clusters should generally decrease from (a) to (c).

of variables. **(c)** In the *condensation* phase transition (1sRSB[6]), the set of solutions condenses into a few dominant clusters. Dominant clusters have roughly the same number of solutions and they collectively contain almost all of the solutions. While SP can be used even within the condensation phase, BP usually fails to converge in this regime. However each cluster of solutions in the clustering and condensation phase is a valid fixed-point of BP, which is called a "quasi-solution" of BP. **(d)** A *rigidity* transition (not included in Figure 3) identifies a phase in which a finite portion of variables are fixed within dominant clusters. This transition triggers an exponential decrease in the total number of solutions, which leads to **(e)** unsatisfiability transition.[7] This rough picture summarizes first order Replica Symmetry Breaking's (1RSB) basic assumptions (Mézard and Montanari, 2009).

From a geometric perspective, the intuitive idea behind Perturbed BP, is to perturb the messages towards a solution. However, in order to achieve this, we need to initialize the messages to a proper neighborhood of a solution. Since these neighborhoods are not initially known, we resort to stochastic perturbation of messages to make local marginals more biased towards a subspace of solutions. This continuous perturbation of all messages is performed in a way that allows each BP message to re-adjust itself to the other perturbations, more and more focusing on a random subset of solutions.

3.1 1RSB Postulate and Survey Propagation

Large random graphs are locally tree-like, which means the length of short loops are typically in the order of $\log(N)$ (Janson et al., 2001). This ensures that, in the absence of long-range correlations, BP is asymptotically exact, as the set of messages incoming to each node or factor are almost independent. Although BP messages remain uncorrelated until the condensation transition (Krzakala et al., 2007), the BP equations do not completely characterize the set of solutions after the clustering transition. This inadequacy is indicated

6. 1sRSB is short for 1st order static Replica Symmetry Breaking.
7. In some problems, the rigidity transition occurs before condensation transition.

by the existence of a set of several valid fixed points (rather than a unique fixed-point) for BP, each of which corresponds to a quasi-solution. For a better intuition, consider the cartoons of Figures 3(b) and (c). During the clustering phase (b), x_i and x_j (corresponding to the x and y axes) are not highly correlated, but they become correlated during and after condensation (c). This correlation between variables that are far apart in the PGM results in correlation between the BP messages. This violates BP's assumption that messages are uncorrelated, which results in BP's failure in this regime.

1RSB's approach to incorporating this clustering of solutions into the equilibrium conditions is to define a new Gibbs measure over clusters. Let $\mathbf{y} \subset \mathcal{S}$ denote a cluster of solutions and $\mathcal{Y}$ be the set of all such clusters. The idea is to treat $\mathcal{Y}$ the same as we treated $\mathcal{X}$, by defining a distribution

$$\boldsymbol{\mu}(\mathbf{y}) \quad \propto \quad |\mathbf{y}|^m \quad \forall\, \mathbf{y} \in \mathcal{Y} \tag{13}$$

where $m \in [0,1]$, called the Parisi parameter (Mézard et al., 1987), specifies how each cluster's weight depends on its size. This implicitly defines a distribution over $\mathcal{X}$

$$\mu(x) \quad \propto \quad \sum_{\mathbf{y} \ni x} \boldsymbol{\mu}(\mathbf{y}) \tag{14}$$

N.b., $m = 1$ corresponds to the original distribution (Equation (1)).

Example 7 *Going back to our simple 3-SAT example,* $\mathbf{y}^{(1)} = \{(True, True, True)\}$ *and* $\mathbf{y}^{(2)} = \{(False, False, False),\ (False, False, True)\}$ *are two clusters of solutions. Using* $m = 1$*, we have*
$\boldsymbol{\mu}(\{\{True, True, True\}\}) = 1/3$ *and* $\boldsymbol{\mu}(\{\{False, False, False\}, \{False, False, True\}\}) = 2/3$*. This distribution over clusters reproduces the distribution over solutions—i.e.,* $\mu(x) = 1/3\ \forall x \in \mathcal{S}$*. On the other hand, using* $m = 0$*, produces a uniform distribution over clusters, but it does not give us a uniform distribution over the solutions.*

This meta-construction for $\mu(\mathbf{y})$ can be represented using an auxiliary PGM. One may use BP to find marginals over this PGM; here BP messages are distributions over all BP messages in the original PGM, as each cluster is a fixed-point for BP. This requirement to represent a distribution over distributions makes 1RSB practically intractable. In general, each original BP message is a distribution over $\mathcal{X}_i$ and it is difficult to define a distribution over this infinite set. However this simplifies if the original BP messages can have limited values. Fortunately if we apply max-product BP to solve a CSP, instead of sum-product BP (of Equations (3) and (4)), the messages can have a finite set of values.

Max-Product BP: Our previous formulation of CSP was using sum-product BP. In general, max-product BP is used to find the Maximum a Posteriori (MAP) assignment in a PGM, which is a single assignment with the highest probability. In our PGM, the MAP assignment is a solution for the CSP. The max-product update equations are

$$\eta_{i\to I}(x_i) = \prod_{J\in\partial i\setminus I} \eta_{J\to i}(x_i) = \Lambda_{i\to I}(\ \{\eta_{J\to i}\}_{J\in\partial i\setminus I}\ )(x_i) \tag{15}$$

$$\eta_{I\to i}(x_i) = \max_{\mathcal{X}_{\partial I\setminus i}} C_I(x_I) \prod_{j\in\partial I\setminus i} \eta_{j\to I}(x_j) = \Lambda_{I\to i}(\ \{\eta_{j\to I}\}_{j\in\partial I\setminus i}\ )(x_i) \tag{16}$$

$$\widehat{\mu}(x_i) = \prod_{J\in\partial i} \eta_{J\to i}(x_i) = \Lambda_i(\ \{\eta_{J\to i}\}_{J\in\partial i}\ )(x_i) \tag{17}$$

where $\Lambda = \{\Lambda_{i\to I}, \Lambda_{I\to i}\}_{i,I\in\partial I}$ is the max-product BP operator and Λ_i represents the marginal estimate as a function of messages. Note that here messages and marginals are not distributions. We initialize $\nu_{i\to I}(x_i) \in \{0,1\},\ \forall I, i \in \partial I, x_i \in \mathcal{X}_i$. Because of the way constraints and update equations are defined, at any point during the updates we have $\nu_{i\to I}(x_i) \in \{0,1\}$. This is also true for $\widehat{\mu}(x_i)$. Here any of $\nu_{i\to I}(x_i) = 1$, $\nu_{I\to i}(x_i) = 1$ or $\widehat{\mu}(x_i) = 1$, shows that value x_i is allowed according to a message or marginal, while 0 forbids that value. Note that $\widehat{\mu}(x_i) = 0\ \forall x_i \in \mathcal{X}_i$ *iff* no solution was found, because the incoming messages were contradictory. The non-trivial fixed-points of max-product BP define quasi-solutions in 1RSB phase, and therefore define clusters $\mathbf{y}$.

Example 8 *If we initialize all messages to* 1 *for our simple 3-SAT example, the final marginals over all the variables are equal to 1, allowing all assignments for all variables. However beside this trivial fixed-point, there are other fixed points that correspond to two clusters of solutions.*

For example, considering the cluster $\{(False, False, False), (False, False, True)\}$*, the following* $\{\eta_{i\to I}\}$ *(and their corresponding* $\{\eta_{I\to i}\}$ *define a fixed-point for max-product BP:*

$$
\begin{aligned}
&\eta_{1\to I}(True) = \widehat{\mu}_1(True) = 0 && \eta_{1\to I}(False) = \widehat{\mu}_1(False) = 1 \quad \forall I \in \partial 1\\
&\eta_{2\to I}(True) = \widehat{\mu}_2(True) = 0 && \eta_{2\to I}(False) = \widehat{\mu}_2(False) = 1 \quad \forall I \in \partial 2\\
&\eta_{3\to I}(True) = \widehat{\mu}_3(True) = 1 && \eta_{3\to I}(False) = \widehat{\mu}_3(False) = 1 \quad \forall I \in \partial 3
\end{aligned}
$$

Here the messages indicate the allowed assignments within this particular cluster of solutions.

3.1.1 Survey Propagation

Here we define the 1RSB update equations over max-product BP messages. We skip the explicit construction of the auxiliary PGM that results in SP update equations, and confine this section to the intuition offered by SP messages. Braunstein et al. (2005) and Mézard and Montanari (2009) give details on the construction of the auxiliary-PGM. Ravanbakhsh and Greiner (2014) present an algebraic perspective on SP. Maneva et al. (2007) provide a different view on the relation of BP and SP for the satisfiability problem and Kroc et al. (2007) present empirical study of SP as applied to SAT.

Let $\mathcal{Y}_i = 2^{|\mathcal{X}_i|}$ be the power-set[8] of $\mathcal{X}_i$. Each max-product BP message can be seen as a subset of $\mathcal{X}_i$ that contains the allowed states. Therefore $\mathcal{Y}_i$ as its power-set contains all possible max-product BP messages. Each message $\boldsymbol{\nu}_{i\to I} : \mathcal{Y}_i \to [0,1]$ in the auxiliary PGM defines a *distribution* over original max-product BP messages.

Example 9 (3-COL) $\mathcal{X}_i = \{1,2,3\}$ *is the set of colors and* $\mathcal{Y}_i = \{\{\}, \{1\}, \{2\}, \{3\}, \{1,2\}, \{2,3\}, \{1,3\}, \{1,2,3\}\}$*. Here* $\mathbf{y}_i = \{\}$ *corresponds to the case where none of the colors are allowed.*

8. The power-set of $\mathcal{X}$ is the set of all subsets of $\mathcal{X}$, including $\{\}$ and $\mathcal{X}$ itself.

Applying sum-product BP to our auxiliary PGM gives entropic SP(m) updates as:

$$\boldsymbol{\nu}_{i\to I}(\mathbf{y}_i) \propto |\mathbf{y}_i|^m \sum_{\{\eta_{J\to i}\}_{J\in\partial i\setminus I}} \delta(\mathbf{y}_i, \Lambda_{i\to I}(\{\eta_{J\to i}\}_{J\in\partial i\setminus I})) \prod_{J\in\partial i\setminus I} \boldsymbol{\nu}_{J\to i}(\eta_{J\to i}) \tag{18}$$

$$\boldsymbol{\nu}_{I\to i}(\mathbf{y}_i) \propto |\mathbf{y}_i|^m \sum_{\{\eta_{j\to I}\}_{j\in\partial I\setminus i}} \delta(\mathbf{y}_i, \Lambda_{I\to i}(\{\eta_{j\to I}\}_{j\in\partial I\setminus i})) \prod_{j\in\partial I\setminus i} \boldsymbol{\nu}_{j\to I}(\eta_{j\to I}) \tag{19}$$

$$\boldsymbol{\nu}_{i\to I}(\{\}) \quad := \quad \boldsymbol{\nu}_{I\to i}(\{\}) \quad := \quad 0 \qquad \forall i,\ I\in\partial i \tag{20}$$

where the summations are over all combinations of max-product BP messages. Here the δ-function ensures that only the set of incoming messages that satisfy the original BP equations make contributions. Since we only care about the valid assignments and $\mathbf{y}_i = \{\}$ forbids all assignments, we ignore its contribution (Equation 20).

Example 10 (3-SAT) *Consider the SP message* $\boldsymbol{\nu}_{1\to C_1}(\mathbf{y}_1)$ *in the factor graph of Figure 1b. Here the summation in Equation* (18) *is over all possible combinations of incoming max-product BP messages* $\eta_{C_2\to 1},\ldots,\eta_{C_5\to 1}$. *Since each of these messages can assume one of the three valid values—e.g.,* $\eta_{C^2\to 1}(x_1) \in \{\ \{True\},\{False\},\{True,False\}\ \}$*—for each particular assignment of* $\mathbf{y}_1$*, a total of* $|\{\{True\},\{False\},\{True,False\}\}|^{|\partial 1\setminus C_1|} = 3^4$ *possible combinations are enumerated in the summations of Equation* (18). *However only the combinations that form a valid max-product message update have non-zero contribution in calculating* $\boldsymbol{\nu}_{1\to C_1}(\mathbf{y}_1)$. *These are basically the messages that appear in a max-product fixed point as discussed in Example 8.*

Each of original messages corresponds to a different sub-set of clusters and m (from Equation (13)) controls the effect of each cluster's size on its contribution. At any point, we can use these messages to estimate the marginals of $\widehat{\boldsymbol{\mu}}(\mathbf{y})$ defined in Equation (13) using

$$\widehat{\boldsymbol{\mu}}(\mathbf{y}_i) \ \propto\ |\mathbf{y}_i|^m \sum_{\{\eta_{J\to i}\}_{J\in\partial i}} \delta(\mathbf{y}_i,\ \Lambda_i(\ \{\eta_{J\to i}\}_{J\in\partial i})\ ) \prod_{J\in\partial i} \boldsymbol{\nu}_{J\to i}(\eta_{J\to i}). \tag{21}$$

This also implies a distribution over the original domain, which we slightly abuse notation to denote by

$$\widehat{\boldsymbol{\mu}}(x_i) \ \propto \sum_{\mathbf{y}_i \ni x_i} \widehat{\boldsymbol{\mu}}(\mathbf{y}_i). \tag{22}$$

The term SP usually refers to SP(0)—that is $m = 0$—where all clusters, regardless of their size, contribute the same amount to $\boldsymbol{\mu}(\mathbf{y})$. Now that we can obtain an estimate of marginals, we can employ this procedure within a decimation process to incrementally fix some variables. Here either $\widehat{\boldsymbol{\mu}}(x_i)$ or $\widehat{\boldsymbol{\mu}}(\mathbf{y}_i)$ can be used by the decimation procedure to fix the most biased variables. In the former case, a variable $\mathbf{y}_i$ is fixed to $\mathbf{y}_i^* = \{x_i^*\}$ when $x_i^* = \arg_{x_i}\max \widehat{\boldsymbol{\mu}}(x_i)$. In the latter case, $\mathbf{y}_i^* = \arg_{\mathbf{y}_i}\max \widehat{\boldsymbol{\mu}}(\mathbf{y}_i)$. Here we use SP-dec(S) to refer to the former procedure (that uses $\widehat{\boldsymbol{\mu}}(x_i)$ to fix variables to a *single* value) and use SP-dec(C) to refer to the later case (in which variables are fixed to a *cluster* of assignments).

The original decimation procedure for κ-SAT (Braunstein et al., 2002) corresponds to SP-dec(S). SP-dec(C) for CSP with Boolean variables is only slightly different, as SP-dec(C) can choose to fix a cluster to $\mathbf{y}_i = \{True, False\}$ in addition to the options of $\mathbf{y}_i = \{True\}$ and $\mathbf{y}_i = \{False\}$, available to SP-dec(S). However, for larger domains (*e.g.*, q-COL), SP-dec(C) has a clear advantage. For example in 3-COL, SP-dec(C) may choose to fix a cluster to $\mathbf{y}_i = \{1, 2\}$ while SP-dec(S) can only choose between $\mathbf{y}_i \in \{\{1\}, \{2\}, \{3\}\}$. This significant difference is also reflected in their comparative success-rate on q-COL.[9] (See Table 1 in Section 3.3.)

During the decimation process, usually after fixing a subset of variables, the SP marginals $\widehat{\boldsymbol{\mu}}(x_i)$ become uniform, indicating that clusters of solutions have no preference over particular assignments of the remaining variables. The same happens when we apply SP to random instances in RS phase. At this point (a.k.a. paramagnetic phase), a local search method or BP-dec can often efficiently find an assignment to the variables that are not yet fixed by decimation. Note that both SP-dec(C) and SP-dec(S) switch to local search as soon as all $\widehat{\boldsymbol{\mu}}(x_i)$ become close to uniform.

The computational complexity of each SP update of Equation (19) is $\mathcal{O}(2^{|\mathcal{X}_i|}-1)^{|\partial I|}$ as for each particular value $\mathbf{y}_i$, SP needs to consider every combination of incoming messages, each of which can take $2^{|\mathcal{X}_i|}$ values (minus the empty set). Similarly, using a naive approach the cost of update of Equation (18) is $\mathcal{O}(2^{|\mathcal{X}_i|}-1)^{|\partial i|}$. However by considering incoming messages one at a time, we can perform the same exact update in $\mathcal{O}(|\partial i|\ 2^{2|\mathcal{X}_i|})$. In comparison to the cost of BP updates, we see that SP updates are substantially more expensive for large $|\mathcal{X}_i|$ and $|\partial I|$.[10]

3.2 Perturbed Survey Propagation

The perturbation scheme that we use for SP is similar to what we did for BP. Let $\boldsymbol{\Phi}_{i\to I}(\ \{\boldsymbol{\nu}_{j\to J}\}_{j\in\Delta i,(J\in\partial i)\setminus I})\ )$ denote the update operator for the message from variable $\mathbf{y}_i$ to factor C_I. This operator is obtained by substituting Equation (19) into Equation (18) to get a single SP update equation. Let $\boldsymbol{\Phi}(\{\boldsymbol{\nu}_{i\to I}\}_{i,I\in\partial i})$ denote the aggregate SP operator, which applies $\boldsymbol{\Phi}_{i\to I}$ to update each individual message.

We perform Gibbs sampling from the "original" domain $\mathcal{X}$ using the implicit marginal of Equation (22). We denote this random operator by $\boldsymbol{\Psi} = \{\boldsymbol{\Psi}_i\}_i$, defined by

$$\boldsymbol{\nu}_{i\to I}(\mathbf{y}_i)\ =\ \boldsymbol{\Psi}_i(\ \{\boldsymbol{\nu}_{j\to J}\}_{j\in\Delta i,J\in\partial i}\ )\ \triangleq\ \delta(\mathbf{y}_i, \{\widehat{x}_i\})\quad \text{where } \widehat{x}_i \sim \widehat{\boldsymbol{\mu}}(x_i)$$

where the second argument of the δ-function is a singleton set, containing a sample from the estimate of marginal. Now, define the Perturbed SP operator as the convex combination of SP and either of the GS operator above:

$$\boldsymbol{\Gamma}(\{\boldsymbol{\nu}_{i\to I}\})\ \triangleq\ \gamma\boldsymbol{\Psi}(\{\boldsymbol{\nu}_{i\to I}\})\ +\ (1-\gamma)\boldsymbol{\Phi}(\{\boldsymbol{\nu}_{i\to I}\}).$$

Similar to perturbed BP, during iterations of Perturbed SP, γ is gradually increased from 0 to 1. If perturbed SP reaches the final iteration, the samples from the implicit

9. Previous applications of SP-dec to q-COL by Braunstein et al. (2003) used a heuristic for decimation that is similar SP-dec (C).
10. Note that our representation of distributions is over-complete—that is we are not using the fact that the distributions sum to one. However even in their more compact forms, for general CSPs, the cost of each SP update remains exponentially larger than that of BP (in $|\mathcal{X}_i|$, $|\partial I|$). However if the factors are sparse and have high order, both BP and SP allow more efficient updates.

marginals represent a satisfying assignment. The advantage of this scheme to SP-dec is that perturbed SP does not require any further local search. In fact we may apply $\mathbf{\Gamma}$ to CSP instances in the RS phase as well, where the solutions form a single giant cluster. In contrast, applying SP-dec, to these instances simply invokes the local search method.

To demonstrate this, we applied Perturbed SP(S) to benchmark CSP instances of Table 2 in which the maximum number of elements in the factor was less than 10. Here Perturbed SP(S) solved 80 instances out of 202 cases, while Perturbed BP solved 78 instances.

3.3 Experiments on random CSP

We implemented all the methods above for general factored CSP using the libdai code base (Mooij, 2010). To our knowledge this is the first general implementation of SP and SP-dec. Previous applications of SP-dec to κ-SAT and q-COL (Braunstein et al., 2003; Mulet et al., 2002; Braunstein et al., 2002) were specifically tailored to just one of those problems.

Here we report the results on κ-SAT for $\kappa \in \{3, 4\}$ and q-COL for $q \in \{3, 4, 9\}$. We used the procedure discussed in the examples of Section 3 to produce 100 random instances with $N = 5,000$ variables for each control parameter α. We report the probability of finding a satisfying assignment for different methods (*i.e.*, the portion of 100 instances that were satisfied by each method). For coloring instances, to help decimation, we break the initial symmetry of the problem by fixing a single variable to an arbitrary value.

For BP-dec and SP-dec, we use a convergence threshold of $\epsilon = .001$ and fix $\rho = 1\%$ of variables per iteration of decimation. Perturbed BP and Perturbed SP use $T = 1000$ iterations. Decimation-based methods use a maximum of $T = 1000$ iterations per iteration of decimation. If any of the methods failed to find a solution in the first attempt, T was increased by a factor of 4 at most 3 times (so in the final attempt: $T = 64,000$). To avoid blow-up in run-time, for BP-dec and SP-dec, only the maximum iteration, T, during the first iteration of decimation, was increased (this is similar to the setting of Braunstein et al. (2002) for SP-dec). For both variations of SP-dec (see Section 3.1.1), after each decimation step, if $\max_{i,x_i} \mu(x_i) - \frac{1}{|\mathcal{X}_i|} < .01$ we consider the instance para-magnetic, and run BP-dec (with $T = 1000$, $\epsilon = .001$ and $\rho = 1\%$) on the simplified instance.

Figure 4(first row) visualizes the success rate of different methods on 100 instances of 3-SAT (right) and 3-COL (left). Figure 4(second row) reports the number of variables that are fixed by SP-dec(C) and (S) before calling BP-dec as local search. The third row shows the average amount of time that is used to find a satisfying solution. This does not include the failed attempts. For SP-dec variations, this time includes the time used by local search. The final row of Figure 4 shows the number of iterations used by each method at each level of difficulty over the successful instances. Here the area of each disk is proportional to the frequency of satisfied instances with that particular number of iterations for each control parameter and inference method[11].

Here we make the following observations:

- **Perturbed BP is much more effective than BP-dec, while remaining ten to hundreds of times more efficient.**

11. The number of iterations are rounded to the closest power of two.

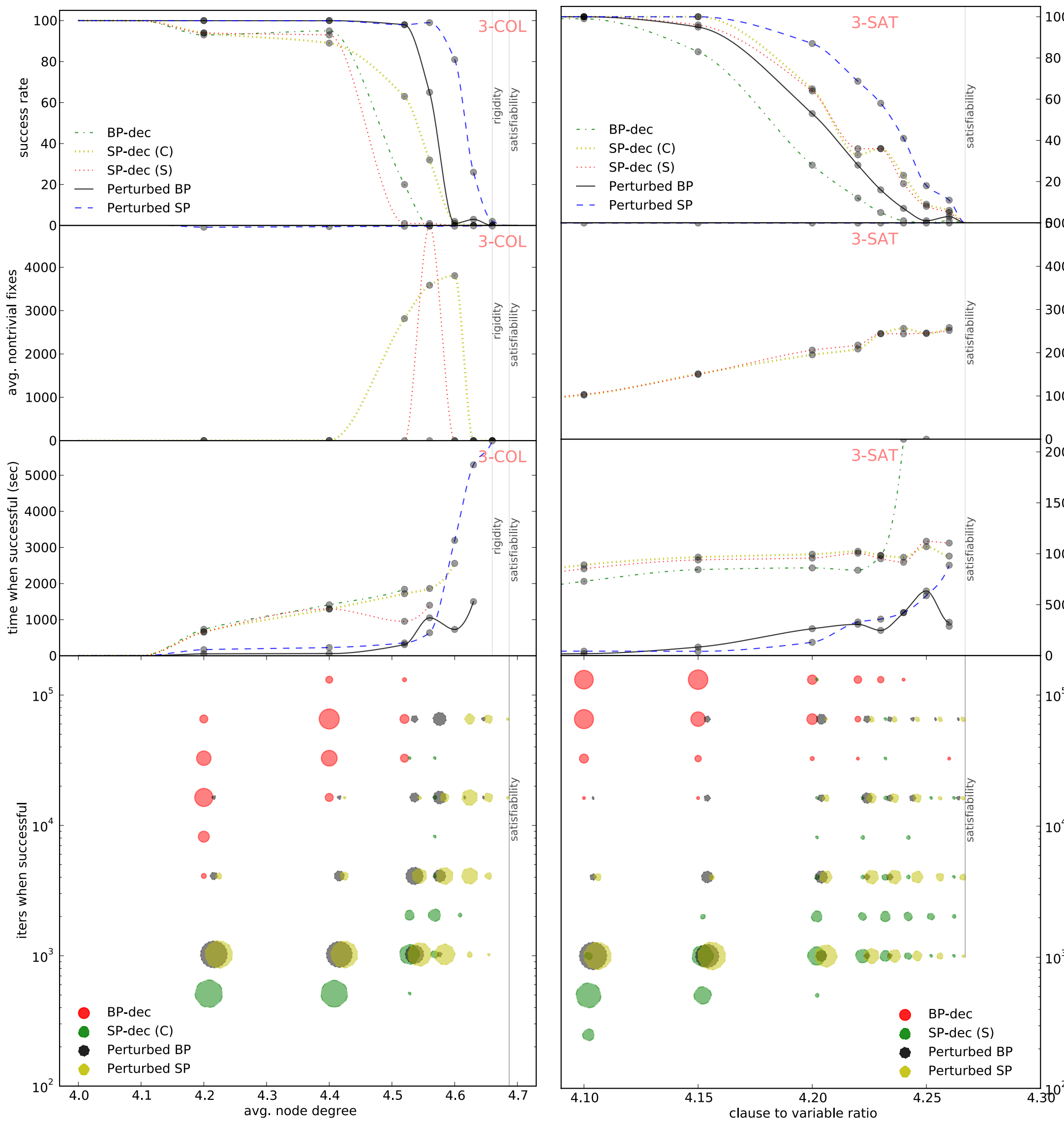


Figure 4: (first row) Success-rate of different methods for 3-COL and 3-SAT for various control parameters. (second row) The average number of variables (out of $N = 5000$) that are fixed using SP-dec (C) and (S) before calling local search, averaged over 100 instances. (third row) The average amount of time (in seconds) used by the successful setting of each method to find a satisfying solution. For SP-dec(C) and (S) this includes the time used by local search. (fourth row) The number of iterations used by different methods at different control parameters, when the method was successful at finding a solution. The number of iterations for each of 100 random instances is rounded to the closest power of 2. This does not include the iterations used by local search after SP-dec.

- As the control parameter grows larger, the chance of requiring more iterations to satisfy the instance increases for all methods.
- Although computationally very inefficient, BP-dec is able to find solutions for instances with larger control parameters than suggested by previous results (*e.g.*, Mézard and Montanari, 2009).
- For many instances where SP-dec(C) and (S) use few iterations, the variables are fixed to a trivial cluster $\mathbf{y}_i = \mathcal{X}_i$, which allows all assignments. This is particularly pronounced for 3-COL, where up to $\alpha = 4.4$ the non-trivial fixes remains zero and therefore the success rate up to this point is solely due to BP-dec.
- While for 3-SAT, SP-dec(C) and SP-dec(S) have a similar performance, for 3-COL, SP-dec(C) significantly outperforms SP-dec(S).

Table 1 reports the success-rate as well as the average of total iterations in the *successful* attempts of each method. Here the number of iterations for SP-dec(C) and (S) is the sum of iterations used by the method and the following local search. We observe that Perturbed BP can solve most of the easier instances using only $T = 1000$ iterations (*e.g.*, see Perturb BP's result for 3-SAT at $\alpha = 4.1$, 3-COL at $\alpha = 4.2$ and 9-COL at $\alpha = 33.4$).

Table 1 also supports our speculation in Section 3.1.1 that **SP-dec(C) is in general preferable to SP-dec(S)**, in particular when applied to the coloring problem.

The most important advantage of Perturbed BP over SP-dec and Perturbed SP is that Perturbed BP can be applied to instances with large factor cardinality (*e.g.*, 10-SAT) and large variable domains (*e.g.*, 9-COL). For example for 9-COL, the cardinality of each SP message is $2^9 = 512$, which makes SP-dec and Perturbed SP impractical. Here BP-dec is not even able to solve a single instance around the dynamical transition (as low as $\alpha = 33.4$) while Perturbed BP satisfies all instances up to $\alpha = 34.1$.[12] Besides the experimental results reported here, we have also used perturbed BP to efficiently solve other CSPs such as K-Packing, K-set-cover and clique-cover within the context of min-max inference (Ravanbakhsh et al., 2014).

3.4 Discussion

It is easy to check that, for $m = 1$, SP updates produce sum-product BP messages as an average case; that is, the SP updates (equations 18, 19) reduce to that of sum-product BP (equations 3, 4) where

$$\nu_{i \to I}(x_i) \quad \propto \sum_{\mathbf{y}_i \ni x_i} \boldsymbol{\nu}_{i \to I}(\mathbf{y}_i)$$

This suggests that the BP equation remains correct wherever SP(1) holds, which has lead to the belief that BP-dec should perform well up to the condensation transition (Krzakala et al., 2007). However in reaching this conclusion, the effect of decimation was ignored. More

12. Note that for 9-COL condensation transition happens after rigidity transition. So if we were able to find solutions after rigidity, it would have implied that condensation transition marks the onset of difficulty. However, this did not occur and similar to all other cases, Perturbed BP failed before rigidity transition.

Problem	ctrl param α	BP-dec		SP-dec(C)		SP-dec(S)		Perturbed BP		Perturbed SP	
		avg. iters.	success rate	avg. iters.	success rate	avg. iters.	success rate	avg. iters.	success rate	avg. iters.	success rate
3-SAT	3.86	dynamical and condensation transition									
	4.1	85405	99%	102800	100%	96475	100%	1301	100%	1211	100%
	4.15	104147	83%	118852	100%	111754	96%	5643	95%	1121	100%
	4.2	93904	28%	118288	65%	113910	64%	19227	53%	3415	87%
	4.22	100609	12%	112910	33%	114303	36%	22430	28%	8413	69%
	4.23	123318	5%	109659	36%	107783	36%	18438	16%	9173	58%
	4.24	165710	1%	126794	23%	118284	19%	29715	7%	10147	41%
	4.25	N/A	0%	123703	9%	110584	8%	64001	1%	14501	18%
	4.26	37396	1%	83231	6%	106363	5%	32001	3%	22274	11%
	4.268	satisfiability transition									
4-SAT	9.38	dynamical transition									
	9.547	condensation transition									
	9.73	134368	8%	119483	32%	120353	35%	25001	43%	11142	86%
	9.75	168633	5%	115506	15%	96391	21%	36668	27%	9783	68%
	9.78	N/A	0%	83720	9%	139412	7%	34001	12%	11876	37%
	9.88	rigidity transition									
	9.931	satisfiability transition									
3-COL	4	dynamical and condensation transition									
	4.2	24148	93%	25066	94%	24634	94%	1511	100%	1151	100%
	4.4	51590	95%	52684	89%	54587	93%	1691	100%	1421	100%
	4.52	61109	20%	68189	63%	54736	1%	7705	98%	2134	98%
	4.56	N/A	0%	63980	32%	13317	1%	28047	65%	3607	99%
	4.6	N/A	0%	74550	2%	N/A	0%	16001	1%	18075	81%
	4.63	N/A	0%	N/A	0%	N/A	0%	48001	3%	29270	26%
	4.66	rigidity transition									
	4.66	N/A	0%	N/A	0%	N/A	0%	N/A	0%	40001	2%
	4.687	satisfiability transition									
4-COL	8.353	dynamical transition									
	8.4	64207	92%	72359	88%	71214	93%	1931	100%	1331	100%
	8.46	dynamical transition									
	8.55	77618	13%	60802	13%	62876	9%	3041	100%	5577	100%
	8.7	N/A	0%	N/A	0%	N/A	0%	50287	14%	N/A	0%
	8.83	rigidity transition									
	8.901	satisfiability transition									
9-COL	33.45	dynamical transition									
	33.4	N/A	0%	N/A	N/A	N/A	N/A	1061	100%	N/A	N/A
	33.9	N/A	0%	N/A	N/A	N/A	N/A	3701	100%	N/A	N/A
	34.1	N/A	0%	N/A	N/A	N/A	N/A	12243	100%	N/A	N/A
	34.5	N/A	0%	N/A	N/A	N/A	N/A	48001	6%	N/A	N/A
	35.0	N/A	0%	N/A	N/A	N/A	N/A	N/A	0%	N/A	N/A
	39.87	rigidity transition									
	43.08	condensation transition									
	43.37	satisfiability transition									

Table 1: Comparison of different methods on $\{3,4\}$-SAT and $\{3,4,9\}$-COL. For each method the success-rate and the average number of iterations (including local search) on successful attempts are reported. The approximate location of phase transitions are given by Montanari et al. (2008); Zdeborova and Krzakala (2007).

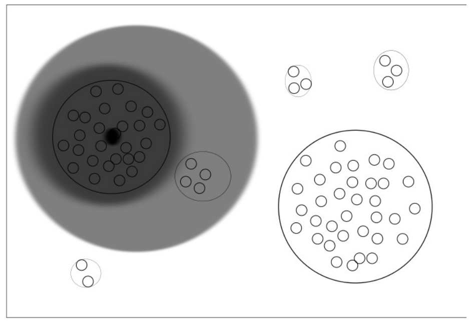


Figure 5: This schematic view demonstrates the clustering during condensation phase. Here assume horizontal and vertical axes correspond to x_1 and x_2. Considering the whole space of assignments, x_1 and x_2 are highly correlated. The formation of this correlation between distant variables on a PGM breaks BP. Now assume that Perturbed BP messages are focused on the largest shaded ellipse. In this case the correlation is significantly reduced.

recent analyses (Coja-Oghlan, 2011; Montanari et al., 2007; Ricci-Tersenghi and Semerjian, 2009) draw a similar conclusion about the effect of decimation: At some point during the decimation process, variables form long-range correlations such that fixing one variable may imply an assignment for a portion of variables that form a loop, potentially leading to contradictions. Alternatively the same long-range correlations result in BP's lack of convergence and error in marginals that may lead to unsatisfying assignments.

Perturbed BP avoids the pitfalls of BP-dec in two ways: **(I)** Since many configurations have non-zero probability until the final iteration, Perturbed BP can avoid contradictions by adapting to the most recent choices. This is in contrast to decimation in which variables are fixed once and are unable to change afterwards. A backtracking scheme suggested by Parisi (2003) attempts to fix the same problem with SP-dec. **(II)** We speculate that simultaneous bias of all messages towards sub-regions over which the BP equations remain valid, prevents the formation of long-range correlations between variables that breaks BP in 1sRSB; see Figure 5.

In all experiments, we observed that Perturbed BP is competitive with SP-dec, while BP-dec often fails on much easier problems. We saw that the cost of each SP update grows exponentially faster than the cost of each BP update. Meanwhile, our perturbation scheme adds a negligible cost to that of BP—*i.e.*, that of sampling from these local marginals and updating the outgoing messages accordingly. Considering the computational complexity of SP-dec, and also the limited setting under which it is applicable, Perturbed BP is an attractive substitute. Furthermore our experimental results also suggest that Perturbed SP(S) is a viable option for real-world CSPs with small variable domains and constraint cardinalities.

4. Conclusion

We considered the challenge of efficiently producing assignments that satisfy hard combinatorial problems, such as κ-SAT and q-COL. We focused on ways to use message passing methods to solve CSPs, and introduced a novel approach, Perturbed BP, that combines BP and GS in order to sample from the set of solutions. We demonstrated that Perturbed BP is significantly more efficient and successful than BP-dec. We also demonstrated that Perturbed BP can be as powerful as a state-of-the-art algorithm (SP-dec), in solving rCSPs while remaining tractable for problems with large variable domains and factor cardinalities. Furthermore we provided a method to apply the similar perturbation procedure to SP, producing the Perturbed SP process that outperforms SP-dec in solving difficult rCSPs.

Acknowledgments

We would like to thank anonymous reviewers for their constructive and insightful feedback. RG received support from NSERC and Alberta Innovate Center for Machine Learning (AICML). Research of SR has been supported by Alberta Innovates Technology Futures, AICML and QEII graduate scholarship. This research has been enabled by the use of computing resources provided by WestGrid and Compute/Calcul Canada.

References

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

D. Achlioptas and A. Coja-Oghlan. Algorithmic barriers from phase transitions. *arXiv:0803.2122*, March 2008.

D. Achlioptas, A. Naor, and Y. Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435(7043):759–764, June 2005. ISSN 0028-0836.

C. Andrieu, N. de Freitas, A. Doucet, and M. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 2003.

A. Braunstein, M. Mézard, and R. Zecchina. Survey propagation: an algorithm for satisfiability. *Random Structures and Algorithms*, 27(2):19, 2002.

A. Braunstein, R. Mulet, A. Pagnani, M. Weigt, and R. Zecchina. Polynomial iterative algorithms for coloring and analyzing random graphs. *Physical Review E*, 68(3):036702, 2003.

A. Braunstein, M. Mézard, M. Weigt, and R. Zecchina. Constraint satisfaction by survey propagation. *Computational Complexity and Statistical Physics*, page 107, 2005.

P. Cheeseman, B. Kanefsky, and W. Taylor. Where the really hard problems are. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume*

1, IJCAI'91, pages 331–337, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. ISBN 1-55860-160-0.

S. Cocco, O. Dubois, J. Mandler, and R. Monasson. Rigorous decimation-based construction of ground pure states for spin-glass models on random rattices. *Physical review letters*, 90(4):047205, 2003.

A. Coja-Oghlan. On belief propagation guided decimation for random k-SAT. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 957–966. SIAM, 2011.

V. Deolalikar. Deolalikar P vs NP paper, 2010. URL `http://michaelnielsen.org/polymath1/index.php?title=Deolalikar_P_vs_NP_paper`.

Y. Fu and P. Anderson. Application of statistical mechanics to NP-Complete problems in combinatorial optimisation. *Journal of Physics A: Mathematical and General*, 19(9): 1605–1620, June 1986. ISSN 0305-4470, 1361-6447.

M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, volume 44 of *A Series of Books in the Mathematical Sciences*. Freeman W., 1979. ISBN 0716710455.

A. Ihler and D. Mcallester. Particle belief propagation. In *International Conference on Artificial Intelligence and Statistics*, pages 256–263, 2009.

S. Janson, T. Luczak, and V. Kolchin. Random graphs. *Bulletin of the London Mathematical Society*, 33:363–383, 2001.

K. Kask, R. Dechter, and V. Gogate. Counting-based look-ahead schemes for constraint satisfaction. In *Principles and Practice of Constraint Programming*, pages 317–331. Springer, 2004.

H. Kautz, A. Sabharwal, and B. Selman. Incomplete algorithms. In *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 185–203. IOS Press, 2009. ISBN 978-1-58603-929-5.

S. Kirkpatrick and B. Selman. Critical behavior in the satisfiability of random boolean expressions. *Science*, 264(5163):1297–1301, 1994.

L. Kroc, A. Sabharwal, and B. Selman. Survey propagation revisited. In *Proceedings of the 23rd conference on Uncertainty in Artificial Intelligence*, pages 217–226, 2007.

L. Kroc, A. Sabharwal, and B. Selman. Message-passing and local heuristics as decimation strategies for satisfiability. In *Proceedings of the 2009 ACM Symposium on Applied Computing*, pages 1408–1414. ACM, 2009.

F. Krzakala, A. Montanari, F. Ricci-Tersenghi, G. Semerjian, and L. Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences*, 104(25):10318–10323, 2007.

F. Kschischang, B. Frey, and H. Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, 2001.

Ch. Lecoutre. A collection of CSP benchmark instances., October 2013. URL `http://www.cril.univ-artois.fr/~lecoutre/benchmarks.html`.

E. Maneva, E. Mossel, and M. Wainwright. A new look at survey propagation and its generalizations. *Journal of the ACM*, 54(4):17, 2007.

M. Mézard and A. Montanari. *Information, Physics, and Computation*. Oxford, 2009.

M. Mézard and G. Parisi. The Bethe lattice spin glass revisited. *The European Physical Journal B-Condensed Matter and Complex Systems*, 20(2):217–233, 2001.

M. Mézard and G. Parisi. The cavity method at zero temperature. *Journal of Statistical Physics*, page 111, 2003.

M. Mézard, G. Parisi, and M. Virasoro. *Spin Glass Theory and Beyond*. Singapore: World Scientific, 1987.

M. Mézard, G. Parisi, and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297(5582):812–815, August 2002. ISSN 0036-8075, 1095-9203.

D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of SAT problems. In *Association for the Advancement of Artificial Intelligence (AAAI)*, volume 92, pages 459–465, 1992.

A. Montanari, F. Ricci-tersenghi, and G. Semerjian. Solving constraint satisfaction problems through belief propagation-guided decimation. In *in Proceedings of the Allerton Conference on Communication, Control, and Computing*, 2007.

A. Montanari, F. Ricci-Tersenghi, and G. Semerjian. Clusters of solutions and replica symmetry breaking in random k-satisfiability. *Journal of Statistical Mechanics*, page 04004, 2008.

J. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010.

R. Mulet, A. Pagnani, M. Weigt, and R. Zecchina. Coloring random graphs. *Physical Review Letters*, 89(26):268701, 2002.

N. Noorshams and M. Wainwright. Belief propagation for continuous state spaces: Stochastic message-passing with quantitative guarantees. *Journal of Machine Learning Research*, 14:2799–2835, 2013.

G. Parisi. A backtracking survey propagation algorithm for k-satisfiability. *Arxiv preprint Condmat:0308510*, page 9, 2003.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, volume 88 of *Representation and Reasoning*. Morgan Kaufmann, 1988. ISBN 1558604790.

S. Ravanbakhsh and R. Greiner. Revisiting algebra and complexity of inference in graphical models. *arXiv:cs/1409.7410v4*, 2014.

S. Ravanbakhsh, C. Srinivasa, B. Frey, and R. Greiner. Min-max problems on factor-graphs. *Proceedings of The 31st International Conference on Machine Learning*, pages 1035–1043, 2014.

F. Ricci-Tersenghi. Being glassy without being hard to solve. *Science*, 330(6011):1639–1640, 2010.

F. Ricci-Tersenghi and G. Semerjian. On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms. *Journal of Statistical Mechanics*, page 9001, April 2009.

Ch. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. ISBN 0387212396.

O. Roussel and Ch. Lecoutre. XML representation of constraint networks: Format XCSP 2.1. *arXiv:0902.2362*, 2009.

B. Selman, H. Kautz, and B. Cohen. Noise strategies for improving local search. In *Association for the Advancement of Artificial Intelligence*, volume 94, pages 337–343, 1994.

L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

L. Zdeborova and F. Krzakala. Phase transitions in the coloring of random graphs. *Physical Review E*, 76(3):031131, 2007.

Appendix A. Detailed Results for Benchmark CSP

In Table 2 we report the average number of iterations and average time for the attempt that successfully found a satisfying assignment (*i.e.*, the failed instances are not included in the average). We also report the number of satisfied instances for each method as well as the number of satisfiable instance in that series of problems (if known). Further information about each data-set maybe obtained from Lecoutre (2013).

				BP-dec			Perturbed BP		
problem	series	instances	# satisfiable	# satisfied	avg. time (s)	avg. iters	# satisfied	avg. time (s)	avg. iters
Geometric	-	100	92	77	208.63	30383	81	.70	74
Dimacs	aim-50	24	16	9	11.41	25344	14	.07	181
	aim-100	24	16	8	18.2	16755	11	.15	213
	aim-200	24	N/A	7	401.90	160884	6	.17	46
	ssa	8	N/A	4	.60	373.25	4	.50	86
	jhnSat	16	16	16	5839.86	141852	13	9.82	117
	varDimacs	9	N/A	4	2.95	715	4	.12	18
QCP	QCP-10	15	10	10	43.87	30054	10	.22	51
	QCP-15	15	10	3	5659.70	600741	4	9.59	530
	QCP-25	15	10	0	0	0	0	0	0
Graph-Coloring	ColoringExt	17	N/A	4	.05	103	5	.04	25
	school	8	N/A	0	N/A	N/A	5	62.86	153
	myciel	16	N/A	5	.21	59	5	.05	11
	hos	13	N/A	5	27.34	606	5	10.04	37
	mug	8	N/A	4	.068	313	4	.004	11
	register-fpsol	25	N/A	0	N/A	N/A	0	N/A	N/A
	register-inithx	25	N/A	0	N/A	N/A	0	N/A	N/A
	register-zeroin	14	N/A	3	5906.16	26544	0	N/A	N/A
	register-mulsol	49	N/A	5	59.27	418	0	N/A	N/A
	sgb-queen	50	N/A	7	35.66	916	11	7.56	81
	sgb-games	4	N/A	1	.91	434	1	.07	21
	sgb-miles	34	N/A	4	20.86	371	2	4.20	181
	sgb-book	26	N/A	5	1.72	444	5	.18	39
	leighton-5	8	N/A	0	N/A	N/A	0	N/A	N/A
	leighton-15	28	N/A	0	N/A	N/A	1	106.46	641
	leighton-25	29	N/A	2	304.49	1516	2	94.11	241
All Interval Series	series	12	12	2	4.78	11319	7	1.85	520
Job Shop	e0ddr1	10	10	9	707.74	9195	5	37	257
	e0ddr2	10	10	5	3640.40	26544	7	74.49	366
	ewddr2	10	10	10	10871.96	48053	9	21.24	72
Schurr's Lemma	-	10	N/A	1	39.89	120152	2	.97	100
Ramsey	Ramsey 3	8	N/A	1	.01	61	4	.75	283
	Ramsey 4	8	N/A	2	12941.51	561300	7	7.39	81
Chessboard Coloration	-	14	N/A	5	35.51	3111	5	.66	27
Hanoi	-	3	3	3	.48	12	3	.52	14
Golomb Ruler	Arity 3	8	N/A	2	1.39	103	2	19.78	660
Queens	queens	8	8	7	3.30	159	8	2.43	57
Multi-Knapsack	mknap	2	2	2	2.44	6	2	4.41	10
Driver	-	7	7	5	10.14	1438	5	4.74	274
Composed	25-10-20	10	10	8	1.62	695	5	.17	38
Langford	lagford-ext	4	2	0	N/A	N/A	1	.002	10
	lagford 2	22	N/A	4	.67	127	10	11.64	10
	lagford 3	20	N/A	0	N/A	N/A	N/A	N/A	N/A

Table 2: Comparison of Perturbed BP and BP-guided decimation on benchmark CSPs.