**Statistical Physics for Communication and Computer Science** 14/04/2011

# Lecture Notes 8: BP-Guided Decimation for SAT Problems

*Lecturer: Rudiger Urbanke* *Scribe:Amir Hesam Salavati*

## Introduction

In this lecture, we will describe an algorithm based on belief propagation to check the satisfiablity of SAT problems. The algorithm is called *BP-Guided Decimation for SAT Problems*. we first introduce an accurate version of the algorithm to be used over trees and then extend it to the general factor graphs. In the second part of the lecture, we will re-parametrize the message passing equations in a more convenient way to help us perform belief propagation for SAT problems.

## 1 BP-Guided Decimation for SAT Problems

Let's start by a very simple example. Suppose we are given the following formula:

$$F = x_1 \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3) \tag{1}$$

The factor graph for this equation is illustrated in figure 1, in which dashed lines means the variable appears negated in the corresponding clause.

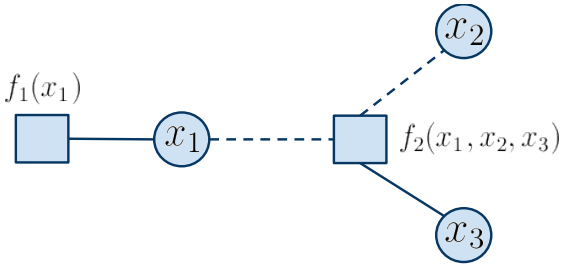


Figure 1: Factor graph of the equation $F = x_1 \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3)$

$F$ is a Boolean function. However, we can slightly modify $F$ and model it as a binary function that can take either of 0 or 1 values. In this case, we can write $F$ as the product of two other binary functions: $f_1 = x_1$ and $f_2 = sign(\overline{x_1} + \overline{x_2} + x_3)$, where $sign$ is the normal sign function with $sign(0) = 0$.

In order to see if $F$ is satisfiable, we can compute the "partition function"

$$\sum_{x_1,x_2,x_3} f_1(x_1) f_2(x_1, x_2, x_3)$$

| $x_1$ | $x_2$ | $x_3$ | $f_1(x_1)$ | $f_1(x_1,x_2,x_3)$ | F |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |

Table 1: Satisfiability of $F$, given by equation (1), for all possible combination of $x_1$, $x_2$ and $x_3$.

This will in fact give us the number of satisfying configurations. Here there are 3 SAT solutions. Table 1 illustrate the satisfiability of $F$ for all possible combination of $x_1$, $x_2$ and $x_3$.

We can also look at marginals with respect to different variables, for instance

$$\sum_{\sim x_1} f_1(x_1) f_2(x_1, x_2, x_3)$$

which will be the number of satisfying clauses given that $x_1$ has a particular fixed value. From table 1 we see that $\mu(x_1 = 0) = 0$ and $\mu(x_1 = 1) = 3$; $\mu(x_2 = 0) = 2$ and $\mu(x_2 = 1) = 1$; $\mu(x_3 = 0) = 1$ and $\mu(x_3 = 1) = 2$.

Since the graph of this simple formula is a tree, the partition function and the whole set of marginals can be exactly calculated using belief propagation. Table 2 summarizes the messages exchanged in each iteration of the belief propagation in order to compute the marginal with respect to $x_1$, denoted by $\mu(x_1)$, for the factor graph given in figure 1. Let us illustrate the use of message passing rules for the derivation of $\mu(x_1)$. The first line of the table gives the initial messages at the leaf nodes.First we compute

$$\mu_{2\to 1} = \sum_{\sim x_1} f_2(x_1, x_2, x_3). \underbrace{\mu_{2\to 2}(x_2)}_{1} \underbrace{\mu_{3\to 2}(x_3)}_{1} = 4 \text{ if } x_1 = 0 \text{ and } 3 \text{ if } x_1 = 1 \tag{2}$$

and

$$\mu_{1\to 1} = f(x_1) = 0 \text{ if } x_1 = 0 \text{ and } 1 \text{ if } x_1 = 1 \tag{3}$$

Finaly,

$$\mu(x_1) = \mu_{2\to 1}(x_1)\mu_{1\to 1}(x_1) = 0 \text{ if } x_1 = 0 \text{ and } 3 \text{ if } x_1 = 1 \tag{4}$$

which agrees with table 1.

| Iteration number | Messages |
|---|---|
| 1 | $\mu_{f_1 \to x_1} = f_1,\ \mu_{x_2 \to f_2} = 1,\ \mu_{x_2 \to f_2} = 1$ |
| 2 | $\mu_{f_2 \to x_1} = \sum_{x_2, x_3} f_2(x_1, x_2, x_3)$ |

Table 2: Messages exchanged in each iteration of the belief propagation performed over the factor graph given in figure 1.

So far, the marginals count the number of satisfying solutions. However, if we are interested in the *fraction* of satisfying solutions with $x_i = 0$ against those with $x_i = 1$, for some $x_i$, then we have to slightly modify the message passing rules by normalizing the messages, as explained in the sequel.

**Notation**: We denote clauses by $a, b, c, \ldots$ and variables by $i, j, k, \ldots$. Furthermore, we denote the neighborhood of a node $x$ by $\partial x$. The same neighborhood excluding a particular node $y$ is indicated by $\partial x \setminus y$.

Having these notations in mind, we start modifying the message passing rules. In the original message passing scheme, the message from variable $i$ to clause $a$ is given by equation (5).

$$\mu_{i \to a}(x_i) = \prod_{b \in \partial i \setminus a} \mu_{b \to i}(x_i) \tag{5}$$

However, since we are interested in the *fraction* of the solutions with $x_i = 0$ and $x_i = 1$, we require the new messages $\widetilde{\mu}_{i \to a}(x_i)$ to satisfy the following equation.

$$\widetilde{\mu}_{i \to a}(x_i = 0) + \widetilde{\mu}_{i \to a}(x_i = 1) = 1$$

Therefore, it is sufficient to set $\widetilde{\mu}_{i \to a}(x_i)$ according to equation (6).

$$\widetilde{\mu}_{i \to a}(x_i) = \frac{\mu_{i \to a}(x_i)}{\mu_{i \to a}(x_i = 0) + \mu_{i \to a}(x_i = 1)} \tag{6}$$

At this point, it seems as if we have to once calculate $\mu_{i \to a}(x_i)$ for $x_i = 0, 1$ and then normalize the messages. However, it is easy to show that we can directly calculate $\widetilde{\mu}_{i \to a}(x_i)$. To simplify the notations, we omit the normalization factor and write the messages as

$$\widetilde{\mu}_{i \to a}(x_i) \propto \prod_{b \in \partial i \setminus a} \widetilde{\mu}_{b \to i}(x_i) \tag{7}$$

Calculating marginals will only give us the number of satisfying combinations to a SAT problem. Finding an actual solution is another story which we address in the next section.

## 2 From Counting the Number of Solutions to Finding a Solution

Given a SAT problem $F$, assume that the factor graph of $F$ is a tree and $F$ has a satisfying solution. Then algorithm 1 will find a solution that satisfies $F$.

**Algorithm 1** BP Guided Decimation over trees

1. Run belief propagation on $F$ and compute the all marginals $\mu(x_i)$ for all of the variables.
2. Pick a variable $i$. If $\mu(x_i = 0) > 0$ (there exists an assignement with $x_i = 0$), then:
   (a) Set $x_i = 0$ in all clauses.
   (b) Eliminate all those clauses that $x_i$ appears negated in them.
   (c) Remove $x_i$ from the other clause.

   If on the other hand $\mu(x_i = 0) = 0$ (there doesnt exist an assignmenet with $x_i = 0$), then:
   (a) Set $x_i = 1$ in all clauses.
   (b) Eliminate all those clauses that $x_i$ appears unnegated in them.
   (c) Remove $x_i$ from the other clause.
3. Repeat the process until no variables are left.

Note that in each step of the above algorithm we must run belief propagation.

**Terminology:** Since we use belief propagation and eliminate a variable in each iteration, the algorithm is called **BP guided decimation**.

Algorithm 1 is only accurate if we have a tree. But what about the more general cases? We will introduce a modified version of the above algorithm in the next section to deal with general factor graphs.

### 2.1 Applying BP Guided Decimation to General Factor Graphs

In this section, we apply a modified version of the BP guided decimation algorithm to general factor graphs. However, note that the graph in this section should be sparse as before.

Over a tree, the previous algorithm yields exact marginals and we can pick anyone of them in each iteration. However, in general graphs it is not the case any more. As a result and in order to deal with the inherent uncertainty in marginals, in each iteration we pick a node $i$ such that the difference $|\mu(x_i = 0) - \mu(x_i = 1)|$ is **maximized**. This way, we hope that this node has such a clear bias that its marginals are are quite exact despite the graph not being a tree.

The rest of the algorithm is the same, summarized below:

Some remarks about running BP on general graphs are in order:

- *Initialization* The typical way of initializing messages is to set all of them equal to 1/2.

**Algorithm 2** BP Guided Decimation over General Graphs

1. Run BP and calculate all marginals.
2. Pick a node $i$ such that $|\mu(x_i = 0) - \mu(x_i = 1)|$ is maximized.
3. Set $x_i$ to the most likely value, i.e. $x_i = 0$ if $\mu(x_i = 0) > \mu(x_i = 1)$ and to 1 otherwise.
4. Eliminate all clauses that the particular choice of $x_i$ make them satisfied. Remove $x_i$ from the other clause.
5. Recurse until all variables are eliminated.

- *Scheduling* In contrast to BP guided decimation over a tree, the choice of node $i$ affect the solution and the whole algorithm. Therefore, scheduling matters. We usually use flooding as a means of scheduling. In other words, in each iteration every node sends its messages over its outgoing links.

Figure 2 illustrates two kinds of probabilities as a function of $\alpha$ (ratio of nb of clauses to variables). One can run pure BP over many instances and compute the empirical probability that it converges. This yields the upper curves in figure 2. For $K = 3$ we get a convergence threshold $\alpha_{\text{BP}\approx 3.86}$ and for $K = 4$ we get $\alpha_{\text{BP}} \approx 10.3$. Now, one can run BP guided decimation (algorithm 2) over many instances and derive the empirical probability of success. The corresponding threshold must in general be lower than $\alpha_{BP}$ since BP must at least converge after each decimation step. This empirivcal probability is given by the lower curve in figure 2. For $K = 3$ the threshold is approximately identical to $\alpha_{BP}$ but for $K = 4$ it is smaller and approximately equal to 9.3.

The actual SAT-UNSAT threshold is for $K = 3$, $\alpha_{\text{sat}-\text{unsat}} \approx 4.26$ and for $K = 4$, $\alpha_{\text{sat}-\text{unsat}} \approx 9.93$. We will see in future lectures how to obtain these thresholds by *survey propagation* algorithms.

# 3 Convenient Re-parametrization

Let $J_{ia}$ denote the weight for each edge of the factor graph where $J_{ia} = 0$ means $x_i$ appears in clause $a$ unnegated and $J_{ia} = 1$ for $x_i$ appearing negated. Furthermore, let $S_{ia}$ be the subset of the neighbors of $a$ that have the same edge type (weight) attached to $a$ as that of $x_i$. Likewise, let $U_{ia}$ be the subset of the neighbors of $a$ with a different edge type as that of $J_{ia}$.

Now let $\xi_{ia}$ denote $\mu_{i\to a}(x_i = J_{ia})$ and $\hat{\xi}_{ai}$ indicate $\hat{\mu}_{a\to i}(x_i = J_{ia})$. The interpretation of this notation is that $\xi_{ia}$ is the probability that $x_i$ has a value which *does not satisfy* the clause corresponding to node $a$. Similarly, $\hat{\xi}_{ai}$ represents the probability that $x_i$ is *not free* to be chosen arbitrarily since the clause $a$ is not satisfied yet. With these notations in mind, we rewrite the message passing equations.

## 3.1 Rewriting Message Passing Equations

The original message passing equations for messages from variable to check nodes is given by:

$$\xi_{ia} \quad = \quad \mu_{i\to a}(x_i = J_{ia}) \propto \prod_{b\in\partial i\setminus a} \hat{\mu}_{b\to i}(x_i = J_{ia})$$

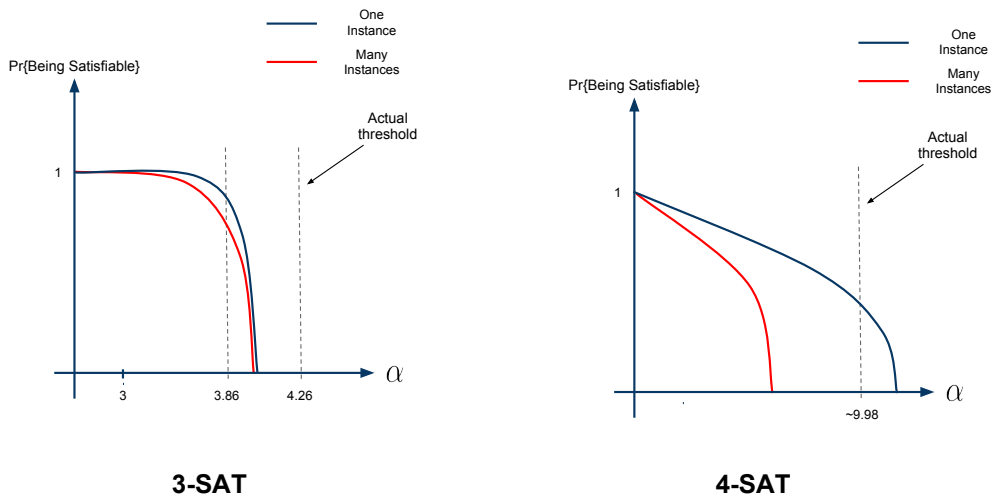


Figure 2: Probability of $3-SAT$ and $4-SAT$ being satisfied by BP guided decimation.

$$\begin{aligned} &\propto \left(\prod_{b\in S_{ia}} \hat{\mu}_{b\to i}(x_i = J_{ib})\right)\left(\prod_{b\in U_{ia}} \hat{\mu}_{b\to i}(x_i = 1 - J_{ib})\right) \\ &\propto \left(\prod_{b\in S_{ia}} \hat{\xi}_{bi}\right)\left(\prod_{b\in U_{ia}} (1-\hat{\xi}_{bi})\right) \end{aligned} \tag{8}$$

Hence, from equation (8) one can rewrite the message passing rule from variable to check nodes as follows.

$$\xi_{ia} = \frac{\left(\prod_{b\in S_{ia}} \hat{\xi}_{bi}\right)\left(\prod_{b\in U_{ia}} (1-\hat{\xi}_{bi})\right)}{\left(\prod_{b\in S_{ia}} \hat{\xi}_{bi}\right)\left(\prod_{b\in U_{ia}} (1-\hat{\xi}_{bi})\right) + \left(\prod_{b\in S_{ia}} (1-\hat{\xi}_{bi})\right)\left(\prod_{b\in U_{ia}} \hat{\xi}_{bi}\right)} \tag{9}$$

In order to rewrite the messages from check to variable nodes, let $\psi(X_{\partial a})$ be the kernel of the node $a$, i.e. it identifies the clause that node $a$ corresponds to. Then, the original message passing rules for messages from constraint to variable nodes yield

$$\hat{\xi}_{ai} = \hat{\mu}_{a\to i}(x_i = J_{ia}) = \sum_{\sim x_i} \psi(X_{\partial a}) \prod_{j\in \partial a\setminus i} \mu_{j\to i}(x_j) \tag{10}$$

If we consider $1-\hat{\xi}_{ai}$, it is obvious that $1-\hat{\xi}_{ai} = \hat{\mu}_{a\to i}(x_i = 1-J_{ia})$. Now since $x_i = 1-J_{ia}$, the clause $a$ is always satisfied irrespective of other variables, i.e. $\psi(X_{\partial a}) = 1$. As a result, equation (10) will be the sum of some products which factorizes into

$$1-\hat{\xi}_{ai} = \hat{\mu}_{a\to i}(x_i = 1-J_{ia}) \propto \prod_{j\in\partial a\setminus i} \sum_{x_j} \mu_{j\to i}(x_j) \propto 1 \tag{11}$$

The last relationship holds because the summation $\sum_{x_j} \mu_{j\to i}(x_j)$ is always equal to 1 due to the normalization of the messages. Thus, we have $1 - \hat{\xi}_{ai} \propto 1$.

Now in order to calculate $\hat{\xi}_{ai}$ exactly, i.e. to get rid of the proportional notation, let's take look at $\psi(X_{\partial a})$. We know that $x_i = J_{ia}$ since we are considering $\hat{\xi}_{ai}$. Thus, $x_i$ does not affect the value of $\psi(X_{\partial a})$. Moreover, among the other possible combinations for other variables, there will be only one combination that results in $\psi(X_{\partial a}) = 0$ (because $\psi(X_{\partial a})$ is the $OR$ product of some literals).

Hence, in order to compute $\hat{\xi}_{ai}$ we can assume that $\psi(X_{\partial a})$ is always equal to 1 and subtract the case for $\psi(X_{\partial a}) = 0$ later. As a result, and from equation (8), we will obtain

$$\begin{aligned} \hat{\xi}_{ai} &\propto \sum_{\sim x_i} \prod_{j\in\partial a\setminus i} \mu_{j\to i}(x_j) - Pr\{\psi(X_{\partial a}) = 0\} \\ &= 1 - \prod_{j\in\partial a\setminus i} \xi_{ja}(x_j) \end{aligned} \tag{12}$$

Because the first term factorizes into the product of summations and as we discussed earlier, it will always be equal to 1 due to normalization of messages.

Combining equation (9) and (12), we get the following equation for $\hat{\xi}_{ai}$.

$$\hat{\xi}_{ai} = \frac{1 - \prod_{j\in\partial a\setminus i} \xi_{ja}(x_j)}{2 - \prod_{j\in\partial a\setminus i} \xi_{ja}(x_j)} \tag{13}$$

In summary, equations (9) and (13) give the new message passing rules.