

Deriving the EM-Based Update Rules in VARSAT

Eric I. Hsu

Department of Computer Science
University of Toronto
{eihsu}@cs.toronto.edu,
<http://www.cs.toronto.edu>

Abstract. Here we show how to derive the EM-Based update rules (EMBP-L, EMBP-G, EMSP-L, EMSP-G) that are used in [1] as bias estimators for SAT. The main idea of the derivation can be viewed as recharacterizing the bias estimation task as a maximum-likelihood parameter estimation task that can be solved by variational methods like EM (Expectation Maximization). The basic framework can actually be applied to arbitrary discrete constraint satisfaction problems, such as the quasigroup-completion problem considered in [2]. For a more in-depth explanation of the difference between BP (Belief Propagation-for SAT) and SP (Survey Propagation), please consult [3].

1 Preliminaries

We will derive four update rules, from the space spanned by choosing either the two-valued model of BP, or the three-valued model of SP, and then applying Expectation Maximization (EM) with either a local or global consistency approximation. Thus the four methods are labeled EMBP-L, EMBP-G, EMSP-L, and EMSP-G.

Both BP and SP attempt to label a variable with ‘+’ or ‘-’. Importantly, this indicates that the variable is positively or negatively *constrained*, as opposed to merely appearing positively or negatively in some satisfying assignment. That is, when we examine a satisfying assignment there must exist some clause that is entirely dependent on the variable being positive in order for us to label it ‘+’. Under BP all variables are assumed to be constrained, while SP introduces the additional ‘*’ case indicating that a variable is unconstrained, i.e. all clauses are already satisfied by the other variables. The derivation will be written out in terms of SP; in the end it will be simple to handle BP just by assigning zero weight to the third ‘*’ state of SP.

The algebra of how clauses are supported comes into play when we consider the local and global consistency approximations. The local consistency considered below corresponds to generalized arc-consistency: according to a specific clause c , variable v can hold a value like ‘+’ only if the other variables of c are consistent with v being positive. Note that this cannot capture whether v is truly constrained to be positive; each individual clause might report that they are fine with v being labeled ‘+’, but between them there is no way to determine whether v is *constrained* to be positive. When we move to the more global form of consistency in the following proof, this shortcoming is remedied at potentially greater computational cost.

2 The Basic EM Framework

We first present the general EM framework [4], from a perspective that motivates its application to traditional message-passing tasks, and ultimately, to bias estimation. The result is an improved way to calculate surveys, one that always converges. At a high level, EM accepts a vector of observations Y , and seeks some model parameters θ that maximize the log-likelihood of having seen Y . This likelihood consists of a posterior probability represented by some model designed for the domain at hand; we seek the best setting of θ for fitting the model to the observations. Maximizing $\log P(Y|\theta)$ would ordinarily be straightforward, but for the additional complication that we posit some latent variables Z that contributed to the generation of Y , but that we did not get to observe. That is, we want to set θ to maximize $\log P(Y, Z|\theta)$, but cannot marginalize on Z .

So, we bootstrap by constructing an artificial probability distribution $Q(Z)$ to estimate $P(Z|Y, \theta)$ and then use this distribution to maximize the expected log-likelihood $\log P(Y, Z|\theta)$ with respect to θ . In a sense we simulate the marginalization over Z by using $Q(Z)$ as a surrogate weight: $E_Q[\log P(Z, Y|\theta)] = \sum_Z Q(Z) \log P(Z, Y|\theta)$. The first step of hypothesizing a $Q(\cdot)$ distribution is called the E-Step, and the second phase of setting θ is the M-Step. The two are repeated until convergence to a local maximum in likelihood, which is guaranteed.

3 SAT Formulation for EM

In this section we introduce the appropriate notation for the SAT problem, and then show how to formulate the bias estimation task within the Expectation Maximization framework.

Definition 1 (SAT instance) *A (CNF) SAT instance is a set C of m clauses, constraining a set V of n Boolean variables. Each clause $c \in C$ is a disjunction of literals built from the variables in V . An assignment $X \in \{0, 1\}^n$ to the variables satisfies the instance if it makes at least one literal true in each clause. The sets V_c^+ and V_c^- comprise the variables appearing positively and negatively in a clause c , respectively. The sets C_v^+ and C_v^- comprise the clauses that contain positive and negative literals for variable v , respectively. $C_v = C_v^+ \cup C_v^-$ comprises all clauses that contain v as a whole.*

Such notation is specifically geared toward the ‘‘Factor Graph’’ representation of a SAT problem [3]. This model can also be useful in understanding the derivation that follows.

Definition 2 (Bias, Profile) *For a satisfiable SAT instance \mathcal{F} , the **bias distribution** ϕ_v of a variable v represents the fraction of solutions to \mathcal{F} wherein v appears positively or negatively. Thus it consists of a **positive bias** ϕ_v^+ and a **negative bias** ϕ_v^- , where $\phi_v^+, \phi_v^- \in [0, 1]$ and $\phi_v^+ + \phi_v^- = 1$. A vector of bias distributions, one for each variable in a theory, will be called a **bias profile**, denoted $\Phi(\mathcal{F})$.*

Equivalently, it can be useful to think of a variable’s bias as the probability of finding the variable set positively or negatively when choosing a solution uniformly at random from the space of satisfying assignments.

Definition 3 (Survey) *Given a SAT instance \mathcal{F} , a bias estimation algorithm outputs a survey $\Theta(\mathcal{F})$ trying to match the true bias profile $\Phi(\mathcal{F})$. Accordingly, Θ will contain an estimated bias distribution θ_v , representing θ_v^+ and θ_v^- , for each variable v in \mathcal{F} .*

To harness the EM framework for estimating bias in SAT, the trick is to tell EM that we have seen that all the clauses are satisfied, but not how exactly they went about choosing satisfying variables for support. We ask the algorithm to find the variable biases that will best support our claimed sighting, via some hypothesized support configuration. This produces the desired $P(\Theta|SAT)$.

Vector	Status	Interpretation	Domain
Y	Observed	whether clauses are SAT	$\{0, 1\}^m$
Z	Unobserved	support configurations for clauses	$\{z_c\}^m$ $c \in C$, $z_c \in \{‘+’, ‘-’, ‘*’\}^k$
Θ	Parameters	variable biases	$\{(\theta_v^+, \theta_v^-, \theta_v^*)\}^n$ $v \in V$, $\theta_v^+ + \theta_v^- + \theta_v^* = 1$

Table 1. SAT Formulation for EM

First we set the EM variables as in Table 1. Y will be a vector of all 1’s, meaning that all clauses are satisfied. (In fact we will not really need to refer to this variable directly when performing our derivation.) Each triple $(\theta_v^+, \theta_v^-, \theta_v^*) \in \Theta$ represents variable v ’s probability of being positively constrained, negatively constrained, or unconstrained, respectively. In other words, the parameters of our model are exactly the bias distributions that comprise the survey that we would ultimately like to compute. Finally, Z contains some support configuration z_c for each clause c , denoting that the tuple of assignments to the clause’s k variables that results in satisfaction. Thus, each z_c is itself a vector, indexed by the variables that appear in clause c . In the language of duality, then, z_c represents a satisfying *extension* to c . The overall process derived below can be seen as a primal/dual optimization iterating over the original and extensional forms of the SAT problem. In constructing $Q()$ we weight the various values z_c that the clauses can hold, where the values range over tuples corresponding to the clauses’ variables. In setting Θ we probabilistically label variables according to the weights of the clause extensions.

4 Deriving Bias Estimation Heuristics using EM

For clarity, we will actually begin the derivation with the (M)aximization step, rather than the customary (E)xpectation step. So, we presume that we have already constructed an artificial distribution $Q(Z)$ that gives us a joint probability for every extension to every clause. Naturally, the ability to enumerate the joint of all satisfying extensions

presupposes finding all solutions to the problem. So, in the derivation below, this $Q(Z)$ will ultimately factorize into individual probability distributions over specific z_c 's. (This is analogous to the ‘‘Mean-Field’’ approximation to a Markov Random Field, one of the most basic members of the variational family of inference techniques [5].)

The main idea, then, is that we will assume that a $Q(Z)$ distribution has already been built using the old (initially, randomly generated) bias distributions in Θ . We use this $Q(Z)$ to update Θ , setting the bias distributions to best support the log-likelihood of having seen all the clauses satisfied according to the relative likelihood of their various support configurations. The result of this M-Step is a set of update rules for Θ that are all expressed in term of $Q(Z)$. When we proceed to the E-Step, we will choose between local and global ways to calculate $Q(Z)$ in terms of the old biases. By substituting these expressions for $Q(Z)$, we produce update rules that tell us how to set the new biases in terms of old biases. So, in practice the end result performs both the E- and the M-step at once.

5 M-Step

Our basic goal is to find variable biases Θ that maximize the expected log-likelihood of satisfaction (Y) by some hidden set of clause configurations (Z). The expectation is with respect to an artificial distribution over clause configurations ($Q(Z)$) that we will construct in the subsequent E-Step.

$$\Theta' = \operatorname{argmax}_{\Theta} \mathcal{L}(\Theta), \quad \text{where } \mathcal{L}(\Theta) = \mathcal{F}(\Theta) - \mathcal{P}(\Theta) \quad (1)$$

We perform the optimization by expressing it as a Lagrangian function comprised of the log-likelihood of interest and the constraint that Θ must be a proper probability distribution. This constraint is represented by the simpler of the two terms, $\mathcal{P}(\Theta)$:

$$\mathcal{P}(\Theta) = \sum_v \lambda_v \cdot (1 - \theta_v^+ - \theta_v^- - \theta_v^*) \quad (2)$$

A series of Lagrange multipliers λ_v , one for each variable v , allows unlimited penalty for any bias distribution that does not marginalize to exactly one. The main subject of the derivation is the $\mathcal{F}(\Theta)$ representation of log-likelihood:

$$\begin{aligned}
 \mathcal{F}(\theta) &= E_Q[\log P(Z, Y|\theta)] \\
 &= \sum_Z Q(Z) \log P(Z, Y|\theta) \\
 &= \sum_Z Q(Z) \log \prod_{c \in C} p(z_c|\theta) \\
 &= \sum_Z Q(Z) \log \prod_{c \in C} \prod_{v \in V_c} \theta_v^{z_c[v]} \tag{3} \\
 &= \sum_Z Q(Z) \sum_{c \in C} \sum_{v \in V_c} \log \theta_v^{z_c[v]} \\
 &= \sum_{c \in C} \sum_Z Q(Z) \sum_{v \in V_c} \log \theta_v^{z_c[v]}
 \end{aligned}$$

Recall that we wish to maximize $\log P(Z, Y|\theta)$, but can't marginalize on Z . So, we have to do so under expectation according to $Q(\cdot)$. This expectation comprises a weighted sum of logs; the first transformation is to decompose $P(Z, Y|\theta)$ into probabilities for individual clauses. That is, we factorize the joint probability of a total configuration Z into the marginal probabilities of its constituent z_c 's. A second point is that Y serves only to indicate satisfiability. By ensuring that we only consider values of Z that satisfy the problem at hand, we can drop any reference to Y from subsequent discussion.

The next step is to represent each clause configuration probability $p(z_c|\theta)$ in terms of the existing biases of the variables that appear in the clause. For instance, if Clause 1 in a SAT problem equals $(x \vee \neg y \vee z)$, then one possible value of z_1 is $['+', '*', '-']$; as long as x is constrained (by *some* clause) to be positive, it doesn't matter if y is allowed to vary and z is already constrained to be negative. The fourth equation above indicates that we calculate the probability of this configuration as the product of θ_x^+ , θ_y^* , and θ_z^- .

Finally, the last two lines of the initial step change products of logarithms into logarithms of products, and rearrange summations. The next step is to optimize this convex function by taking its derivative:

$$\begin{aligned}
 \frac{d\mathcal{F}}{d\theta_v^+} &= \sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \cdot \frac{1}{\theta_v^+} \\
 \frac{d\mathcal{P}}{d\theta_v^+} &= -\lambda_v \tag{4}
 \end{aligned}$$

We differentiate $\mathcal{L}(\theta)$'s two terms with respect to an individual variable v 's positive bias. (Analogous expressions apply for v 's negative and joker biases.) The more interesting derivative is \mathcal{F}' 's: it iterates through all the clauses that contain v and adds a

term for only those total configurations Z which instantiate the current clause using the ‘+’ value for v . We now proceed to set the full derivative to zero:

$$\begin{aligned} \frac{d\mathcal{L}}{d\theta_v^+} = 0 &\Rightarrow \sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \cdot \frac{1}{\theta_v^+} - \lambda_v = 0 \\ &\Rightarrow \theta_v^+ = \left(\sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \right) / \lambda_v \end{aligned} \quad (5)$$

By algebraic re-arrangement, we produce a formula specifying the optimal value for θ_v^+ for maximizing the log-likelihood expected by $Q(Z)$. We will abbreviate the numerator of this expression by ω_v^+ :

$$\omega_v^+ \triangleq \left(\sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \right) \Rightarrow \theta_v^+ = \omega_v^+ / \lambda_v \quad (6)$$

Shortly we will see that ω_v^+ serves as a “weight” on v ’s positive bias. If we define analogous weights for its negative and joker biases, then we will see shortly that λ_v serves as a normalizing constant (or significantly, a *partition function* [5]) that converts such weights into a proper probability.

$$\begin{aligned} \frac{d\mathcal{L}}{d\theta_v^-} = 0 &\Rightarrow \theta_v^- = \omega_v^- / \lambda_v, \text{ where } \omega_v^- = \left(\sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '-'}} Q(Z) \right) \\ \frac{d\mathcal{L}}{d\theta_v^*} = 0 &\Rightarrow \theta_v^* = \omega_v^* / \lambda_v, \text{ where } \omega_v^* = \left(\sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '*'}} Q(Z) \right) \end{aligned} \quad (7)$$

At this point it is worth noting the difference between regular BP/SP and EM-based methods. The weights here are constructed by summation and will be normalized into an arithmetic average. Similar BP/SP rules could have been made up merely by changing the sums into products! Doing so would produce a harsher geometric average. If we view message passing algorithms in terms of conjugate gradient ascent in the space of surveys, this in turn means that BP/SP will take larger steps than EM versions even when they are both following the same gradient to the same local optimum. These larger steps are the root of non-convergence for BP/SP. In applying the basic EM framework, we introduce logarithms and can thus use Jensen’s Inequality [4] to guarantee that our process will never increase in distance from the nearest local optimum. The summations in these weights can be seen as a reflection of these logarithms.

Finally, we solve for λ_v :

$$\begin{aligned} \theta_v^+ + \theta_v^- + \theta_v^* = 1 &\Rightarrow (\omega_v^+ + \omega_v^- + \omega_v^*) / \lambda_v = 1 \\ &\Rightarrow \lambda_v = \omega_v^+ + \omega_v^- + \omega_v^* \end{aligned} \quad (8)$$

Recall that a variable's three biases must sum to one. By substituting in the three expressions for the biases, we can solve for λ_v and confirm that it serves as a normalizer for ω_v^+ , ω_v^- , and ω_v^* . The completed update rules are listed below. All that remains is the E-Step, whereby we substitute for the $Q(Z)$ expressions that appear in the ω 's.

$$\theta_v^{+'} \leftarrow \frac{\omega_v^+}{\omega_v^+ + \omega_v^- + \omega_v^*} \quad \theta_v^{-'} \leftarrow \frac{\omega_v^-}{\omega_v^+ + \omega_v^- + \omega_v^*} \quad \theta_v^{*'} \leftarrow \frac{\omega_v^*}{\omega_v^+ + \omega_v^- + \omega_v^*} \quad (9)$$

6 E-Step

Recall that the EM framework can be bootstrapped by picking a random initial survey Θ . In this section we show how the current setting of Θ can be used to produce two different versions of $Q(Z)$; the versions differ in using local or global approximations and determine whether we produce EMBP-L versus EMBP-G, or EMSP-L versus EMSP-G.

Starting with a random Θ , the EM framework produces a $Q(Z)$ that is used in turn to produce a new Θ , repeating until guaranteed convergence. In fact, we will ultimately combine such E- and M-steps into a single update rule that expresses a new Θ' based on the old θ_v biases from the previous iteration.

In the following derivations, a specific form arises often enough to merit an abbreviation and some explanation:

$$\begin{aligned} \sigma(v, c) \triangleq \prod_{i \in V_{c^+} \setminus \{v\}} \theta_i^- \prod_{j \in V_{c^-} \setminus \{v\}} \theta_j^+ \\ \text{“}v \text{ is the sole support of } c\text{.”} \end{aligned} \quad (10)$$

The expression $\sigma(v, c)$ indicates the probability that variable v is the sole support of clause c . We iterate through all the other literals and consult their variables' biases toward the opposite polarity. In other words, the formula forms a product expressing the probability that the variables for all positive literals turned out constrained to be negative, and all negative literals turned out positive. Thus, c is solely dependent on v for support, and v must be positively or negatively constrained, based on whether c contains it as a positive or negative literal.

Under survey propagation, a variable is free to assume the ‘*’ value if there exists no clause for which it is the sole support. (Whether it actually *must* do so is controlled by the optional ρ parameter that is not discussed here [6].) Under regular belief propagation, there is no ‘*’ value available, codifying the assumption that every variable must be the sole support of *some* clause. Pragmatically speaking, this is like evenly dividing the weight assigned to θ_v^* between θ_v^+ and θ_v^- after every iteration.

At this point we address task of representing the $Q(Z)$ expression in the weighting formulas from the M-Step, as reproduced below for the case of ω_v^+ :

$$\begin{aligned}\omega_v^+ &= \left(\sum_{c \in C_v} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \right) \\ &= \sum_{c \in C_v^+} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) + \sum_{c \in C_v^-} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z)\end{aligned}\tag{11}$$

Intuitively, the weight is just the sum (over each of v 's clauses) of probabilities representing whether the dual configuration Z lists v as positive (within each clause.) Above we perform a preliminary step, of separating this summation into v 's positive and negative clauses, in order to consider the two distinct cases of whether v can be positive according to a clause where it appears as a positive literal, versus the more demanding situation where it appears as a negative literal.

How we express these two cases determines whether we will ultimately produce a EMBP or an EMSP rule, and whether it is of the local (“-L”) or global (“-G”) variety. We first address the question in the case of BP, under the local approximation scheme, producing EMBP-L:

$$\begin{aligned}\mathbf{EMBP-L:} \quad \omega_v^+ &= \sum_{c \in C_v^+} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) + \sum_{c \in C_v^-} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \\ &= \sum_{c \in C_v^+} [1] + \sum_{c \in C_v^-} [1 - \sigma(v, c)] \\ &= |C_v^+| + |C_v^-| - \sum_{c \in C_v^-} \sigma(v, c) \\ &= |C_v| - \sum_{c \in C_v^-} \sigma(v, c) \\ \omega_v^- &= |C_v| - \sum_{c \in C_v^+} \sigma(v, c) \\ \omega_v^* &= 0\end{aligned}\tag{12}$$

Consider the first summation over v 's positive clauses. We will not literally create any sort of a table listing probabilities for all possible total clause configurations that could instantiate Z . Rather, we note that if we could consult such a table, we would iterate through its entries and construct a running sum of probabilities for entries that make v positive for the current clause c . The local approximation is so-called because it corresponds to generalized arc-consistency; a necessary (but not sufficient) condition for v to be assigned ‘+’ is for all the other variables appearing in the clause to be set in

some way that allows it. For this first summation, there is not really any reason at all why v could not be ‘+’; after all it appears as a positive literal in each c . So, there is a probability of 1 that the other variables in the clause will allow this value, i.e. that they will be generalizedly arc-consistent. Thus, we can replace the inner summation with the constant value of 1. In so substituting we will enable a single update rule that performs both of the E- and M- Steps at the same time, by directly representing the E-Step within the rule derived during the M-Step.

Also, because generalized arc-consistency is necessary but not sufficient, such a substitution represents a (trivial) upper bound on the probability in question. Later we will be able to make tighter bounds by moving to more global consistency and to the survey propagation model.

Turning to the second summation, over clauses where v appears as a negative variable, we see that it is possible for the other variables to be unsupportive of v ’s assignment to ‘+’. Specifically, if all other variables are constrained not to satisfy the clause, then v must not be ‘+’. In other words, the assignment $v = ‘+’$ is not generalizedly arc-consistent with the other variables exactly when it is the sole support, as denoted by $\sigma(v, c)$. Thus, instead of explicitly going through all possible extensions as represented by Z and summing the probabilities of those that assign v to ‘+’, we can analytically substitute a single expression for this entire sum. As show in the second line above, this expression is $1 - \sigma(v, c)$.

The remaining lines for ω_v^+ simplify summations over the constant term 1 into cardinalities of the sets being summed over. An analogous process applies for ω_v^- , and ω_v^* can just be set to zero since we are using the BP model.

An alternative way of representing the summation over $Q(Z)$ applies a global rather than a local sense of consistency to BP, producing the EMBP-G update rule:

$$\begin{aligned}
 \text{EMBP-G: } \omega_v^+ &= \sum_{c \in C_v^+} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) + \sum_{c \in C_v^-} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \\
 &= \sum_{c \in C_v^+} [1] + \sum_{c \in C_v^-} \left[\prod_{c \in C_v^-} (1 - \sigma(v, c)) \right] \\
 &= |C_v^+| + |C_v^-| \left[\prod_{c \in C_v^-} (1 - \sigma(v, c)) \right] \tag{13} \\
 \omega_v^- &= |C_v^-| + |C_v^+| \left[\prod_{c \in C_v^+} (1 - \sigma(v, c)) \right] \\
 \omega_v^* &= 0
 \end{aligned}$$

As before, there is no reason why a positive clause would prohibit v from holding the value ‘+’, so the first substitution is again for the constant 1. But turning to the second summation over negative clauses, we take a more global view of consistency in assessing whether v can be positively constrained. Whereas generalized arc-consistency

did not care about whether other negative clauses will allow this assignment so long as $z_c[v] = '+'$ for the current clause, we will now enforce consistency across all negative clauses. For example, if $c1 = (\neg x \vee y_1 \vee z_1)$ and $c2 = (\neg x \vee y_2 \vee z_2)$, then local consistency would implicitly include models where $z_{c1}[x]$ chooses '+' even if $z_{c2}[x]$ holds the value '-'. Under global consistency, we say that a negative clause can hold an extension where v is '+' only if *all* negative clauses allow it. Thus we substitute a product of event probabilities whereby all negative clauses do not need v as a sole support.

This produces a tighter upper bound on the sum of $Q(Z)$'s for Z 's wherein clause c considers v '+'. Why use a local consistency rather than a global consistency approximation, then? For other situations like CSP's, or even for the SP case in SAT, such an approximation can be much faster to compute than a global one. In the particular case of EMBP-L vs. EMBP-G the computational cost is comparable—but it was important to empirically confirm that EMBP-G's tighter bound does in fact eventually yield a better bias estimator than EMBP-L's, and ultimately a better search heuristic [1].

The rest of the above equations again group constant terms, and present the analogous result for ω_v^- . Again ω_v^* is set to 0. Next, we lift this latter condition by switching to the SP model. Now variables can be set to the '*' state whereby they are not the sole support of *any* clause. Once again we start with local (generalized arc-) consistency.

$$\begin{aligned}
\text{EMSP-L: } \omega_v^+ &= |C_v| - \sum_{c \in C_v^-} \sigma(v, c) \\
\omega_v^- &= |C_v| - \sum_{c \in C_v^+} \sigma(v, c) \\
\omega_v^* &= \sum_{c \in C_v^+} \sum_{\substack{Z: z_c[v] \\ = '*'}} Q(Z) + \sum_{c \in C_v^-} \sum_{\substack{Z: z_c[v] \\ = '*'}} Q(Z) \\
&= \sum_{c \in C_v^+} [1 - \sigma(v, c)] + \sum_{c \in C_v^-} [1 - \sigma(v, c)] \\
&= |C_v| - \sum_{c \in C_v} \sigma(v, c)
\end{aligned} \tag{14}$$

The first two rules of EMSP-L, for ω_v^+ and ω_v^- are actually the same as before, with EMBP-L. We now consider the question of whether v can be '*', or unconstrained. We use the same upper bound as we used on whether a negative clause could make v positive: v cannot be the sole support of such a clause. Here, though, we make the substitution twice: v can be the sole support of neither a positive *nor* a negative clause. Thus the end result is similar to the other two rules, but iterates over all clauses.

We now derive the final update rule by applying global consistency to the SP framework, resulting in EMSP-G:

$$\begin{aligned}
 \text{EMSP-G: } \omega_v^+ &= \sum_{c \in C_v^+} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) + \sum_{c \in C_v^-} \sum_{\substack{Z: z_c[v] \\ = '+'}} Q(Z) \\
 &= \sum_{c \in C_v^+} \left[1 - \prod_{c \in C_v^+} (1 - \sigma(v, c)) \right] + \sum_{c \in C_v^-} \left[\prod_{c \in C_v^-} (1 - \sigma(v, c)) \right] \\
 &= |C_v^+| \left[1 - \prod_{c \in C_v^+} (1 - \sigma(v, c)) \right] + |C_v^-| \prod_{c \in C_v^-} (1 - \sigma(v, c)) \\
 \omega_v^- &= |C_v^-| \left[1 - \prod_{c \in C_v^-} (1 - \sigma(v, c)) \right] + |C_v^+| \prod_{c \in C_v^+} (1 - \sigma(v, c)) \\
 \omega_v^* &= \sum_{c \in C_v^+} \sum_{\substack{Z: z_c[v] \\ = '*'}} Q(Z) + \sum_{c \in C_v^-} \sum_{\substack{Z: z_c[v] \\ = '*'}} Q(Z) \\
 &= \sum_{c \in C_v^+} \prod_{c \in C_v^+} (1 - \sigma(v, c)) + \sum_{c \in C_v^-} \prod_{c \in C_v^-} (1 - \sigma(v, c)) \\
 &= |C_v| \prod_{c \in C_v} (1 - \sigma(v, c))
 \end{aligned} \tag{15}$$

Here, when we determine whether v can be '+', we must consider some stricter criteria than before, with BP or with local consistency. We begin with the sum of probabilities over Z 's where a positive clause c allows the assignment $v = '+'$. Under global consistency, it is not enough that just c allows it; we should consider all the other positive clauses. And under the survey propagation model, it is not enough that all positive clauses simply allow it; one of them must actively *require* it, or else the actual state of v is '*' rather than '+'. Thus for each positive clause we substitute in the probability that *some* positive clause requires v as its sole support. In the second equation this is realized as the negation of the event that *all* positive clauses *do not* require v .

For the second summation, over negative clauses, we use the same expression that we used before for EMBP-G: an individual negative clause is allowed to assign '+' to v only if *no* negative clause needs v for sole support. This completes the rule for ω_v^+ , and an analogous process holds for ω_v^- .

The final step is to express the sum of $Q(Z)$'s for Z 's that allow positive or negative clauses to assign '*' to v . In either case, this is just the probability that *all* positive or negative clauses do not need v . So we make the same sort of substitution into both summations, over positive and negative clauses, completing the rule for ω_v^* .

7 Conclusion

This completes the derivation of the four EM-based methods for bias estimation in SAT. All methods are guaranteed to converge, but still seek to maximize the same log-likelihood as regular BP and SP. When BP and SP do converge, they can return better or worse answers than these EM-based methods. For the SAT problems considered in [1], the EM-based methods were generally better bias estimators, and ultimately produced more accurate heuristics for actual SAT-Solving. For reference, the completed rules are presented together at the end of this document.

References

1. Hsu, E., Muise, C., Beck, C., McIlraith, S.: From backbone to bias: Probabilistic inference and heuristic search performance. In: Currently under review. (2008)
2. Hsu, E., Kitching, M., Bacchus, F., McIlraith, S.: Using EM to find likely assignments for solving CSP's. In: Proc. of 22nd National Conference on Artificial Intelligence (AAAI '07), Vancouver, Canada. (2007)
3. Hsu, E., McIlraith, S.: Characterizing propagation methods for boolean satisfiability. In: Proc. of 9th International Conference on Theory and Applications of Satisfiability Testing (SAT '06), Seattle, WA. (2006)
4. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* **39**(1) (1977) 1–39
5. Jordan, M.I., Ghahramani, Z., Jaakkola, T., Saul, L.K.: An introduction to variational methods for graphical models. *Machine Learning* **37**(2) (1999) 183–233
6. Maneva, E., Mossel, E., Wainwright, M.: A new look at survey propagation and its generalizations. *Journal of the ACM* **54**(4) (2007) 2–41

$\sigma(v, c) \triangleq \prod_{i \in V_c^+ \setminus \{v\}} \theta_i^- \prod_{j \in V_c^- \setminus \{v\}} \theta_j^+$ <p>(a) $\sigma(v, c)$: v is the sole support of c.</p>	$\theta_v^{+'} \leftarrow \frac{\omega_v^+}{\omega_v^+ + \omega_v^-} \quad \theta_v^{-'} \leftarrow \frac{\omega_v^-}{\omega_v^+ + \omega_v^-}$ <p>(b) Bias normalization for BP methods.</p>
$\theta_v^{+'} \leftarrow \frac{\omega_v^+}{\omega_v^+ + \omega_v^- + \omega_v^*} \quad \theta_v^{-'} \leftarrow \frac{\omega_v^-}{\omega_v^+ + \omega_v^- + \omega_v^*} \quad \theta_v^{*'} \leftarrow \frac{\omega_v^*}{\omega_v^+ + \omega_v^- + \omega_v^*}$ <p>(c) Bias normalization for SP methods.</p>	

Fig. 1. Formula for “sole-support”; normalizing rules for BP and SP families of bias estimators.

$\omega_v^+ = \prod_{c \in C_v^-} (1 - \sigma(v, c))$	$\omega_v^+ = C_v - \sum_{c \in C_v^-} \sigma(v, c)$
$\omega_v^- = \prod_{c \in C_v^+} (1 - \sigma(v, c))$	$\omega_v^- = C_v - \sum_{c \in C_v^+} \sigma(v, c)$
(a) Regular BP update rule.	(b) EMBP-L update rule.
$\omega_v^+ = C_v^- \left[\prod_{c \in C_v^-} (1 - \sigma(v, c)) \right] + C_v^+ $	
$\omega_v^- = C_v^+ \left[\prod_{c \in C_v^+} (1 - \sigma(v, c)) \right] + C_v^- $	
(c) EMBP-G update rule.	

Fig. 2. Update rules for the Belief Propagation (BP) family of bias estimators.

$\omega_v^+ = \prod_{c \in C_v^-} (1 - \sigma(v, c)) \cdot \left[1 - \prod_{c \in C_v^+} (1 - \sigma(v, c)) \right]$	$\omega_v^+ = C_v - \sum_{c \in C_v^-} \sigma(v, c)$
$\omega_v^- = \prod_{c \in C_v^+} (1 - \sigma(v, c)) \cdot \left[1 - \prod_{c \in C_v^-} (1 - \sigma(v, c)) \right]$	$\omega_v^- = C_v - \sum_{c \in C_v^+} \sigma(v, c)$
$\omega_v^* = \prod_{c \in C_v} (1 - \sigma(v, c))$	$\omega_v^* = C_v - \sum_{c \in C_v} \sigma(v, c)$
(a) Regular SP update rule.	(b) EMSP-L update rule.
$\omega_v^+ = C_v^- \prod_{c \in C_v^-} (1 - \sigma(v, c)) + C_v^+ \left[1 - \prod_{c \in C_v^+} (1 - \sigma(v, c)) \right]$	
$\omega_v^- = C_v^+ \prod_{c \in C_v^+} (1 - \sigma(v, c)) + C_v^- \left[1 - \prod_{c \in C_v^-} (1 - \sigma(v, c)) \right]$	
$\omega_v^* = C_v \prod_{c \in C_v} (1 - \sigma(v, c))$	
(c) EMSP-G update rule.	

Fig. 3. Update rules for the Survey Propagation (SP) family of bias estimators.