Using EM to Derive a Convergent Alternative to Loopy Belief Propagation

University of Toronto Technical Report CSRG-579

Eric I. Hsu and Sheila A. McIlraith
Department of Computer Science
University of Toronto
{eihsu,sheila}@cs.toronto.edu

Abstract

EMBP is a variant of Belief Propagation (BP) that always converges, even on graphical models with loops. Its initial *ad hoc* development was driven by the search for likely variable assignments in Constraint Satisfaction problems, but in general it can estimate marginal probabilities over any model that BP can. Thus we derive a canonical version of EMBP from first principles by applying the Expectation Maximization (EM) framework. In so doing we re-characterize the inference task as parameter estimation over a redundant problem representation combining the primal and dual versions of a graphical model, yielding a new technique whose formula is remarkably similar to BP's despite guaranteeing convergence.

1 Introduction

Belief Propagation [12] (a.k.a. the Sum-Product algorithm [9]) has proved highly successful when applied in various forms to Boolean Satisfiability and the more general Constraint Satisfaction Problem (CSP) [2, 1]. Problems from such domains can be represented as factor graphs that BP uses to weigh the likelihood of possible variable assignments, guiding the search for a satisfying configuration of all the variables. But while exact on tree graphical models, BP is inexact and non-convergent on arbitrary graphs containing loops. EMBP was developed within this context as a convergent variant of BP [6]. It is also inexact, but for the specialized CSP domains considered to date, it typically produces more useful results than BP–even in trials where BP employs fairly sophisticated termination techniques or otherwise manages to converge [6, 5, 7]. Until now, though, EMBP has only been represented on an *ad hoc* basis for problems in this domain, for lack of any formal characterization.

Here we demonstrate the use of Expectation Maximization (EM) [3, 11] to derive a general form for EMBP from first principles. EM is traditionally used to fit a series of observations Y to a model parameterized by Θ in the presence of hidden variables Z. Starting from an initial random setting of Θ, the E-Step constructs a $Q(Z)$ distribution estimating the likelihood of hidden variable configurations with respect to Θ. Next the M-Step assigns Θ a new value maximizing the expected log-likelihood of the observations according to the current Q. The two steps are repeated to guaranteed convergence at a local maximum in likelihood.

Abstracting away from domain-specific special cases and approximations uncovers a primal/dual statistical framework that "re-derives" a formula very similar to BP using a redundant reformulation of a given graphical model. (The presentation is in terms of factor graphs [9] for ease of expression but as with BP, EMBP can be applied easily to other models like Bayes nets or Markov random fields [15].) The ultimate goal is to clarify the relationship between EM, BP, and EMBP, so that a single canonical representation can be applied to future problems with the same fine control of efficiency and approximation that characterizes the general family of variational methods. So, after defining notation and motivational graph transformations in Section 2, the bulk of this presentation will concern the derivation of EMBP using EM, in Section 3. This allows comparison with other methods and some conclusions inviting future study, in Sections 4 and 5.

2 Definitions and Terminology

A factor graph is a bipartite graph containing two types of nodes: variables and functions (factors). Let $X = \{X_1, X_2, \ldots, X_n\}$ be the set of **variables** in a given factor graph. Typically, we will use 'i' as an index over this set. Let $D = \{v_1, v_2, \ldots, v_d\}$ be the set of **values** that each X_i can hold. In actuality each variable can have its own arbitrary finite domain, but here we make the domains identical to simplify notation. Values in D will typically be indexed by the subscript 'j'. A **configuration** $C : X \rightarrow D$ is a function representing an assignment to all the variables in X. A factor graph encodes a probability density function over the possible configurations of its variables.

Let $F = \{f_1, f_2, \ldots, f_m\}$ be the set of **factors**, (a.k.a. "functions") in a factor graph; their typical subscript will be 'a'. Associated with each factor f_a is a **signature** $\sigma_a \subseteq X$ representing the set of variables that serve as arguments to f_a. Thus each $f_a : D^{|\sigma_a|} \rightarrow \mathbb{R}^+ \cup 0$ is a function that receives assignments to the variables in its signature, and maps them to the non-negative reals.

An **extension** (a.k.a. "tuple") $z_a : \sigma_a \rightarrow D$ represents an assignment to all the variables in the signature of some factor f_a. Thus, "$z_a(X_i) = v_j$" indicates that the extension z_a assigns the value v_j to variable X_i. Extensions z_a and z_b to two different factors f_a and f_b are **consistent** iff they make the same assignment to any variables their factors have in common: $\forall X_i \in \sigma_a \cap \sigma_b$, $z_a(X_i) = z_b(X_i)$. We stretch notation a bit further by expressing the **instantiation** of a function by some extension as $f_a(z_a) \in \mathbb{R}^+ \cup \{0\}$. This represents the value of f_a when evaluated with its arguments assigned to the values specified by z_a.

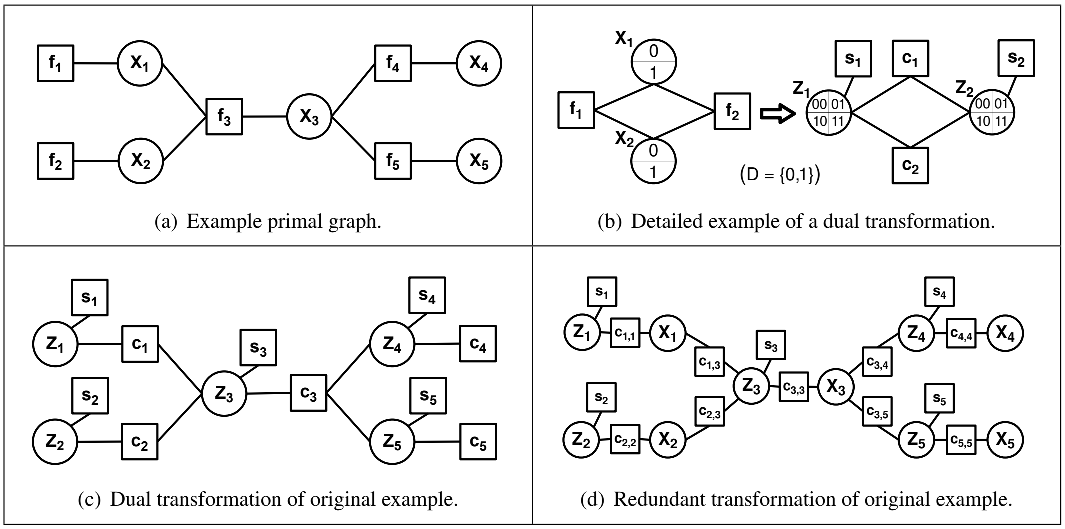
Given a factor graph, we wish to compute the marginal probability for each variable to hold each of the possible values in D. The probabilities can be represented by a matrix $\Theta \in [0,1]^{n \times d}$ where each row contains d multinomial parameters for a given variable. For clarity, then, we can use $\theta_i[v_j] \triangleq \theta[i,j]$ to denote the probability that variable X_i holds value v_j. Thus each θ_i must act as a proper probability distribution, as in, $\forall i : 1 \leq i \leq n$, $\sum_{j=1}^{d} \theta_i[v_j] = 1$. For consistency with [6, 5] we will call θ_i the (multinomial) **bias distribution** for a variable X_i, and refer to each $\theta_i[v_j]$ as an individual **bias**. (The term "marginal probability" can be freely substituted for "bias".)

2.1 Graph Transformations: the Primal, Dual, and Redundant Graphs

Primal Graph. A factor graph can be denoted by its variables and factors alone; the edges are determined by the factor signatures. A given factor graph $G = (X, F)$ specifies a probability distribution over configurations of X as a product of factor instantiations: $p(x_1, \ldots, x_n) = \prod_{a=1}^{m} f_a(X_a)$. In the equation, $(x_1, \ldots, x_n)$ represents a vector of values assigned to $X_1, \ldots, X_n$, and X_a is the projection of this configuration onto the signature σ_a. We will call this original version of the problem the primal graph, as exemplified in Figure 1(a). The ultimate goal is to compute marginal distributions for individual variables by exploiting the factorization inherent in the graph's structure. But to derive a new method for doing so, it is illuminating to consider two transformations to this graph.

Dual Graph. Let $G = (X, F)$ be a factor graph with variables, values, and factors indexed as in the previous definitions. Then its dual formulation DUAL$(G) = (Z, C \cup S)$ consists of dual variables $Z = \{z_1, \ldots z_m\}$ representing extensions to corresponding factors in F, plus factors $C = \{c_1, \ldots c_n\}$ and $S = \{s_1, \ldots s_m\}$ representing "consistency" and "scoring," respectively. The basic idea is to convert factors into variables representing tuples and variables into c-factors enforcing consistency between tuples. Additional s-factors score tuples by simulating the original factors.

Thus if Z_i is a vector of all extensions whose associated signatures contain variable X_i, then $c_i(Z_i) = \prod_{z_a, z_b \in Z_i} \delta(z_a(X_i) = z_b(X_i))$, where $\delta()$ is the Kronecker delta function that evaluates to 1 or 0 based on the truth of a given proposition. Further, to each dual variable z_a we connect a scoring factor s_a where $s_a(z_a) = f_a(z_a)$. In



(a) Example primal graph.

(b) Detailed example of a dual transformation.

(c) Dual transformation of original example.

(d) Redundant transformation of original example.

Figure 1: Transformations of an example factor graph.

other words, each s_a simulates the result of instantiating factor f_a from the original graph with the extension z_a prescribed by the dual graph. Figure 1(b) illustrates this process for a four-node factor graph where the variable domain is $D = \{0, 1\}$. Factors f_1 and f_2 both have signature $\sigma_1 = \sigma_2 = \{X_1, X_2\}$. So in the dual they become variables Z_1 and Z_2, which share a common domain consisting of the cross product of X_1's and X_2's: $z_1, z_2 \in \{\langle 0,0\rangle, \langle 0,1\rangle, \langle 1,0\rangle, \langle 1,1\rangle\}$. Consistency factor c_1 ensures that the two extension variables agree on the value of X_1: $c_1(z_1, z_2) = \delta(z_1(X_1) = z_2(X_1))$, and likewise for c_2 with respect to X_2. Finally, scoring factor s_1 simulates the instantiation of f_1 with z_1. For example, $s_1(\langle 0,1\rangle) = f_1(0, 1)$. Figure 1(c) depicts the dual transformation of the factor graph in 1(a).

Redundant Graph. The dual graph can be further transformed by replacing its consistency factors with variable nodes directly representing the original variables from the primal graph. These nodes are then connected to the rest of the graph by introducing new consistency factors of the form $c_{i,a}$ ensuring that any relevant extension Z_a uses the value for X_i that is specified by its node. Thus, the term "redundant" is meant to reflect how a variable's value is encoded by a graph configuration multiple times: once in its variable node and once in each of the extension nodes representing its relevant primal factors. More formally, the redundant transformation of a primal graph $G = (X, F)$ is REDUNDANT$(G) = (X \cup Z, C' \cup S)$, where Z and S are defined as in DUAL(G), and $C' \subseteq \{c_{i,a}, 1 \leq i \leq n, 1 \leq a \leq m\}$. Each $c_{i,a}$ connects a primal variable X_i to a subsuming dual variable Z_a, ensuring consistency: $c_{i,a}(v_j, z_a) = \delta(v_j = z_a(X_i))$. For instance, in Figure 1(d) the factor $c_{1,1}$ ensures that the chosen tuple for Z_1 projects X_1 onto the same value held by the node for X_1, while the factor $c_{1,3}$ ensures that the chosen value of Z_3 does the same.

If factors in a primal graph are viewed as potential functions over cliques of variables, then the dual variables Z can be viewed as explicit statements of such struc-

Vector	Status	Interpretation	Domain
Y	Observed	***Consistency***: whether variables agree with extensions, *i.e.* all $c_{i,a}$ factors evaluate to 1.	$\{0,1\}^{nm}$ *(notional)*
Z	Unobserved	***Extensions***: tuples of variable assignments that instantiate the original factors.	$\{z_a\}^m, \;\; z_a : \sigma_a \rightarrow D$
Θ	Parameters	***Biases***: marginal distributions over original variables.	$\{\theta_i\}^n, \;\; \theta_i : D \rightarrow [0,1]$, $\sum_{j=1}^{d} \theta_i[v_j] = 1$.

Table 1: EM formulation for estimating marginals using the redundant factor graph transformation.

ture. Further, their surrounding c-factors embody the "running intersection" property between such cliques that figures prominently in variable elimination techniques. So while adjacent nodes in "join-trees" (a.k.a. "junction-trees", etc. [12]) might pass messages about shared variables, such sets of variables are now represented explicitly and individually as extension variables in the redundant graph.

3 Deriving the EMBP Update Rule within the EM Framework.

Here we show how to derive EMBP by treating the extraction of marginals from a factor graph as a maximum-likelihood parameter estimation task within the EM framework. The derivation is motivated by the redundant transformation of a factor graph, but it is different from simply running the standard sum-product algorithm on the transformed graph. Rather, if we view a distribution over configurations from a frequentist perspective, then we can imagine that such configurations represent samples from a generative process driven by the marginal probabilities of individual variables. That is, a factor graph is in itself a sufficient statistic for a hypothetical process of sampling complete assignments to all of its variables. If we assume an approximate (mean-field [8]) model whereby such configurations are generated by individual variables operating independently, then their individual marginal probabilities constitute the parameters of this model. Thus the task of estimating bias is reformulated as the task of fitting an approximate model to an imaginary series of observations.

It is also worth noting here that the resulting methodology will not involve an explicit sampling process. Instead we will optimize the parameters setting Θ with respect to a $Q()$ distribution that captures the relative weights of such samples in closed form. To that end, Table 1 shows how a factor graph can be recast in terms of the observations Y, hidden variables Z, and parameters Θ of the EM framework. Intuitively, we will tell EM that we have indirect access to some randomly sampled configuration of the variables in a factor graph. We posit that there is some underlying configuration to the dual variables (factor extensions), but that we were unable to observe this configuration Z. All we know is that the extensions are consistent with one another: $Y = [1]^{nm}$. (This constant setting of Y is conceptual and does not have to be manipulated directly when deriving EMBP.) Ultimately we ask the framework to hypothesize a (consistent) dual configuration via a $Q()$ distribution, and to find the most likely marginals Θ on individual variables with respect to $Q()$.

In terms of the redundant factor graph, this can be viewed as a primal-dual optimization process where we softly assign factor extensions to their most likely values according to the biases of variables in their signatures, and the weights that the factors assign to resulting instantiations. In turn, we softly assign variables to their most likely values with regard to the most likely extensions. The first operation is the E-Step, and the second is the M-Step; the two are repeated to guaranteed convergence, even on factor graphs with loops.

3.1 M-Step

In presenting the derivation we begin with the M-Step for clarity. Here our basic goal is to find variable biases Θ that maximize the expected log-likelihood of consistency Y within some hidden set of factor extensions Z. The expectation is with respect to an artificial distribution over extensions $Q(Z)$ that we will construct in the subsequent E-Step. So, we set $\Theta' = \text{argmax}_{\Theta}\, \mathcal{L}(\Theta)$, where $\mathcal{L}(\Theta) = \mathcal{F}(\Theta) - \mathcal{P}(\Theta)$ is a Lagrangian function comprised of the expectation in question and a penalty term that forces each θ_i to be a proper probability distribution:

$$\mathcal{F}(\Theta) = E_Q[\log P(Z,Y|\Theta)] \qquad \text{and} \qquad \mathcal{P}(\Theta) = \sum_{i=1}^{n} \lambda_i (\sum_{j=1}^{d} \theta_i[v_j] - 1) \tag{1}$$

The penalty term $\mathcal{P}(\Theta)$ introduces a series of Lagrange multipliers, one for each variable X_i, that allows unlimited penalty for any bias distribution that does not sum to exactly one. The main object of interest is the expected log-likelihood represented by $\mathcal{F}(\Theta)$:

$$\begin{align}
\mathcal{F}(\Theta) &= E_Q[\log P(Z,Y|\Theta)] \tag{2a}\\
&= \sum_{Z} Q(Z)\, \log\, P(Z,Y|\Theta) \tag{2b}\\
&\cong \sum_{Z} Q(Z)\, \log\, \prod_{a=1}^{m} p(z_a|\Theta) \tag{2c}\\
&\cong \sum_{Z} Q(Z)\, \log\, \prod_{a=1}^{m} \prod_{\substack{i:\, X_i \\ \in \sigma_a}} \theta_i[z_a(X_i)] \tag{2d}\\
&= \sum_{Z} Q(Z) \sum_{a=1}^{m} \sum_{\substack{i:\, X_i \\ \in \sigma_a}} \log\, \theta_i[z_a(X_i)] \quad = \quad \sum_{a=1}^{m} \sum_{Z} Q(Z) \sum_{\substack{i:\, X_i \\ \in \sigma_a}} \log\, \theta_i[z_a(X_i)] \tag{2e}
\end{align}$$

Our goal is to express the expected log-likelihood in a form that can be readily optimized in terms of $Q()$. Recall that Y indicates the observation of consistency between extensions; if we consider only consistent values of Z from this point forward, we can drop Y from future steps.

Thus, Equation (2c) focuses on consistent dual configurations, decomposing them into individual extensions. Equation (2d) further decomposes each such extension into the individual probabilities of the variable assignments it represents: the probability of generating a specific tuple z_a is the product of probabilities for generating the prescribed assignment $z_a(X_i)$ for each of the variables X_i in its signature. Such probabilities are precisely the biases represented by θ_i. Note that both of these decompositions represent mean-field approximations that upper-bound the free energy underlying our EM formulation. Thus, while EMBP is convergent, its results are approximate.

Equation (2e) rearranges the expression by converting logarithms of products into sums of logarithms, and by exchanging summations. Now we are ready to recombine $\mathcal{F}(\Theta)$ with $\mathcal{P}(\Theta)$ and optimize the convex function $\mathcal{L}(\Theta)$ by setting its first derivative to zero:

$$\frac{d\mathcal{L}}{d\theta_i[v_j]} = 0 \quad \Rightarrow \quad \left(\sum_{\substack{a:\, X_i \\ \in \sigma_a}} \; \sum_{\substack{Z:\, z_a(X_i) \\ = v_j}} Q(Z) \cdot \frac{1}{\theta_i[v_j]} \right) - \lambda_i = 0 \tag{3a}$$

$$\Rightarrow \quad \theta_i[j] = \left(\sum_{\substack{a:\, X_i \\ \in \sigma_a}} \; \sum_{\substack{Z:\, z_a(X_i) \\ = v_j}} Q(Z) \right) / \, \lambda_i \tag{3b}$$

$$\Rightarrow \quad \lambda_i = \sum_{j=1}^{d} \left(\sum_{\substack{a:\, X_i \\ \in \sigma_a}} \; \sum_{\substack{Z:\, z_a(X_i) \\ = v_j}} Q(Z) \right) \tag{3c}$$

We treat each bias $\theta_i[v_j]$ as a variable and differentiate $\mathcal{L}(\Theta)$ with respect to each such component of Θ. Thus Equation (3b) constitutes an M-Step rule for updating a variable X_i's bias toward value v_j with respect to a fixed $Q()$. The numerator directs us to sum through the signatures in which X_i appears, and add up the $Q(Z)$-weights of all dual configurations Z wherein the corresponding extension has X_i set to v_j. The denominator is just a normalizing function to make θ_i a proper distribution, as in (3c); this can be confirmed by summing both sides of (3b) over j from 1 to d.

In reference to the redundant graph in Figure 1(b), the formula specifies, for example, how to set the multinomial parameter for the assignment $X_3 = v$ according to a presumed $Q()$ distribution over values of Z_3, Z_4, and Z_5. For each of these extensions, we determine the marginal probability that they project X_3 to v, by summation. These values are then *arithmetically* averaged to create an overall weight for X_3 to hold the value v. Weights for other values in D can be constructed in the same way, and then normalized to form a proper probability distribution θ_3.

3.2 E-Step

For the second part of our derivation we use an existing specification of variable marginals Θ to construct a $Q()$ distribution over factor extensions. Note, however, that we do not necessarily have to construct an explicit table of probabilities over all possible consistent configurations to a dual problem. If in possession of such a table, we would only use it for the specific operation of summing rows where specific extensions project specific variables to specific values, as in the numerator of the M-Step (3b). The relevant operation is reproduced as the left-hand expression below:

$$\sum_{\substack{a:\,X_i\\ \in\sigma_a}} \sum_{\substack{Z:\,z_a(X_i)\\ =v_j}} Q(Z) \quad\cong\quad \sum_{\substack{a:\,X_i\\ \in\sigma_a}} \sum_{\substack{z_a:\,z_a(X_i)\\ =v_j}} q_a(z_a) \tag{4}$$

In previous projects the left-hand side has been handled directly in terms of "global consistency." [4]. But in general we can decompose $Q(Z)$ into a set of independent $\{q_a(Z_a), 1 \leq a \leq m\}$ distributions over extensions, just as we applied the mean-field approximation to the primal variables. Under this approximation, we now need only to consider the sum of q_a probabilities for individual extensions z_a that support a specific variable assignment $X_i = v_j$.

Taken literally, the time complexity of this process is exponential over the size of σ_a, but for specific applications it can be reduced dramatically, for instance to linear time over signatures in the case of SAT. Still, here we present the process in its most general form: we determine the probability that an extension supports a particular variable assignment by marginalizing out all of its remaining variables. This is analogous to the "summary" operation in, for example, [9], and similarly, it encumbers us with some extra notation.

The **s-signature** $\sigma_{a-i} = \sigma_a - \{X_i\}$ of a factor f_a with respect to a variable X_i represents the set of arguments to f_a, excluding X_i.

The **s-extension** $z_{a-i} : \sigma_{a-i} \rightarrow D$ of f_a with respect to X_i is an extension representing an assignment to to all the arguments of f_a except for X_i.

The **s-instantiation** $f_a(z_{a-i}, X_i = v_j) \in \mathbb{R}^+ \cup \{0\}$ of a factor f_a with respect to a corresponding s-extension z_{a-i} and variable assignment $X_i = v_j$ represents the value of f_a when evaluated with X_i set to v_j and all other arguments set according to z_{a-i}.

Using this notation we can finally express the summary process for estimating the probability of a variable assignment with respect to a specific factor:

$$\sum_{\substack{z_a:\,z_a(X_i)\\ =v_j}} q_a(z_a) \quad\propto\quad \sum_{z_{a-i}} \left(\prod_{\substack{i':\,X_{i'}\\ \in\sigma_{a-i}}} \theta_{i'}[z_{a-i}(X_{i'})] \right) \cdot f_a(z_{a-i}, X_i = v_j) \tag{5}$$

The right-hand side can be normalized across all extensions to produce proper equality, but because we will ultimately substitute this expression into the M-Step (Equation (3b)), which will itself be normalized, the step is unnecessary here. The formula weights the probability of tuple z_a mapping X_i to v_j by iterating through all dual values that do so and summing their individual probabilities.

Estimating such individual probabilities decomposes into the two factors at the right-hand side of Eq. (5). The first exhibits a generative flavor in considering the likelihood of producing a given tuple by way of the individual marginals of its constituent variables. The second expression factors in the *a priori* likelihood of ever seeing this tuple, by using it to instantiate the associated factor.

Returning to the redundant graph in Figure 1(d), for example, and assuming a domain of $D = \{0, 1\}$, we can apply Eq. (4) to the assignment $X_3 = 0$. For the M-Step, we only need to know the sum over extensions Z_1, Z_2, Z_3 of the sum of $Q(Z)$ probabilities for dual configurations wherein the current extension includes $X_3 = 0$. By the equation, we can approximate the term for Z_3, for example, by consulting a decomposed $q_3()$ distribution over of series of extension values z_3 that project $z_3(X_3) = 0$. Equation (5) gives the expression that we substitute for this sum of $q_3(z_3)$'s:

$$\sum_{\substack{z_3\colon z_3(X_3)\\ =0}} q_3(z_3) \quad \propto \quad \begin{array}{rl} & \theta_1[0] \cdot \theta_2[0] \cdot f_3(0,0,0) \\ + & \theta_1[0] \cdot \theta_2[1] \cdot f_3(0,1,0) \\ + & \theta_1[1] \cdot \theta_2[0] \cdot f_3(1,0,0) \\ + & \theta_1[1] \cdot \theta_2[1] \cdot f_3(1,1,0) \end{array} \tag{6}$$

For each pair of assignments for X_1 and X_2, we multiply the likelihoods of generating these two assignments (according to θ_1 and θ_2) with the value of f_3 when they are combined with the assignment $X_3 = 0$. (In the above example we assumed that the arguments to f_3 are ordered lexicographically, as in, $f_3(X_1, X_2, X_3)$.)

3.3 The EMBP Update Rule

To complete the derivation of EMBP we can substitute Equations (4) and (5) into (3b), producing a single update rule for individual biases. (Again the denominator $\mathcal{N}_i$ is a normalizing value representing the summation of the numerator over $1 \leq j \leq d$):

$$\boxed{\theta_i[v_j]' \leftarrow \sum_{\substack{a\colon X_i\\ \in\sigma_a}} \left[\sum_{z_{a-i}} \left(\prod_{\substack{i'\colon X_{i'}\\ \in\sigma_{a-i}}} \theta_{i'}[z_{a-i}(X_{i'})] \right) \cdot f_a(z_{a-i}, X_i = v_j) \right] / \mathcal{N}_i} \tag{7}$$

The overall EMBP algorithm begins by selecting an initial Θ at random. We then consider each variable X_i in sequence, calculating the numerator of Equation (7) for each of X_i's possible values. Summing these weights produces the normalizer $\mathcal{N}_i$ and we can now update the bias distribution θ_i before proceeding in sequence to the next variable. We can call a single pass through all the variables an "iteration;" at the end of each we check whether any bias has changed since the previous iteration. If so, we begin a new pass through the variables; otherwise we terminate with Θ set to a local maximum in approximate likelihood. Due entirely to the properties of EM, the overall EMBP algorithm is guaranteed to converge after some number of iterations.

The complexity of a single iteration is bounded by $\mathcal{O}(m^2 n d^k)$, where k is the size of the largest signature amongst functions in a given factor graph. As with the general class of variational methods, though, specific domains can be amenable to closed-form representations that seek to reduce the runtime or reduce the degree of approximation for the general derivation shown herein [4].

4 Relation to (Loopy) BP and Other Methods

When written out within the same notational framework as used here, the regular Belief Propagation algorithm produces almost the same update rule as Equation (7)![1] The only difference is the outer summation over factors that neighbor a given variable: changing this summation into a product produces regular BP. In EMBP, the summation results naturally from taking logarithms and applying Jensen's Inequality to get guaranteed convergence [3]. This produces a softer, arithmetic average over the weights that a variable node receives from its neighboring factors. On the other hand, the harsher, geometric average of BP embodies its inherent non-convergence. If both approaches are viewed as coordinate ascent in likelihood over the space of bias distributions [11, 15], it is possible to demonstrate that EMBP takes smaller steps that are guaranteed never to increase the distance to the nearest local maximum, while BP can take larger ones that overshoot maxima and produce orbiting non-convergence.

The otherwise close correspondence between EMBP and BP makes sense in light of how EM and BP are both based on the minimization of "Variational" or "Gibbs" Free Energy [11, 15]. Each seeks to minimize the KL-Divergence between two distributions; one goal of deriving EMBP from first principles is to demonstrate the primal/dual nature of this process. In BP it is clear that variables weight themselves by summarizing reports from surrounding factors. The EM framework emphasizes that such reports should themselves be interpreted as summaries over constraint extensions, and provides a different way of combining them that produces convergence. As already discussed toward the beginning of Section 3, such weighting of constraint extensions in the dual problem can be viewed in terms Variable Elimination, providing another perspective on how EMBP is able to handle loops.

Such similarities beg the question of how EMBP differs from EM itself. At the lowest level, it cannot be different at all, as it was derived entirely within the EM framework. Rather, the significance of the primal/dual derivation lies in its isolation of the relationship between parameter estimation and approximately inferring marginals, and the similarity between the resulting rule and regular BP–which was designed with no such relationship in mind. That being said, there are a number of other methods that can also address various shortcomings of BP. Expectation Propagation (EP) [10] and related "moment-matching" methods assume a similar perspective to EMBP's in seeking to summarize the behavior of factors as though they were variables, but such methods do not guarantee convergence. Other research characterizes message-passing algorithms in terms of optimization across the "marginal polytope" of valid marginal distributions for a given graph structure [13, 14]. Together with novel approximations to the entropy term of the free energy underlying such an optimization, alternative ap-

[1] Usually BP is written as two sets of rules, but in the context of sequential updates there is a natural way to combine them into a single rule since estimated marginals for a variable node can be calculated at any point in the algorithm by taking the product of incoming messages [9].

proximations to the marginal polytope can produce new methods that also converge, but that minimize different energy functions than BP and EMBP. Finally, Generalized Belief Propagation (GBP) [15] takes a similar tack to EMBP's by replacing pairwise (Bethe) free energy with more general Kikuchi representations. In the context of variable elimination this can be seen as a soft "clustering" of the nodes in a graph.

5 Conclusions and Future Work

EMBP is a variant of BP that guarantees convergence even on graphs with loops. First developed in the context of Constraint Satisfaction, it has demonstrated nearly-universal empirical superiority to BP within this specific domain of evaluation [6, 5, 7]. However, assessing its general usefulness not only requires further empirical study on a wider variety of inference tasks, but it also motivates a better theoretical characterization of EMBP's underlying dynamics.

For one thing, despite searching the same landscape, BP and EMBP can wind up at different local maxima in likelihood when starting with the same initialization, due to BP's larger steps. So while EMBP is guaranteed to converge, BP may explore a larger portion of the search space. Thus the two may be attracted to different sorts of maxima, and formally representing this difference will improve our understanding of the techniques and their performance on different domains. A second area of future research is to better understand the approximations used in the derived version of EMBP–to what extent are they also inherent in regular BP and other variants? As with many probabilistic methods, applying EMBP to a specific domain can instigate a series of re-formulations that yield improved efficiency or accuracy, or that trade one for the other.

Here, the intended contribution of deriving EMBP from first principles has been to generalize from its original domain-specific form and thus elucidate the primal/dual insights that underly its guaranteed convergence. In doing so, the ultimate aim is to invite interest from the wider Machine Learning community in the resulting issues that are mentioned here, and also in those that are not.

References

[1] A. Braunstein, M. Mezard, and R. Zecchina. Survey propagation: An algorithm for satisfiability. *Random Structures and Algorithms*, 27:201–226, 2005.

[2] R. Dechter, K. Kask, and R. Mateescu. Iterative join-graph propagation. In *Proc. of 18th Int'l Conference on Uncertainty in A.I. (UAI '02), Edmonton, Canada*, pages 128–136, 2002.

[3] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–39, 1977.

[4] E. Hsu. http://www.cs.toronto.edu/˜eihsu/varsat/. In *VARSAT Homepage*, 2008.

[5] E. Hsu, M. Kitching, F. Bacchus, and S. McIlraith. Using EM to find likely assignments for solving CSP's. In *Proc. of 22nd Nat'l Conf. on AI (AAAI '07), Vancouver, Canada*, 2007.

[6] E. Hsu and S. McIlraith. Characterizing propagation methods for boolean satisfiability. In *Proc. of 9th Int'l Conf. on Theory and Applications of SAT (SAT '06), Seattle, WA*, 2006.

[7] E. Hsu, C. Muise, J. C. Beck, and S. McIlraith. From backbone to bias: Probabilistic inference and heuristic search performance. In *Proc. of 1st Int'l Symposium on Search Techniques for AI and Robotics (STAIR '08), Chicago, Illinois*, 2008.

[8] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. In M. Jordan, editor, *Learning in Graphical Models*. MIT Press, 1998.

[9] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2), 2001.

[10] T. Minka. A family of approximate algorithms for Bayesian inference. In *Ph.D. Thesis*, 2001.

[11] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[12] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[13] D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *22nd Conf. on Neural Information Processing Systems (NIPS '07), Vancouver, Canada*, 2007.

[14] M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *[IEEE Transactions on Information Theory*, 51:2313–35, 2005.

[15] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In B. Nebel and G. Lakemeyer, editors, *Exploring Artificial Intelligence in the New Millennium*, pages 239–256. Morgan Kaufmann, 2003.