Locally Conditioned Belief Propagation

Thomas Geier and **Felix Richter** and **Susanne Biundo**
Institute of Artificial Intelligence
Ulm University, Germany
{thomas.geier, felix.richter, susanne.biundo}@uni-ulm.de

Abstract

Conditioned Belief Propagation (CBP) is an algorithm for approximate inference in probabilistic graphical models. It works by conditioning on a subset of variables and solving the remainder using loopy Belief Propagation. Unfortunately, CBP's runtime scales exponentially in the number of conditioned variables. Locally Conditioned Belief Propagation (LCBP) approximates the results of CBP by treating conditions locally, and in this way avoids the exponential blow-up. We formulate LCBP as a variational optimization problem and derive a set of update equations that can be used to solve it. We show empirically that LCBP delivers results that are close to those obtained from CBP, while the computational cost scales favorably with problem size.

1 INTRODUCTION

Modern SAT solvers are capable of solving problem instances with hundreds of thousands of variables (Katebi et al., 2011), despite the fact that SAT is an NP-hard problem. Most of today's practical solvers are CDCL (conflict-driven, clause-learning) solvers (Marques-Silva et al., 2009). Their main algorithmic components are branching, unit propagation, and clause learning (Katebi et al., 2011). Generalizing these concepts, we could talk of branching as analysis by cases, unit propagation as inference within a single case (both already found in the classic DPLL algorithm (Davis et al., 1962)), and clause learning (Silva and Sakallah, 1996) as reusing inference results across cases.

The #P-hard (Roth, 1996) problem of computing marginal probabilities (or the partition function) in discrete-valued graphical models is closely related to #SAT—the task of counting the models of a propositional formula. Probabilistic inference generalizes the boolean conjunction of clauses to a product over local real-valued functions. The #SAT problem is usually tackled using modified CDCL solvers (Bayardo Jr and Pehoushek, 2000; Sang et al., 2004; Huang and Darwiche, 2005). Both for probabilistic inference and #SAT, it is not enough to find one satisfying case, but one has to take into consideration all cases. But while SAT problems are usually sparse, probabilistic problems can often be strictly positive. It is thus not very surprising that the basic probabilistic inference algorithms do not employ analysis by cases, but rely on inference by propagation only: Variable elimination (Koller and Friedman, 2009, Chapter 9), the Junction tree method (Shenoy and Shafer, 1990), loopy Belief Propagation (Pearl, 1986), and more generally the class of algorithms with variational interpretations (Wainwright and Jordan, 2008) can be counted towards this class.

But there are also algorithms that complement propagation with an analysis by cases, such as the exact Recursive Conditioning (Darwiche, 2001) and Value Elimination (Bacchus et al., 2002). Also there exist approximate instances: Cutset Sampling (Bidyuk and Dechter, 2007), SampleSearch (Gogate and Dechter, 2011), Conditioned Belief Propagation (Eaton and Ghahramani, 2009) and collapsed sampling algorithms in general, just to name a few. These approaches appear to have an advantage when the problem encodes a distribution that is not strictly positive, i.e., factors can evaluate to zero (we call these factors deterministic dependencies). Under the presence of deterministic dependencies, analysis by cases is able to reveal context-specific independencies, and prune the search space without incurring an approximation error.

The third algorithmic component in SAT solvers is reusing results across cases. Notably both named exact algorithms heavily rely on this concept under the name of caching. Of the named approximate inference algorithms that employ analysis by cases, only the importance sampler SampleSearch shares work across cases, and only for deterministic dependencies, by using no-good learning (Dechter, 1990). Conditioned Belief Propagation (CBP) (Eaton and Ghahramani, 2009; Geier et al., 2014a) is the straight-

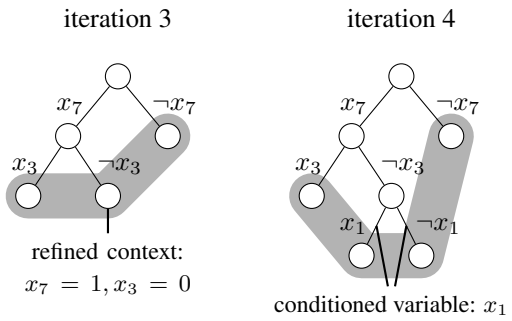


Figure 1: Iterative CBP (Geier et al., 2014a) is a divide-and-conquer algorithm that splits the problem by recursive conditioning. The state of the algorithm is defined by a tree, where edges represent assignments to variables, and nodes represent the partial assignment defined by their path from the root (or the sub-problem obtained by conditioning on this assignment). In each iteration a leaf node (case, context) is chosen and further refined by splitting on an unassigned variable. An approximation to the partition function is obtained by summing the partition function estimates found by BP on each leaf. Marginal probabilities can be obtained by forming a convex combination of the corresponding estimates for the leafs, using their estimated partition functions as weights.

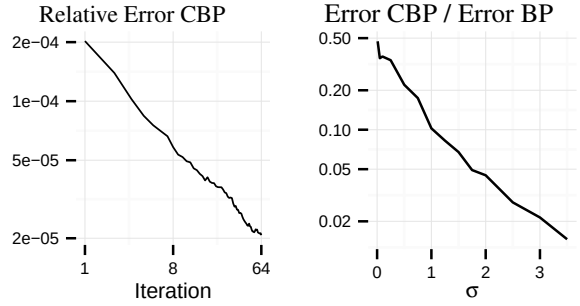


Figure 2: The plots show the typical behavior of iterative CBP on random binary-valued 8×8 grid problems (Geier et al., 2014b). The left plot shows the relative error in $\ln Z$ (median over 500 problems, factor values sampled from $\exp(\mathcal{N}(0, \sigma))$ with $\sigma = 1$), which *improves only logarithmically* with the number of distinguished cases (Iterations). The right plot shows the CBP error after 64 iterations as a fraction of the BP error for problems with varying strength of interaction (higher σ corresponds to stronger interactions, $\sigma = 0$ excluded, median over 250 problems). The approximation error of CBP compared to the error of BP consistently decreases with stronger dependencies.

forward combination of systematic analysis by cases (conditioning) with loopy Belief Propagation (BP) as approximate inference within each case. It works by recursively splitting on the assignments to single variables, producing an unbalanced and dynamically ordered tree in the process (Figure 1). This appears to be a fertile combination, as BP yields good results in weakly coupled (high entropy) models and suffers under the presence of strong dependencies (Montanari and Rizzo, 2005; Mooij and Kappen, 2007). Contrarily, conditioning provides benefits for low entropy models with strong dependencies, but fails when the probability mass is spread out evenly over a large number of similar conditions. As shown empirically by Geier et al. (2014b), CBP is indeed able to deliver good improvements over plain BP in particular for low entropy distributions (Figure 2). But the same work also highlights one major shortcoming of CBP: To sustain the same proportional improvement, the number of cases CBP has to evaluate increases exponentially with problem size (Figure 2).

In this essay we describe a method to improve the CBP algorithm in such a way that work between cases is shared approximately—thus adding the third algorithmic component found in modern SAT solvers. The basic idea focuses on the observation that the influence of conditioning on variables usually diminishes with graphical distance. We underpin this assumption empirically by visualizing the effect of conditioning a single variable in randomly generated grid problems in Figure 3. We exploit this "locality of effect" assumption in the proposed *Locally Conditioned Belief Propagation* (LCBP) model. LCBP conceptually works by merging nodes of the BP graph between different cases of CBP, thus effectively sharing message values. An intuition of the difference between CBP and LCBP is conveyed by Figure 4.

2 PRELIMINARIES

We focus on undirected graphical models over n random variables $X_1, X_2, \ldots, X_n$, referring to the set of all variables as $\mathcal{X}$. Each variable $X_i \in \mathcal{X}$ may assume values out of its finite domain $\mathrm{Dom}(X_i)$. A problem is given by a finite set Φ of non-negative local functions (factors). Each function $\phi_a \in \Phi$ is defined over the valuations $\mathrm{Val}(\mathbf{X}_a)$ (assignments of values to variables) for a subset $\mathbf{X}_a \subseteq \mathcal{X}$ of variables. The (unnormalized) product over all factors is $\tilde{p}(x) = \prod_a \phi_a(x_a)$, and it implies a proper distribution by

$$p(x) = \frac{1}{Z}\tilde{p}(x) \text{ with } Z = \sum_x \tilde{p}(x). \tag{1}$$

The normalizing constant Z is called the *partition function*.

2.1 BELIEF PROPAGATION AS OPTIMIZATION

Given a factorized distribution p, the basic problem of probabilistic inference is to compute some property of it. These properties are usually expectations, marginal probabilities, the partition function, or most probable

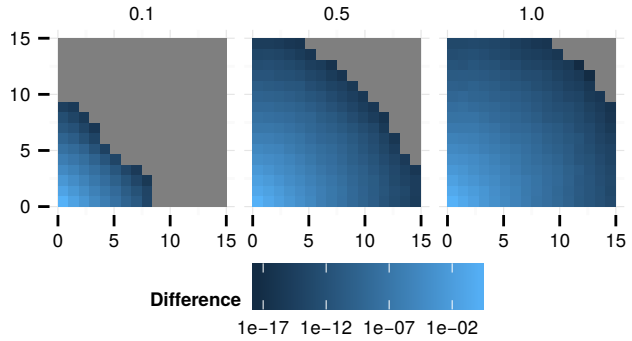


Figure 3: Comparison between two runs of BP on a 16×16 grid problem of binary-valued variables. The color encodes the difference of the marginal probabilities after conditioning the variable in the lower left (median over 100 random instances). Factor values are drawn from an exponentiated normal distribution $\exp(\mathcal{N}(0, \sigma))$ with standard deviation $\sigma \in \{0.1, 0.5, 1\}$. Gray means the difference is lower than than numerical accuracy. One can see that the effect of the conditioning is local to the conditioned variable. The range of the effect increases with stronger potentials.

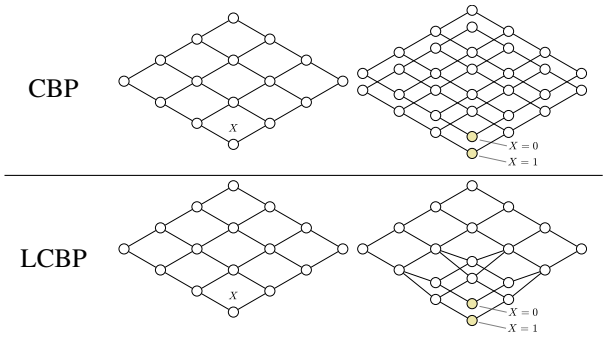


Figure 4: The upper row represents how CBP works by making full copies of the problem for each case. The lower row shows how LCBP only makes copies of the nodes that are local to the conditioned variable.

assignments—and their exact computation is often intractable (Roth, 1996). Variational inference (Wainwright and Jordan, 2008) is a form of approximate inference that works by substituting p by some element q from a class of (pseudo-) distributions $\mathcal{Q}$, on the members of which inference is tractable. The instance q is chosen to be as close to p as possible. The notion of closeness is captured by some distance measure, which is often taken to be the Kullback-Leibler divergence—though other measures are possible (Minka, 2005).

From the KL-divergence between q and p one can obtain

$$\ln Z = \mathbb{E}_q[\ln \tilde{p}] + \mathbb{H}(q) + \mathrm{KL}(q \parallel p). \tag{2}$$

Here, $\mathbb{H}(q)$ denotes the entropy of q, and $\mathbb{E}_q[f(x)]$ denotes the expectation of $f(x)$ taken with respect to the measure q. From Equation 2 we identify the functional $F(q)$, known as the negative free energy:

$$F(q) = \mathbb{E}_q[\ln \tilde{p}] + \mathbb{H}(q) \tag{3}$$

It yields the exact log-partition function if $q = p$, and can serve as a lower bound to Z if the class Q contains only valid distributions. The task in variational inference is to find a q^* that maximizes F. In general, either because there exists no exact representation of p in $\mathcal{Q}$, because the class $\mathcal{Q}$ also contains non-distribution functions, or because we cannot solve the optimization problem perfectly, the found value of $F(q)$ can only serve as an approximation for $\ln Z$. In addition, for many interesting classes $\mathcal{Q}$, the functional cannot be given in closed form and one has to resort to further approximations.

We briefly summarize how to express the BP algorithm as a variational optimization problem. A more detailed exposition can be found in Yedidia et al. (2005) and Koller and Friedman (2009, Chapter 11). For deriving the BP message update equations using the variational approach, members q of class $\mathcal{Q}_{\mathrm{BP}}$ are defined by marginal distributions $q_i(X_i)$ over the variables (called variable beliefs), and marginal distributions $q_a(\mathbf{X}_a)$ over the factors in Φ (called factor beliefs):

$$q(x) = \prod_a q_a(x_a) \prod_i q_i(x_i)^{(1-d_i)} \tag{4}$$

Here, $d_i = |\{\phi_a \in \Phi | X_i \in \mathbf{X}_a\}|$ represents the number of factors that depend on variable X_i. In addition to being proper probability measures (sum to one, non-negative), the variable and factor beliefs have to be consistent with respect to their marginal probabilities. This is formalized by requiring for all factors ϕ_a, adjacent variables $X_i \in \mathbf{X}_a$, and values $x_i \in \mathrm{Val}(X_i)$:

$$\sum_{x_a \models x_i} q_a(x_a) = q_i(x_i) \tag{5}$$

Note that we write $x_a \models x_i$ for all the (partial) assignment $x_a \in \mathrm{Val}(\mathbf{X}_a)$ that are an extension of x_i. A further ingredient in the variational derivation of standard BP exists in an approximation to the entropy, known as the Bethe-Peierls (also BP) approximation and given by

$$\mathbb{H}_{\mathrm{BP}}(q) = \sum_a \mathbb{H}(q_a) + \sum_i (1 - d_i)\mathbb{H}(q_i). \tag{6}$$

A justification for $\mathbb{H}_{\mathrm{BP}}$ is usually given by the fact that it is exact for tree-structured problems. The BP approximation together with the assumption that the functions q_a resemble marginal distributions of q over the variables in $\mathbf{X}_a$ yields the functional

$$F_{\mathrm{BP}}(q) = \sum_a \mathbb{E}_{q_a}[\ln \phi_a] + \mathbb{H}_{\mathrm{BP}}(q). \tag{7}$$

Optimizing $F_{\mathrm{BP}}(q)$ under the given constraints using the method of Lagrange multipliers yields the update equations

of the BP algorithm. Loosely speaking, the Lagrange multipliers assume the role of messages between variables and factors ($m_{i\to a}(x_i)$ and $m_{a\to i}(x_i)$), each encoding a distribution over the respective variable X_i. With abuse of notation, writing $i \in \mathbf{X}_a$ instead of $X_i \in \mathbf{X}_a$, the update equations are

$$m_{i\to a}(x_i) \propto \prod_{b:i\in\mathbf{X}_b, b\neq a} m_{b\to i}(x_i), \tag{8}$$

$$m_{a\to i}(x_i) \propto \sum_{x_a \models x_i} \phi_a(x_a) \prod_{j\in\mathbf{X}_a, j\neq i} m_{j\to a}(x_j). \tag{9}$$

The BP algorithm recomputes the message values according to those equations until convergence (which is not guaranteed). The variable beliefs can then be computed by $b_i(x_i) = \prod_{a:i\in\mathbf{X}_a} m_{a\to i}(x_i)$, and factor beliefs are given by $b_a(x_a) = \phi_a(x_a) \prod_{i\in\mathbf{X}_a} m_{i\to a}(x_i)$.

3 VARIATIONAL CBP

Before introducing the LCBP model, we want to interpret CBP in a variational way as a mixture model. For this we reduce the iterative CBP algorithm (Figure 1) to the BP inference on the induced partitioning into cases (the leafs of the tree), and ignore the way in which the partition was obtained. We call this non-iterative interpretation *variational CBP*, and use the term *iterative CBP* when we want to emphasize the recursive conditioning aspect. In variational CBP, we are given a set of partial assignments/conditions C whose extensions partition the set of all assignments $\mathrm{Val}(\mathcal{X})$. The set C corresponds to the leafs of a tree produced when running iterative CBP. A member q of class $\mathcal{Q}_{\text{CBP}}$ is then defined by

$$q(x) = \sum_{c\in C} q_C(c) \prod_a q_a^c(x_a) \prod_i q_i^c(x_i)^{1-d_i}. \tag{10}$$

It can be interpreted as a mixture of BP approximations, where $q_C(c)$ encodes the mixture weight. The necessary constraints are the BP constraints for each set of beliefs q_a^c, q_i^c. The weight vector $q_C : C \to [0,1]$ is required to be a proper distribution (non-negative, sum to one). And in addition to the BP constraints, we require $q_i^c(x_i) = 1$ for $c \models x_i$ to enforce the conditions within the mixture components. As a result of this constraint, the mixture components have mutually exclusive support. By defining an appropriate energy functional, and solving the variational problem for $\mathcal{Q}_{\text{CBP}}$, one finds that a solution can be found by solving the BP variational problem for each mixture component independently.

The computational cost of CBP is about linear in the number of conditions, as each condition implies one run of the BP algorithm. Let us assume that the number of conditioned variables has to attain a certain ratio of the total number of variables for CBP to be able to produce a good approximation. This implies that the number of distinguished conditions $|C|$ (and thus inference cost) grows exponentially with problem size when sustaining good approximation quality.

4 LCBP

LCBP is designed with the goal that its computational cost scales sub-exponentially in the number of (fully) conditioned variables. This means, we want to approximate variational CBP for an exponentially large set C, and have the computational cost scaling only polynomially with $\ln|C|$. To achieve this we have to overcome two obstacles. The first one is getting rid of the exponential number of parameters q_a^c, q_i^c present in the variational CBP approximation. Under the assumptions that BP messages do not differ much when far away from a disturbance (Figure 3), we can substitute some $q_a^{c_1}$ by $q_a^{c_2}$ in equation 10 given that factor ϕ_a is far enough from all variables where conditions c_1 and c_2 differ. The second problem is representing the weight distribution q_C. As we will observe in the sequel, this problem will be solved by representing q_C in factored form, necessitating probabilistic inference over the condition variables.

4.1 CONDITIONING SCHEME

To formalize which local functions q_a^c, q_i^c can be shared between conditions, we introduce a concept termed *conditioning scheme*. We focus on a particular form of conditioning scheme that we call factored, local scheme (FL-scheme). FL-schemes are not powerful enough to capture all aspects of iterative CBP, i.e., they emulate only balanced and statically ordered search trees. But their simple structure allows a formal derivation of the LCBP algorithm, while they are expressive enough to capture the essential improvement LCBP offers over CBP. For a discussion on lifting the restrictions implied by FL-schemes see Section 6.1.

An FL-scheme $\mathcal{S} : \mathcal{X} \to 2^{\mathcal{X}}$ assigns a set of conditioning variables (conditioners) to each variable in $\mathcal{X}$. The idea is that, locally at a variable X_i, we have a copy of the BP messages and beliefs for each assignment to the conditioners $\mathcal{S}(X_i)$ of X_i. We use the notations $S_i = \mathcal{S}(X_i)$ and $S_a = \bigcup_{i\in\mathbf{X}_a} \mathcal{S}(X_i)$ for the set of variable conditioners and factor conditioners respectively. We write $C = \bigcup_{X_i\in\mathcal{X}} \mathcal{S}(X_i)$ for the set of all conditioners. Given some variable X_c, we call the set $\{X_i \mid X_c \in S_i\}$ the area of influence of conditioner X_c or the set of X_c's conditionees. For the lower right example in Figure 4, we have $C = \{X\}$. The three variables around X, and X itself are the conditionees of X. They have only X as their conditioner, and thus are replicated for each possible value of X (0 and 1). The variable at the left-most corner is not conditioned, and thus has an empty set assigned by the scheme. Note that an FL-scheme only tells how to split variables and the associated variable

beliefs. Factor beliefs are split by assignments to the union of the conditioners of the variables in their scope S_a. They are thus always split in a more fine-grained way than the adjacent variables.

4.2 APPROXIMATING CLASS $\mathcal{Q}_{\text{LCBP}}$

Given an FL-scheme $\mathcal{S}$ for a problem Φ, we define a pseudo distribution q from class $\mathcal{Q}_{\text{LCBP}}$. The parameters are the variable beliefs $q_i^{c_i}$ and factor beliefs $q_a^{c_a}$ known from BP, but now each in multiple versions for each local variable condition $c_i \in \text{Val}(S_i)$ or local factor condition $c_a \in \text{Val}(S_a)$. In addition we require a normalized probability measure $q_C : \text{Val}(C) \to [0, 1]$ over all possible conditions. Until the end of this section we assume that q_C is represented as a flat (unstructured) function. Writing $x[A]$ for restricting the assignment $x \in \text{Val}(\mathcal{X})$ to the variables in $A \subseteq \mathcal{X}$, we define one mixture component q^c as

$$q^c(x) = \prod_a q_a^{c[S_a]}(x_a) \prod_i \left(q_i^{c[S_i]}(x_i)\right)^{1-d_i}. \tag{11}$$

We define the pseudo distribution for the LCBP model as

$$q(x) = \sum_{c \in \text{Val}(C)} q_C(c) q^c(x). \tag{12}$$

To enforce that q is close to a probabilistic measure, we formulate a set of constraints on its parameters. Non-negativity constraints are assumed implicitly.

The normalization of the conditioning distribution:

$$\sum_c q_C(c) = 1 \tag{13}$$

The normalization of factor beliefs for all $\phi_a \in \Phi, c_a \in \text{Val}(S_a)$:

$$\sum_{x_a} q_a^{c_a}(x_a) = 1 \tag{14}$$

The normalization of variable beliefs for all $X_i \in \mathcal{X}, c_i \in \text{Val}(S_i)$:

$$\sum_{x_i} q_i^{c_i}(x_i) = 1 \tag{15}$$

The marginal consistency constraints for all $\phi_a \in \Phi, X_i \in \mathbf{X}_a, c_i \in \text{Val}(S_i), x_i \in \text{Val}(X_i)$:

$$\sum_{c_a \models c_i} q_C(c_a|c_i) \sum_{x_a \models x_i} q_a^{c_a}(x_a) = q_i^{c_i}(x_i) \tag{16}$$

We enforce the condition for all $X_i \in C, c_i \in \text{Val}(S_i)$:

$$q_i^{c_i}(x_i) = 1 \tag{17}$$

Equation 16 is the LCBP version of the marginal consistency constraints of the BP approximation. It formalizes the way in which beliefs are merged between conditions by taking the expectation with respect to the distribution over conditions q_C.

4.3 FREE ENERGY APPROXIMATION

We are now going to derive a set of fixed-point equations that can be used to implement a message passing algorithm. By partitioning the set of variables into conditioners and the rest $\bar{C} = \mathcal{X} \setminus C$, applying the identity $\mathbb{H}(C, \bar{C}) = \mathbb{H}(C) + \mathbb{H}(\bar{C}|C)$ for the conditioned entropy, and using the Bethe-Peierls approximation (6), we obtain

$$\mathbb{H}_{\text{LCBP}}(q) = \mathbb{H}(q_C) + \mathbb{E}_{q_C}[\mathbb{H}_{\text{BP}}(q^c)]. \tag{18}$$

Under the assumption that the conditioned factor beliefs q_a^c are truly the marginals of q over $\mathbf{X}_a$ under given condition c, we can write the energy functional for LCBP as

$$\begin{aligned} F_{\text{LCBP}}(q) = &\sum_a \sum_{c_a} q_C(c_a) \sum_{x_a} q_a^{c_a}(x_a) \ln \phi_a(x_a) \\ &+ \mathbb{H}_{\text{LCBP}}(q). \end{aligned} \tag{19}$$

4.4 UPDATE EQUATIONS

Optimizing (19) under the constraints (13) to (17) using the method of Lagrange multipliers lets us derive the message update rules[1]. The update equations work on these additional entities:

1. $m_{i\to a}^{c_i}(x_i)$ is a message from variable i to factor a under variable condition $c_i \in \text{Val}(S_i)$.
2. $m_{a\to i}^{c_i}(x_i)$ is a message from factor a to variable i under variable condition $c_i \in \text{Val}(S_i)$.
3. $n_{a\to i}^{c_a}(x_i)$ is a message from factor a to variable i under factor condition $c_a \in \text{Val}(S_a)$.

We use the artificial factor $\rho_i^{c_i}(x_i) = \mathbf{1}[c_i[X_i] = x_i]$ to enforce condition (17) on the variable beliefs[2]. The update equations are as following:

$$m_{i\to a}^{c_i}(x_i) \propto \rho_i^{c_i}(x_i) \prod_{b: i \in \mathbf{X}_b, b \neq a} m_{b\to i}^{c_i}(x_i) \tag{20}$$

$$m_{a\to i}^{c_i}(x_i) \propto \sum_{c_a \models c_i} q_C(c_a|c_i) \cdot n_{i\to a}^{c_a}(x_i) \tag{21}$$

$$n_{a\to i}^{c_a}(x_a) \propto \sum_{x_a \models x_i} \phi_a(x_a) \prod_{j \in \mathbf{X}_a, j \neq i} m_{j\to a}^{c_a[S_j]}(x_j) \tag{22}$$

The variable and factor beliefs are computed from the messages via the following formulas:

$$q_a^{c_a}(x_a) \propto \phi_a(x_a) \prod_{i \in \mathbf{X}_a} m_{i\to a}^{c_i}(x_i) \tag{23}$$

$$q_i^{c_i}(x_i) \propto \rho_i^{c_i}(x_i) \prod_{a: i \in \mathbf{X}_a} m_{a\to i}^{c_i}(x_i) \tag{24}$$

[1] A more detailed derivation of the update equations is provided in the appendix available in the supplied materials.

[2] $\mathbf{1}[A]$ represents the indicator function that yields 1 when the condition A is true and 0 otherwise.

And the update equation for the condition distribution is

$$q_C(c) \propto \exp\left[F_{\text{BP}}(q^c)\right] \prod_a \prod_{i \in \mathbf{X}_a} \prod_{x_i} \delta_{ai}^{c_a}(x_i), \qquad (25)$$

with

$$\delta_{ai}^{c_a}(x_i) = m_{i\to a}^{c_i}(x_i)^{m_{i\to a}^{c_i}(x_i)(n_{a\to i}^{c_a}(x_i) - m_{a\to i}^{c_i}(x_i))}. \quad (26)$$

The fact that the stated update equations are suited to optimize the formulated variational problem is formalized by the following theorem.

Theorem 1. *The interior stationary points of the variational problem specified by maximizing the LCBP functional (19) under the given constraints (13) through (17) are exactly the fixed points of the LCBP update equations (20) through (25).*

The proof is given by the derivation in the appendix available in the extended version of this paper.

The term (26) (and thus the triple product in (25)) vanishes when the messages $n_{a\to i}^{c_a}$ agree with the aggregate message $m_{a\to i}^{c_i}$. According to Figure 3 this can happen when the set of conditionees is chosen to be large. Empirically we could not detect a significant difference in inference quality between calculating the $\delta_{ai}^{c_a}$ terms according to (26) or setting them to 1.

Until now we have treated the distribution over the conditions $q_C(c)$ as flat. Taking a closer look at equation (25), we notice that the right hand side is a product, with factors coming from the exponentiated BP energy and the $\delta_{ai}^{c_a}$ terms. These factors all depend on different subsets of variables from C. Thus the right side of equation (25) describes an undirected graphical model. We call it the *condition problem*, while referring to the original problem as the *primal problem*. When calibrating the message beliefs, it becomes necessary to calculate conditional probabilities $q_C(c_a|c_i)$ for this problem, and this can be done using any inference algorithm for graphical models. The graphical structure of the condition problem is determined by the overlap between the sets of conditionees for different conditioners, and exact inference in the condition problem can become intractable.

5 EMPIRICAL EVALUATION

We conducted two experiments examining the performance of LCBP on randomly generated problem instances. The first experiment is meant to demonstrate that the quality of the LCBP approximation approaches that of variational CBP when increasing the area of influence around conditioned nodes. A second experiment examines how the computational effort of LCBP scales when the problem size increases. In both experiments we condition fully on all conditioners to obtain the variational CBP approximation.

For the experiments, we have implemented two variants of LCBP using the derived update equations. The first variant (LCBP-JT) employs exact inference over the condition problem using the Junction tree method (Shenoy and Shafer, 1990). The second variant (LCBP-BP) uses BP to approximate the marginals of the condition problem. All algorithms are implemented using Round-robin message schedules with no damping. Except for plain BP, the algorithms managed to converge every time. When reporting accuracy we remove all instances where BP did not converge, and thus favor BP in our presentation. We like to remark that when running algorithms from the CBP class, the tolerance for the convergence check has to be set very low. Otherwise the numerical errors may pile up and deteriorate the result even below BP level.

For the first set of experiments, we applied LCBP-JT and CBP to 6×6 grid problems with binary variables and random interactions (Figure 5). We selected four fixed variables as conditioners. We varied both the number of conditionees and the interaction strength of the random grids. As expected, the error produced by LCBP-JT approaches the error of CBP both with decreasing interaction strength, and with growing area of influence. We can thus conclude that LCBP-JT acts as an approximation to the CBP result.

The second set of experiments is meant to examine the scaling behavior of LCBP. While the number of parameters of the variational approximation of LCBP grows more slowly than variational CBP, it is conceivable that LCBP takes significantly longer to converge (or even fails to converge at all). To be able to demonstrate that LCBP can yield good quality approximations, we designed a special problem class that we call two-layer grid model (Figure 6). Models with a similar geometry are used in image classification (Kato et al., 1996), sometimes called hierarchical Markov random fields. CBP can achieve good results for this class of problems when conditioning on the nodes of the upper layer, if the variables in the remaining problem (the lower layer) interact only weakly and can thus be approximated well by BP. We applied BP, CBP, LCBP-JT and LCBP-BP to two-layer grid problems with weak interaction on the first and second layer, and strong interactions between layers. We varied the size of the problems to examine the computational effort of the various algorithms. The chosen FL-scheme marks all second layer nodes as conditioners, with all directly connected first layer nodes as respective conditionees.

Figure 7a shows the relative error in the inferred log partition function for varying problem sizes. Notice how all examined algorithms maintain about the same approximation quality once boundary effects are overcome; starting with widths greater than 10. We can observe that all conditioning algorithms improve over the plain BP approximation. CBP performs best, followed closely by LCBP-JT. LCBP-BP produces the worst result among the condition-

ing approaches, though its result is still better than BP by about two orders of magnitude. The difference between the LCBP-JT and the LCBP-BP result was expected, since the condition problem is not acyclic—it contains strong dependencies induced by the explicit interactions between the variables in the second layer. Experiments without interactions in the second layer (not shown) put the LCBP-BP result much closer to LCBP-JT, as paths going through the lower grid induce only weak coupling in the condition problem, while experiments with stronger interactions put LCBP-BP closer to the BP results. For all examined parameter combinations the experiment produces the same qualitative result, i.e., the same order among the examined algorithms. Figure 7b shows the CPU time for the different algorithms. As expected CBP shows an exponential growth with problem size and thus the number of conditioners. In contrast, the effort for the LCBP variants grows approximately linearly with problem size. Note that the LCBP implementations underwent only moderate optimization, as we are interested only in their asymptotic behavior. Thus, one should not draw any conclusions from the concrete slopes of the curves in Figure 7b. Also note that the generated problems have fixed tree-width; both the primal problem and the condition problem. This explains the linear scaling of LCBP-JT.

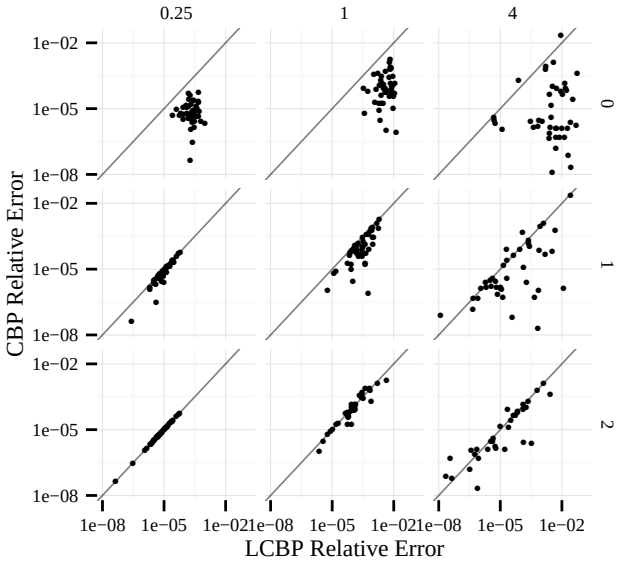


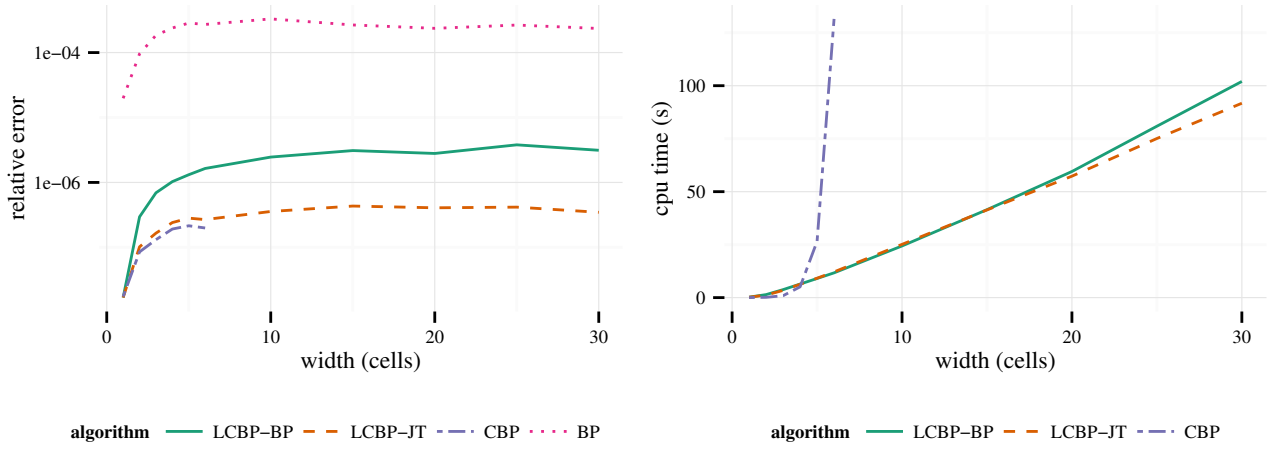
Figure 5: Evaluation results comparing LCBP-JT relative error in log partition function with CBP relative error on randomly generated binary, 6×6 grid networks. Four symmetrically placed variables are conditioned. The set of conditionees for LCBP-JT is increased along the rows, including variables with a distance of at most 0, 1 or 2 according to the max norm on the grid coordinates. The columns stand for grids with stronger potentials from left to right (sampled from $\exp(\mathcal{N}(0, \sigma))$, with $\sigma \in \{0.25, 1, 4\}$). One can see that both with growing area of influence (going down), and with weaker coupling (right to left!) the result of LCBP-JT approaches that of CBP. If both algorithms disagree, CBP yields a lower error.

6 DISCUSSION AND FUTURE WORK

The presented LCBP algorithm provides a scalable approximation to the variational interpretation of CBP. By shifting our focus from iterative CBP to variational CBP, we have lost the anytime behavior that iteratively refines the approximation over time in a heuristically guided way. The step from variational CBP to LCBP further removed the possibility of having an unbalanced and dynamically ordered tree to represent the set of examined conditions. This was necessary to achieve the factorization of the condition problem. According to our assessment, LCBP can be extended to resemble iterative CBP more closely, although we expect that this requires substantial further work. The situation is not much different for Generalized Belief Propagation (Yedidia et al., 2005), though. An iterative and heuristically guided construction of region graphs for GBP is as desirable as the adaptive construction of conditioning schemes in the LCBP setting. Approaches to this problem for GBP are still rare, although some work does exist (Welling, 2004; Sibel et al., 2012). In this section we will discuss some steps that point in this direction. Note that some of the discussion is also applicable to GBP.

6.1 DESIGNING CONDITIONING SCHEMES

When looking at the iterative CBP algorithm, it is apparent that one of its main strengths is its ability to focus its work on the modes of the target distribution. The trees that can be constructed have no constraint on their shape, and they can become very deep and narrow. This is also an advantage of iterative CBP over GBP, where the approximation is improved by using marginals over larger clusters of variables than the factor clusters q_a used in BP. Given a cluster, GBP can only improve by adding another variable to the cluster, which multiplies the computational burden associated with the cluster by the domain size of the added variable. CBP can circumvent this problem by refining single leafs of its tree; basically making context-specific refinements. LCBP faces the same problem as GBP, because of the limited expressiveness of FL-schemes. With FL-schemes the complexity of LCBP scales exponentially with the number of conditioners a conditionee has—they define the size of the factor scopes for the condition problem. For practical purposes one would aim at using more expressive schemes that allow for context-specific refinements, e.g., conditioning some variable on the conditions $\{X_1 = 0, X_2 = 0\}, \{X_1 = 0, X_2 = 1\}, \{X_1 = 1\}$, thus condition only on X_2 for $X_1 = 0$. A simple improvement for problems with large variable domains is to branch on elements of arbitrary partitions of assignments to single



(a) relative error in log partition function

(b) CPU time

Figure 7: Inference results and used CPU time for two-layer grid problems of varying size (see Figure 6). The x-axis shows the number of columns (width) of the upper grid, where a "cell" is supposed to be a group of nodes on the lower layer connected to a single node on the upper layer. The height of the upper grid is fixed to 2 to obtain problems where exact inference is tractable. All variables have binary domains. Interactions within the lower and the upper layer are weak (factor values drawn from $\exp\{\mathcal{N}(0, 0.5)\}$), while interactions between layers are strong (factor values drawn from $\exp\{\mathcal{N}(0, 4)\}$). CBP was not applied to the larger instances, due to resource constraints. All lines are means over 500 random instances.

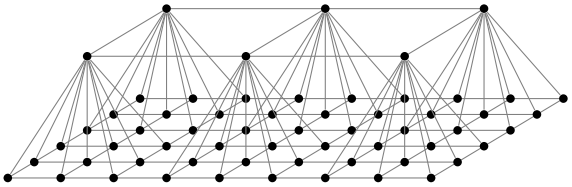


Figure 6: Illustration of the two-layer grid model used for evaluation. Each upper node is connected to 3×3 lower nodes. The upper nodes form a two-by-n grid, where n is varied to obtain problems of different size.

variables, e.g., distinguish between $X_1 < 3$ and $X_1 \geq 3$.

6.2 ITERATIVE, HEURISTIC CONSTRUCTION OF SCHEMES

The iterative CBP algorithm has the big advantage of offering an anytime approximation scheme that can be sensitive both to problem structure, and to parameters. This is achieved through the use of different types of heuristics, choosing how to refine the approximation over the course of the computation (Geier et al., 2014b).

Looking at iterative CBP, it appears natural to build the scheme for LCBP incrementally—refining the approximation after running inference and looking at the result. For CBP there exist basically two decision points: Choose the condition/leaf on which to work, then choose the variable to condition on. For LCBP with an FL-scheme choosing a branch is not possible, as this requires context-specific schemes. If those are available, then LCBP must blur both decision points of CBP into one: Choose which variable to refine under which condition—as the available conditions depend on the chosen variable. This is in contrast to CBP, where all variables are available under every condition (unless they are already conditioned). In addition, for LCBP there exists the new choice of extending the set of conditionees of a conditioner. For this decision we can think of promising candidates for evaluation, like the disagreement among the aggregated messages of the sub-conditions. Using this heuristic would result in splitting variables on the condition until the effect of conditioning has fallen below some threshold (remember Figure 3). Clearly this requirement is too strong, as for tree-structured problems the messages under different conditions can be combined at any moment while still obtaining an exact result.

To find truly informed heuristics, we have to look at the source of the error within the BP approximations. A promising way to construct heuristics for iteratively refining LCBP appears to be exploitation of the loop series expansion (Montanari and Rizzo, 2005). It specifies a correction for the BP functional (Equation 4), that allows to reconstruct the exact value of the partition function. This is done by adding a term for each generalized loop of the graph. Since error contribution is associated with loops, it is not focused on variables, but decentralized. By conditioning on one variable of a loop, while placing the complete loop in the area of influence, the loop can be cor-

rected. In this way the loop series expansion could provide guidance on choosing both conditioners *and* conditionees consistently in an error-oriented manner.

6.3 THE CONDITION PROBLEM

One nice aspect about FL-schemes is their property to induce ordinary Markov networks as condition problem. As demonstrated in the evaluation, one can use any algorithm that computes (conditional) marginal probabilities for Markov networks to solve the condition problem. This choice can be influenced by the expected characteristics of the condition problem. If the primal problem contains deterministic dependencies, it is conceivable to "presolve" the condition problem. When inference during presolving assigns zero probability to some marginal assignments, the corresponding elements of the LCBP calculation can be safely pruned. In addition, elements with very low marginal probability can be pruned on a heuristic basis, incurring a further approximation of the final result.

If a message passing algorithm is chosen for inference within the condition problem, it becomes possible to run it interleaved with the LCBP message updates. This opens the door to using more sophisticated message update schedules, for example Residual Belief Propagation (Elidan, 2006), making it possible to balance the ratio of LCBP updates against inference in the condition problem.

An interesting idea is recursively using LCBP for inference in the condition problem. A perceivable application are hierarchical grid problems with more layers. We expect this construction to scale well, meaning that nesting analysis by cases using LCBP does not incur an exponential growth in model size. It is not clear how to create such a deep hierarchical approximation using other variational techniques, such as GBP.

7 RELATED WORK

There exists some prior work on using mixture models for variational inference. Jaakkola and Jordan (1998) use mixtures of mean field approximations to improve inference quality. Beside the weaker approximation of mean field compared to BP and the locality of conditions, the main difference to LCBP is the use of mixture components with overlapping support in contrast to mutually exclusive conditions. The overlapping approach is more powerful in theory, because the mixture components are not restricted in the sense that they are clamped to an intended condition. But in contrast to this, only a weaker approximation to the entropy is used by Jaakkola and Jordan (1998) as the mutually exclusiveness allows for better analytical treatment. Split Variational Inference (Bouchard and Zoeter, 2009) is another application of conditioning and the variational method applied to arbitrary integrals.

The "Gates" model (Minka and Winn, 2008) is also intended as a variational treatment of local mixture components, and arrives at similar update equations for expectation propagation and variational message passing. The LCBP model can be described using Gates with the conditioners being the selector variables, and the conditionees (and incident factors) being placed inside the gate. The LCBP derivation is more explicitly cast as a variational problem by specifying the variational distribution and the constraints, and, more importantly, it allows overlap between gates, which Minka and Winn explicitly forbid. One could say that FL-schemes are more expressive then the (implicit) schemes allowed by Minka and Winn.

8 CONCLUSION

We have formulated a variational interpretation of CBP as a mixture of BP approximations. Based on this, we have derived LCBP, which yields inference results that approximate those obtained from CBP. We have shown empirical evidence that supports the claims that LCBP approximates CBP, while scaling much more favorably with problem size.

LCBP allows a non-trivial integration between an arbitrary probabilistic inference algorithm used for solving the condition problem and BP used for inference over the remainder. The automatic construction of good conditioning schemes for LCBP remains an open research question. But we were able to construct schemes for a motivated problem class resembling hierarchical Markov random fields, which are used in image recognition. We are currently working on formulating more expressive classes of conditioning schemes, together with an informed heuristic based on the loop series expansion for BP. We also plan to investigate the relationship between LCBP and GBP more closely.

Acknowledgements

This work was done within the Transregional Collaborative Research Centre SFB/TRR 62 "Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG).

References

Bacchus, F., S. Dalmao, and T. Pitassi (2002). Value elimination: Bayesian inference via backtracking search. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pp. 20–28.

Bayardo Jr, R. J. and J. D. Pehoushek (2000). Counting models using connected components. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pp. 157–162.

Bidyuk, B. and R. Dechter (2007). Cutset sampling for

Bayesian networks. *Journal of Artificial Intellenge Research 28*, 1–48.

Bouchard, G. and O. Zoeter (2009). Split variational inference. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 57–64. ACM.

Darwiche, A. (2001). Recursive conditioning. *Artificial Intelligence 126*(1), 5–41.

Davis, M., G. Logemann, and D. Loveland (1962). A machine program for theorem-proving. *Communications of the ACM 5*(7), 394–397.

Dechter, R. (1990). Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition. *Artificial Intelligence 41*(3), 273–312.

Eaton, F. and Z. Ghahramani (2009). Choosing a variable to clamp: Approximate inference using conditioned belief propagation. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, Volume 5, pp. 145–152.

Elidan, G. (2006). Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*.

Geier, T., F. Richter, and S. Biundo (2014a). Conditioned belief propagation revisited. In *Proceedings of the 21st European Conference on Artificial Intelligence*, pp. 1011–1012.

Geier, T., F. Richter, and S. Biundo (2014b). Conditioned belief propagation revisited: Extended version. Technical Report UIB 2014-03, Ulm University.

Gogate, V. and R. Dechter (2011). SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence 175*(2), 694–729.

Huang, J. and A. Darwiche (2005). DPLL with a trace: From SAT to knowledge compilation. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Volume 5, pp. 156–162.

Jaakkola, T. S. and M. I. Jordan (1998). Improving the mean field approximation via the use of mixture distributions. In M. Jordan (Ed.), *Learning in Graphical Models*, Volume 89, pp. 163–173. Springer.

Katebi, H., K. A. Sakallah, and J. P. Marques-Silva (2011). Empirical study of the anatomy of modern SAT solvers. In *Theory and Applications of Satisfiability Testing*, pp. 343–356. Springer.

Kato, Z., M. Berthod, and J. Zerubia (1996). A hierarchical Markov random field model and multitemperature annealing for parallel image classification. *Graphical models and image processing 58*(1), 18–37.

Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Marques-Silva, J., I. Lynce, and S. Malik (2009). *Handbook of satisfiability*, Chapter CDCL Solvers, pp. 131 – 150. IOS Press.

Minka, T. (2005). Divergence measures and message passing. Technical report, Microsoft Research.

Minka, T. and J. Winn (2008). Gates. In *Advances in Neural Information Processing Systems*, pp. 1073–1080.

Montanari, A. and T. Rizzo (2005). How to compute loop corrections to the Bethe approximation. *Journal of Statistical Mechanics: Theory and Experiment 2005*(10), 10011.

Mooij, J. M. and H. J. Kappen (2007). Sufficient conditions for convergence of the sum–product algorithm. *IEEE Transactions on Information Theory 53*(12), 4422–4437.

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence 29*(3), 241–288.

Roth, D. (1996). On the hardness of approximate reasoning. *Artificial Intelligence 82*(1), 273–302.

Sang, T., F. Bacchus, P. Beame, H. A. Kautz, and T. Pitassi (2004). Combining component caching and clause learning for effective model counting. In *Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing*.

Shenoy, P. P. and G. Shafer (1990). Axioms for probability and belief-function proagation. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pp. 169–198.

Sibel, J.-C., S. Reynal, and D. Declercq (2012). A novel region graph construction based on trapping sets for the generalized belief propagation. In *International Conference on Communication Systems (ICCS)*, pp. 305–309. IEEE.

Silva, J. P. M. and K. A. Sakallah (1996). GRASP—a new search algorithm for satisfiability. In *Proceedings of the International Conference on Computer-Aided Design*, pp. 220–227. IEEE.

Wainwright, M. J. and M. I. Jordan (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning 1*(1-2), 1–305.

Welling, M. (2004). On the choice of regions for generalized belief propagation. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 585–592.

Yedidia, J. S., W. T. Freeman, and Y. Weiss (2005). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory 51*(7), 2282–2312.