arXiv
Manuscript No.

Improving Probabilistic Inference in Graphical Models with Determinism and Cycles

Mohamed-Hamza Ibrahim · Christopher Pal · Gilles Pesant

Abstract Many important real-world applications of machine learning, statistical physics, constraint programming and information theory can be formulated using graphical models that involve determinism and cycles. Accurate and efficient inference and training of such graphical models remains a key challenge. Markov Logic Networks (MLNs) have recently emerged as a popular framework for expressing a number of problems which exhibit these properties. While loopy belief propagation (LBP) can be an effective solution in some cases; unfortunately, when both determinism and cycles are present, LBP frequently fails to converge or converges to inaccurate results. As such, sampling based algorithms have been found to be more effective and are more popular for general inference tasks in MLNs.

In this paper, we introduce Generalized arc-consistency Expectation Maximization Message-Passing (GEM-MP), a novel message-passing approach to inference in an extended factor graph that combines constraint programming techniques with variational methods. We focus our experiments on Markov logic and Ising models but the method is applicable to graphical models in general. In contrast to LBP, GEM-MP formulates the Message-Passing structure as steps of variational expectation maximization. Moreover, in the algorithm we leverage the local structures in the factor graph by using generalized arc consistency when performing a variational mean-field approximation. Thus each such update increases a lower bound on the model evidence. Our experiments on Ising grids, entity resolution and link prediction problems demonstrate the accuracy and convergence of GEM-MP over existing state-of-the-art inference algorithms such as MC-SAT, LBP, and

Mohamed-Hamza Ibrahim
Department of Computer and Software Engineering,
École Polytechnique de Montréal, Montréal, Québec, Canada.
E-mail: mohamed.ibrahim@polymtl.ca

Christopher Pal
Department of Computer and Software Engineering,
École Polytechnique de Montréal, Montréal, Québec, Canada.
E-mail: christopher.pal@polymtl.ca

Gilles Pesant
Department of Computer and Software Engineering,
École Polytechnique de Montréal, Montréal, Québec, Canada.
E-mail: gilles.pesant@polymtl.ca

Gibbs sampling, as well as convergent message passing algorithms such as the Concave-Convex Procedure (CCCP), Residual BP, and the L2-Convex method.

Keywords Markov Logic · Message Passing · Constraint Propagation · Statistical Relational Learning · Expectation Maximization

1 Introduction

Graphical models that involve cycles and determinism are applicable to a growing number of applications in different research communities, including machine learning, statistical physics, constraint programming, information theory, bioinformatics, and other sub-disciplines of artificial intelligence. Accurate and efficient inference within such graphical models is thus an important issue that impacts a wide number of communities. Inspired by the substantial impact of statistical relational learning (SRL) (Getoor and Taskar, 2007), Markov logic (Richardson and Domingos, 2006; Singla, 2012) is a powerful formalism for graphical models that has made significant progress towards the goal of combining the powers of both first-order logic (Flach, 2010) and probability. However, probabilistic inference represents a major bottleneck and can be problematic for learning when using it as a subroutine.

Loopy belief propagation (LBP) is a commonly used message-passing algorithm for performing approximate inference in graphical models in general, including models instantiated by an underlying Markov Logic. However, LBP often exhibits erratic behavior in practice. In particular, it is still not well understood when LBP will provide good approximations in the presence of cycles and when models possess both probabilistic and deterministic dependencies. Therefore, the development of more accurate and stable message passing based inference methods is of great theoretical and practical interest. Perhaps surprisingly, belief propagation achieves good results for coding theory problems with loopy graphs (Mceliece et al., 1998; Frey and MacKay, 1998). In other applications, however, LBP often leads to convergence problems. In general LBP therefore has the following limitation:

(Limitation 1) *In the presence of cycles, LBP is not guaranteed to converge.*

It is known that the local optima of the Bethe free energy correspond to local minima of LBP, and it has been proven that violating the uniqueness condition for the Bethe free energy generates several local minima (i.e., fixed points) in the space of LBP's marginal distributions (Heskes, 2004; Yedidia et al., 2005). From a variational perspective, it is known that if a factor graph has more than one cycle, then the convexity of the Bethe free energy is violated. A graph involving a single cycle has a unique local minimum and usually guarantees the convergence of LBP (see Heskes, 2004). From the viewpoint of a local search, LBP performs a gradient-descent/ascent search over the marginal space, endeavoring to converge to a local optimum (see Heskes, 2002). Heskes viewpoint is that the problem of non-convergence is related to the fact that LBP updates the unnormalized marginal of each variable by computing a coarse geometric average of the incoming messages received from its neighboring factors (see Heskes, 2002). Under Heskes' line of analysis, LBP can make large moves in the space of the marginals and therefore it becomes more likely to overshoot the nearest local optimum. This produces

an orbiting effect and increases the possibility of non-convergence. Other lines of analysis are based on the fact that messages in LBP may circulate around the cycles, which can lead to local evidence being counted multiple times (see Pearl, 1988). This, in turn, can aggravate the possibility of non-convergence. In practice, non-convergence occasionally appears as oscillatory behavior when updating the marginals (Koller and Friedman, 2009).
Determinism plays a substantial role in reducing the effectiveness of LBP (Heskes, 2004). For example, hard clauses in a Markov logic lead to deterministic dependencies in the corresponding factor graphs for groundings and therefore are particularly challenging for inference with LBP. It has been observed empirically that carrying out LBP on cyclic graphical models with determinism is more likely to result in a two-fold problem of non-convergence or incorrectness of the results (Mooij and Kappen, 2005; Koller and Friedman, 2009; Potetz, 2007; Yedidia et al., 2005; Roosta et al., 2008). A second limitation of LBP could thus be formulated as:

(Limitation 2) *In the presence of determinism (a.k.a. hard clauses), LBP may deteriorate to inaccurate results.*

In its basic form LBP also does not leverage the local structures of factors, handling them as black boxes. Using Markov logic as a concrete example, LBP often does not take into consideration the logical structures of the underlying clauses that define factors (Gogate and Domingos, 2011). Thus, if some of these clauses are deterministic (e.g., hard clauses) or have extreme skewed probabilities, then LBP will be unable to reconcile the clauses. This, in turn, impedes the smoothing out of differences between the messages. The problem is particularly acute for those messages that pass through hard clauses which fall inside dense cycles. This can drastically elevate oscillations, making it difficult to converge to accurate results, and leading to the instability of the algorithm with respect to finding a local minimum (see pages 413-429 of Koller and Friedman, 2009, for more details). On the flip side of this issue Koller and Friedman point out that one can prove that if the factors in a graph are less extreme - such that the skew of the network is sufficiently bounded - it can give rise to a contraction property that guarantees convergence (Koller and Friedman, 2009). In our work here we are interested in taking advantage of determinism when it exists in the factors of an underlying graph in a way that does not increase the threat of non-convergence.

The literature available on LBP - which is perhaps the most widely used form of message-passing based inference - is heavily influenced by ideas from machine learning (ML) and constraint satisfaction (CS) among others. Although LBP has been scrutinized both theoretically and practically in various ways, most of the existing research either avoids the limitation of determinism when handling cycles, or does not take into consideration the limitation of cycles when handling determinism.

It is well known that techniques such as the junction tree algorithm (Lauritzen and Spiegelhalter, 1988) are able to transform a graphical model into larger clusters of variables such that the clusters satisfy the running intersection property and that such a structure can then be used to obtain exact inference results. Such results also hold when the underlying graphical models possess deterministic dependencies. For many problems however, the tree width of the resulting junction tree may be so large that inference becomes intractable. More recent work has explored the

interesting question of how to construct thin junction trees (Bach and Jordan, 2001). However, many graphical models derived from a Markov logic or problems with complex constraints quickly lead to trees with large tree widths.

In this paper, one of our key objectives is to bring probabilistic Artificial Intelligence, Machine Learning and Constraint Programming techniques closer together through the lens of variational message-passing inference. That is, to address the limitations of LBP discussed above, we introduce **G**eneralized arc-consistency **E**xpectation-**M**aximization **M**essage-**P**assing (GEM-MP), a novel approach to inference for graphical models based on variational message-passing and arc-consistency within extended factor graphs. In this work we have focused on Markov logic and Ising models, but our GEM-MP framework is applicable to other representations, including standard graphical models defined in terms of factor graphs. We achieve this by first re-parameterizing the factor graph in such a way that the inference task of computing the probability of unobserved variables given observed variables can be formulated as a variational message passing procedure using auxiliary variables in the extended and re-parameterized factor graph. We then take advantage of the fact that procedures such as variational inference and EM can be viewed in terms of free energy minimization equations. We formulate our Message-Passing approach as the E and M steps of a variational message passing technique reminiscent of classical variational EM procedures (Beal and Ghahramani, 2003; Neal and Hinton, 1999). This variational formulation leads to the synthesis of new rules that update an approximation to the joint conditional distribution, minimizing the Kullback-Leibler (KL) divergence in a way that also maximizes a lower bound on the true model evidence. Since the procedure monotonically decreases the KL divergence it alleviates Limitation 1 and leads to convergence of the lower bound and KL divergence to a fixed quantity. Furthermore, we exploit the logical structures within factors by applying a generalized arc-consistency concept (see Rossi et al., 2006), and to use that to perform a variational mean-field approximation when updating the marginals. Since the procedure is cast within a variational framework, variational bounds apply which can ensure the algorithm converges to a local minimum in terms of the KL divergence.

We have organized the rest of the paper in the following manner. In Section 2, we review some key basic concepts in further detail including: Markov Logic, LBP, constraint propagation techniques, Variational Bounds, Expectation maximization (EM) and KL Divergences. In Section 3, we demonstrate the framework of GEM-MP variational inference. In Section 4 we then derive GEM-MP's general update rule for Markov logic. In section 5, we generalize GEM-MP's update rules to be applicable for MRFs. In Section 6, we conduct a thorough experimental study. This is followed by a discussion in Section 7. In Section 8 we examine related work. Finally, in Section 9, we present our conclusions and discuss directions for future research. The Appendix contains the proofs of all propositions used in the paper.

2 Preliminaries

To set the stage for our work here in this section we provide a more detailed discussion of: Markov logic; belief propagation; constraint satisfaction problems, constraint propagation and generalized arch consistency; and variational meth-

ods. We begin by reviewing Markov logic using a concrete explanatory example presented in Table 1. This example is an excerpt of the knowledge base for the Cora dataset. That is, suppose that we are given a citation database in which each citation has author, title, and venue fields. We need to know which pairs of citations refer to the same citation and the same authors (i.e. both the *SameBib* and *SameAuthor* relations are unknown). For simplicity, our objective will be to predict the *SameBib* ground atoms' marginals. At this point, let us first express our basic notation.

Table 1 An excerpt of Markov logic for the Cora dataset. The atoms SameBib and SameAuthor are unknown. Ar() is an abbreviation for atom Author(), SAr() for SameAuthor(), and SBib() for SameBib(). a_1, a_2 define authors and r_1, r_2, r_3 define citations.

Rule	First-order Logic	Clausal Form	W
Regularity	$\forall a_1, a_2, \forall r_1, r_2,\ \mathrm{Ar}(r_1, a_1) \wedge \mathrm{Ar}(r_2, a_2) \wedge \mathrm{SAr}(a_1, a_2) \Rightarrow \mathrm{SBib}(r_1, r_2)$	$\neg\mathrm{Ar}(r_1, a_1) \vee \neg\mathrm{Ar}(r_2, a_2) \vee \neg\mathrm{SAr}(a_1, a_2) \vee \mathrm{SBib}(r_1, r_2)$	1.1
Transitivity	$\forall r_1, r_2, r_3\ \mathrm{SBib}(r_1, r_2) \wedge \mathrm{SBib}(r_2, r_3) \Rightarrow \mathrm{SBib}(r_1, r_3)$	$\neg\mathrm{SBib}(r_1, r_2) \vee \neg\mathrm{SBib}(r_2, r_3) \vee \mathrm{SBib}(r_1, r_3)$	∞

Notation. A first-order knowledge base (KB) is a set of formulas in first-order logic. Traditionally, as shown in Table 1, it is convenient to convert formulas to clausal form (CNF). After propositional grounding, we get a formula $\mathcal{F}$, which is a conjunction of m ground clauses. We use $f \in \mathcal{F}$ to denote a ground clause which is a disjunction of literals built from $\mathcal{X}$, where $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ is a set of n Boolean random variables representing ground atoms. The set $\mathcal{X}_f$ corresponds to the variables appearing in the scope of a ground clause f. Both "+" and "−" will be used to denote the positive (true) and negative (false) appearance of the ground atoms. We use Y_i as a subset of satisfying (or valid) entries of ground clause f_i, and $y_k \in Y_i,\ k \in \{1, .., |Y_i|\}$ denotes each valid entry in Y_i, where the local entry of a factor is valid if it has non-zero probability. We use f_i^s (resp. f_i^h) to indicate that the clause f_i is soft (resp. hard); the soft and the hard clauses are included in the two sets $\mathcal{F}^s$ and $\mathcal{F}^h$ respectively. The sets $\mathcal{F}_{X_j+}$ and $\mathcal{F}_{X_j-}$ include the clauses that contain positive and negative literals for ground atom X_j, respectively. Thus $\mathcal{F}_{X_j} = \mathcal{F}_{X_j+} \cup \mathcal{F}_{X_j-}$ denotes the whole of X_j's clauses, and its cardinality as $|\mathcal{F}_{X_j}|$. For each ground atom X_j, we use $\beta_{X_j} = \left[\beta^+_{X_j}, \beta^-_{X_j}\right]$ to denote its positive and negative marginal probabilities, respectively.

Markov Logic (Richardson and Domingos, 2006) is a set of first-order logic formulas (or CNF clauses), each of which is associated with a numerical weight w. Larger weights w reflect stronger dependencies, and thereby *deterministic dependencies* have the largest weight ($w \to \infty$), in the sense that they must be satisfied. We say that a clause has deterministic dependency if at least one of its entries has zero probability.

The power of Markov logic appears in its ability to bridge the gap between logic and probability theory. Thus it has become one of the preferred probabilistic graphical models for representing both probabilistic and deterministic knowledge,

with deterministic dependencies (for short we say determinism) represented as *hard clauses*, and probabilistic ones represented as *soft clauses*.

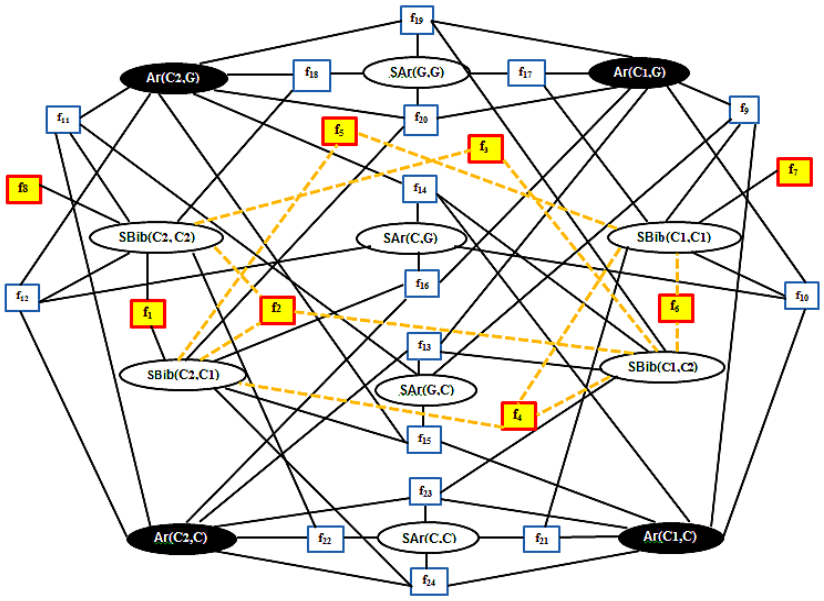


Fig. 1 Grounded (factor graph) obtained by applying clauses in Table 1 to the constants: {Gilles(G), Chris(C)} for a_1 and a_2; {Citation1(C_1) and Citation2(C_2)} for r_1, r_2, and r_3. The factor graph involves: 12 ground atoms in which 4 are evidence (dark ovals) and 8 are non-evidence (non-dark ovals); 24 ground clauses wherein 8 are hard ($\mathcal{F}^h = \{f_1, \cdots, f_8\}$) and 16 are soft ($\mathcal{F}^s = \{f_9, \cdots, f_{24}\}$).

To understand the semantics of Markov logic, recall the explanatory example in Table 1. In this example, Markov logic enables us to model the KB by using rules such as the following: 1. Regularity rules of the type that say "if the authors are the same, then their records are the same." This rule is helpful but innately uncertain (i.e., it is not true in all cases). Markov logic considers this rule as soft and attaches it to a weight (say, 1.1); 2. Transitivity rules that state "If one citation is identical to two other citations, then these two other citations are identical too." These types of rules are important for handling non-unique names of citations. Therefore, we suppose that Markov logic considers these rules as hard and assigns them an infinite weight.[1]

Subsequently, we will represent Markov logic as a factor graph after grounding it using a small set of typed constants (say, for example, $a_1, a_2 \in \{\text{Gilles(G)}, \text{Chris(C)}\}$, and $r_1, r_2, r_3 \in \{\text{Citation1}(C_1), \text{Citation2}(C_2)\}$). The output is a factor graph that is shown in Figure 1, which is a bipartite graph $\left(\mathcal{X}, \mathcal{F} = \left\{\mathcal{F}^h, \mathcal{F}^s\right\}\right)$ that has a variable node (oval) for each ground atom $X_j \in \mathcal{X}$ (here $\mathcal{X}$ includes the ground atoms: $\text{SBib}(C_1, C_1)$, $\text{SBib}(C_2, C_1)$, $\text{SBib}(C_2, C_2)$, $\text{SBib}(C_1, C_2)$, $\text{Ar}(C_1, G)$, $\text{Ar}(C_2, G)$,

[1] In practice, the transitivity rules are assigned very high weights, which complicates the inference.

$\mathrm{Ar}(C_1, C)$, $\mathrm{Ar}(C_2, C)$, $\mathrm{SAr}(C, C)$, $\mathrm{SAr}(C, G)$, $\mathrm{SAr}(G, C)$, and $\mathrm{SAr}(G, G)$). If the truth value of the ground atom is known from the evidence database, we mark it as evidence (dark ovals). It also involves a factor node for each hard ground clause $f_i^h \in \mathcal{F}^h$ (bold red square) and each soft ground clause $f_i^s \in \mathcal{F}^s$ (non-bold blue square), with an edge linking node X_j to factor f_i, if f_i involves X_j. This factor graph compactly represents the joint distribution over $\mathcal{X}$ as:

$$P(X_1, \cdots, X_n) = \frac{1}{\lambda} \prod_{i=1}^{|\mathcal{F}^h|} f_i^h(\mathcal{X}_{f_i^h}) \cdot \prod_{i=1}^{|\mathcal{F}^s|} f_i^s(\mathcal{X}_{f_i^s}), \tag{1}$$

where λ is the normalizing constant, f_i^s and f_i^h are soft and hard ground clauses respectively, and $|\mathcal{F}^h|$ and $|\mathcal{F}^s|$ are the number of hard and soft ground clauses, respectively. Note that, typically, the hard clauses are assigned the same weight ($w \to \infty$). But, without loss of accuracy, we can recast them as factors that allow $\{0, 1\}$ probabilities without recourse to infinity.

Loopy Belief Propagation. The object of the inference task is to compute the marginal probability of the non-evidence atoms (e.g., SameBib) given some others as evidence (e.g., Author). One widely used approximate inference technique is loopy belief propagation (LBP) (Yedidia et al., 2005), which provides exact marginals of query atoms conditional on evidence ones when the factor graph is a tree or a forest, and approximate marginals if the factor graph has cycles. LBP proceeds by alternating the passing of messages between variable (ground atom) nodes and their neighboring factor (ground clause) nodes. The message from a variable X_j to a factor f_i is:

$$\mu_{X_j \to f_i} = \prod_{f_k \in \mathcal{F}_{X_j} \setminus \{f_i\}} \mu_{f_k \to X_j} \tag{2}$$

The message from a factor f_i to variable X_j is:

$$\mu_{f_i \to X_j} = \sum_{X_1} .. \sum_{X_{j-1}} \sum_{X_{j+1}} .. \sum_{X_l} \left(f_i(X_1, .., X_j, .., X_l) \prod_{X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}} \mu_{X_k \to f_i} \right). \tag{3}$$

The messages are frequently initialized to 1, and the unnormalized marginal of a single variable X_j can be approximated by computing a *coarse geometrical average of its incoming messages*[2]:

$$\beta_{X_j} \propto \prod_{f_i \in \mathcal{F}_{X_j}} \mu_{f_i \to X_j}. \tag{4}$$

While there are different schedules for passing messages in graphs with loops, one of the most commonly used is synchronous scheduling, wherein all messages are simultaneously updated by using the messages from the previous iteration.

Now consider the atoms that we are interested in as a query ($\mathrm{SBib}(C_1, C_1)$, $\mathrm{SBib}(C_2, C_1)$, $\mathrm{SBib}(C_1, C_2)$, and $\mathrm{SBib}(C_2, C_2)$) on the factor graph represented in Figure 1. Remarkably, these query atoms are involved in many cycles. This

[2] Note that a geometrical average of values $\{a_i\}_{i=1}^n$, is computed as $\sqrt[n]{\prod_i a_i}$. Without $\sqrt[n]{\cdot}$, it becomes a coarse geometric average because the result would be an extreme value.

emphasizes, at least theoretically, the existence of more than one fixed point (or local optimum) which raises the threat of non-convergence (Limitation 1). In addition, six of these cycles (i.e., those represented with dashed orange lines) such as SBib(C_1, C_1) - f_5 - SBib(C_2, C_1) - f_4 - SBib(C_1, C_2) have no evidences (i.e., all the atoms in the cycles are queries). Therefore, the double counting problem is expected to happen (Limitation 1). Moreover the six cycles contain only hard clauses, which hinders the process of smoothing out the messages to converge to accurate results (Limitation 2).

Constraint Propagation. A Constraint Satisfaction Problem (Rossi et al., 2006) is a triple $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$ where $\mathcal{X}$ is an n-tuple of *variables* $\mathcal{X} = \langle X_1, \ldots, X_n \rangle$, $\mathcal{D}$ is a corresponding n-tuple of *domains* $\mathcal{D} = \langle D_1, \ldots, D_n \rangle$ such that $X_j \in D_j$, and $\mathcal{C}$ is a m-tuple of *constraints* $\mathcal{C} = \langle c_1, \ldots, c_m \rangle$. A constraint c_i is a pair $\langle \mathcal{R}_{\mathcal{X}_{c_i}}, \mathcal{X}_{c_i} \rangle$ where $\mathcal{R}_{\mathcal{X}_{c_i}}$ is a relation on the variables $\mathcal{X}_{c_i} = \text{scope}(c_i)$. A solution to the CSP is a complete assignment (or a possible world) $s = \langle v_1, \ldots, v_n \rangle$ where $v_j \in D_j$ and each $c_i \in \mathcal{C}$ is satisfied in that $\mathcal{R}_{\mathcal{X}_{c_i}}$ holds on the projection of s onto the scope $\mathcal{X}_{c_i}$. S denotes the set of solutions to the CSP. Constraint propagation (see Rossi et al., 2006) is the process of removing inconsistent values in the domains that violate some constraint in $\mathcal{C}$. One form of constraint propagation is to apply generalized arc consistency for each constraint $c \in \mathcal{C}$ until a fixed point is reached.

Definition 1 (Generalized arc Consistency (GAC)) Given a constraint $c \in \mathcal{C}$ which is defined over the subset of variables $\mathcal{X}_c$, it is *generalized arc consistent* (GAC) *iff* for each variable $X_j \in \mathcal{X}_c$ and for each value $d \in \mathcal{D}_{X_j}$ in its domain, there exists a value $d_k \in \mathcal{D}_{X_k}$ for each variable $X_k \in \mathcal{X}_c \setminus \{X_j\}$ that constitutes at least one valid tuple (or valid local entry) that satisfies c.

We can extend this CSP formalism to Weighted CSPs (see Rossi et al., 2006) to include soft constraints. This too requires extending GAC to soft generalized arc consistency (soft GAC) to tackle the soft constraints (see van Hoeve et al., 2006). At a high level, one can view GAC (or soft GAC) as a function that takes any variable $X_j \in \mathcal{X}$ and returns all other consistent variables' values that support the values of X_j with respect to the constraints $c \in \mathcal{C}$. For instance, in our example of Cora in Figure 1, applying GAC to the hard constraint (or clause) $f_6 : \neg\text{SBib}(C_1, C_1) \vee \neg\text{SBib}(C_1, C_2)$ with respect to ground atom assignment SBib$(C_1, C_1) = true$ implies maintaining only the truth value "false" in the domain of SBib(C_1, C_2). This is because the only valid local entry of f_6 that supports SBib$(C_1, C_1) = true$ is $\{(true, false)\}$.

We can also apply GAC in a probabilistic form. For instance, probabilistic arc consistency (pAC) (Horsch and Havens, 2000) performs BP in the form of arc consistency to compute the relative frequency of a variable taking on a particular value in all solutions for binary CSPs (see Horsch and Havens, 2000, for more details). pAC can be summarized as follows. We start by initializing all variables to have uniform distributions. At each step, each variable stores its previous solution probability distribution, then incoming messages from neighbouring variables are processed, and the results are maintained locally so that there is no need to send messages to all neighbours when no changes are made in the distribution. The new distribution is approximated by multiplying all information maintained from the recent message received from all neighbours. If the variable's solution distribution has changed then a new message is sent to all neighbours.

Variational Bounds, EM and KL Divergences. To derive a method with enhanced algorithmic behavior and theoretical semantics for BP, we shall be interested in a variational bound that is widely known in the context of variational expectation maximization algorithms (cf. Beal and Ghahramani, 2003; Neal and Hinton, 1999). Suppose that we have a model $\mathcal{M}_\theta$ with parameters θ, observed data $\mathcal{O} = \{O_1, \ldots, O_n\}$ and hidden variables $\mathcal{H} = \{H_1, \ldots, H_n\}$. By introducing an approximation to our distribution over hidden variables given by $q_\mathcal{H}(\mathcal{H})$, we can leverage Jensen's inequality to obtain a lower bound $\mathcal{F}_{\mathcal{M}_\theta}$ on the log marginal likelihood of the form $\log\ P(\mathcal{O}|\mathcal{M}_\theta)$ as follows:

$$\log\ P(\mathcal{O}|\mathcal{M}_\theta) = \log \sum_\mathcal{H} P(\mathcal{O}, \mathcal{H}|\mathcal{M}_\theta) \tag{5a}$$

$$= \log \sum_\mathcal{H} P(\mathcal{O}, \mathcal{H}|\mathcal{M}_\theta)\ \frac{q_\mathcal{H}(\mathcal{H})}{q_\mathcal{H}(\mathcal{H})} \tag{5b}$$

$$\geq \sum_\mathcal{H} q_\mathcal{H}(\mathcal{H})\ \log \frac{P(\mathcal{O}, \mathcal{H}|\mathcal{M}_\theta)}{q_\mathcal{H}(\mathcal{H})} \tag{5c}$$

$$= E_{q_\mathcal{H}(\mathcal{H})}\big[\log P(\mathcal{O}, \mathcal{H}|\mathcal{M}_\theta)\big] + H\big(q_\mathcal{H}(\mathcal{H})\big) \tag{5d}$$

$$= \mathcal{F}_{\mathcal{M}_\theta}(q_\mathcal{H}(\mathcal{H})). \tag{5e}$$

This lower bound $\mathcal{F}_{\mathcal{M}_\theta}$ in Eq. (5e) is called the free-energy. In Eq. (5d), $E_{q_\mathcal{H}(\mathcal{H})}$ is the expected log marginal likelihood and H is the shannon entropy term. Its role in variational EM (Beal and Ghahramani, 2003) is that it justifies an iterative optimization algorithm for the lower bound whereby one performs the following steps: (the E-step) in which one makes the bound tighter by computing and updating $q_\mathcal{H}(\mathcal{H})$, and (the M-step) which uses the approximation to update the parameters of the model, which typically will increase the log marginal likelihood. If the exact posterior is used, or if the approximation to the posterior is exact, then the inequality is met with equality and the original EM algorithm is obtained. Both LBP and variational EM approaches share a similar objective which is to minimize a corresponding energy equation (see Yedidia et al., 2005), the Gibbs free energy and variational free energy, respectively. Variational inference over hidden or unobserved variables in the E-step of traditional variational EM has an advantage in that it corresponds to minimizing the KL divergence of an approximation and our quantity of interest as we discuss below.

With a little more analysis it is possible to also determine that the free-energy is smaller than the log-marginal likelihood by an amount equal to the Kullback-Leibler (KL) divergence between $q_\mathcal{H}(\mathcal{H})$ and the posterior distribution of the hidden variables $P(\mathcal{H}|\mathcal{O}, \mathcal{M}_\theta)$:

$$\mathcal{F}_{\mathcal{M}_\theta}(q_\mathcal{H}(\mathcal{H})) = \sum_\mathcal{H} q_\mathcal{H}(\mathcal{H})\ \log \frac{P(\mathcal{H}|\mathcal{O}, \mathcal{M}_\theta)P(\mathcal{O}|\mathcal{M}_\theta)}{q_\mathcal{H}(\mathcal{H})} \tag{6a}$$

$$= \log\ P(\mathcal{O}|\mathcal{M}_\theta) - \sum_\mathcal{H} q_\mathcal{H}(\mathcal{H})\ \log \frac{q_\mathcal{H}(\mathcal{H})}{P(\mathcal{O}, \mathcal{H}|\mathcal{M}_\theta)} \tag{6b}$$

$$= \log\ P(\mathcal{O}|\mathcal{M}_\theta) - KL\big[q_\mathcal{H}(\mathcal{H})\ ||\ P(\mathcal{H}|\mathcal{O}, \mathcal{M}_\theta)\big]. \tag{6c}$$

That is, since the marginal likelihood of the observed data is a fixed quantity, maximizing the lower bound (or the variational free energy) through variational

inference over hidden variables is equivalent to minimizing the KL divergence between our approximation and the true distribution over hidden variables. Thus in the E-step of a variational EM algorithm one can perform iterative updates for $q_{\mathcal{H}}(\mathcal{H})$ in a class of distributions $\mathcal{Q}$ in a way that minimizes the KL divergence between the posterior $P(\mathcal{H}|\mathcal{O}, \mathcal{M}_\theta)$ with the goal of obtaining

$$q_{\mathcal{H}}^{\text{VE}}(\mathcal{H}) = \underset{q \in \mathcal{Q}}{\text{argmin}} \; KL\big[q_{\mathcal{H}}(\mathcal{H}) \;||\; P(\mathcal{H}|\mathcal{O}, \mathcal{M}_\theta)\big]. \tag{7}$$

In our work here we will perform variational inference reminiscent of this approach.

3 GEM-MP Framework

At a conceptual level our overall GEM-MP approach consists of the following three key elements. First, we extend the factor graph used to represent a given problem using mega-node random variables which behave identically to groups of variables participating in a factor. Second, we perform variational inference to update an approximation over the original variables and the mega-nodes. Third, we use a probabilistic form of generalized arc consistency to more efficiently make inferences about hard constraints. Unlike inference operations formulated using LBP, since we formulate inference using variational updates we directly minimize the KL divergence between our approximation for the joint conditional distribution and the true distribution of interest.

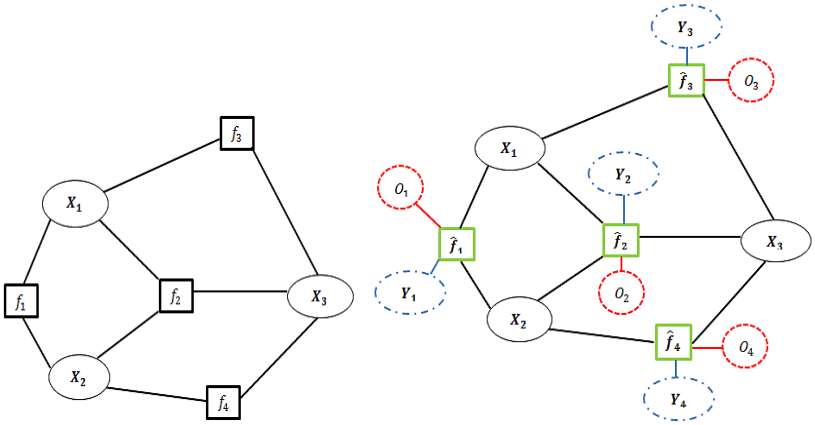


Fig. 2 An example factor graph $\mathcal{G}$ (*left*) which is a fragment of the Cora example in Figure 1, that involves factors $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$ and three random variables $\{X_1, X_2, X_3\}$ representing query ground atoms $\{\text{SBib}(C_2, C_2), \text{SBib}(C_2, C_1), \text{SBib}(C_1, C_2)\}$. The extended factor graph $\hat{\mathcal{G}}$ (*right*) which is a transformation of the original factor graph after adding auxiliary mega-node variables $\mathcal{Y} = \{y_1, y_2, y_3, y_4\}$, and auxiliary activation-node variables $\mathcal{O} = \{O_1, O_2, O_3, O_4\}$, which yields extended factors $\hat{\mathcal{F}} = \left\{\hat{f}_1, \hat{f}_2, \hat{f}_3, \hat{f}_4\right\}$.

Before presenting the inference components of GEM-MP in detail, we will first examine a small concrete example, then present our more general approach for

extending factor graphs. Let us consider a simple example factor graph $\mathcal{G}$ (Figure 2 (*left*)), which is a fragment of the Cora example in Figure 1, that involves factors $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$ and three random variables $\{X_1, X_2, X_3\}$ denoting query ground atoms $\{\text{SBib}(C_2, C_2), \text{SBib}(C_2, C_1), \text{SBib}(C_1, C_2)\}$ respectively. In our GEM-MP framework the first thing we do is to modify the factor graph. Specifically, we need to re-parameterize the factor graph in such a way that carrying out a learning task on the new parameterization is equivalent to running an inference task on the original factor graph. That is, we modify the original factor graph $\mathcal{G}$ (depicted in Figure 2 (*left*)) by transforming it into an *extended factor graph* $\hat{\mathcal{G}}$ (depicted in Figure 2(*right*)) as follows:

- We attach an *auxiliary mega-node* Y_i (dashed oval) to each factor node $f_i \in \mathcal{F}$. Each of these mega-nodes Y_i captures the local entries of its corresponding factor f_i. Thus, it has a domain size that equals (at the most) the number of local entries in the factor f_i (i.e., the states of each mega-node correspond to a subset of the Cartesian product of the domains of the variables that are the arguments to the factor f_i). $\mathcal{Y} = \{Y_i\}_{i=1}^{m}$ is the set of mega-nodes in the extended factor graph, where $m = 4$ in the example factor graph.
- In addition, we connect an *auxiliary activation node*, O_i (dashed circle), to each factor f_i. The auxiliary activation node O_i enforces an indicator constraint $\mathbb{1}_{(Y_i, f_i)}$ for ensuring that the particular configuration of the variables that are the argument to the original factor f_i is identical to the state of the mega-node Y_i:

$$\mathbb{1}_{(Y_i, f_i)} = \begin{cases} 1 & \text{If the state of } Y_i \text{ is identical to local entry of } f_i. \\ 0 & \text{Otherwise} \end{cases} \tag{8}$$

- Now, since we expand the arguments of each factor f_i by including both auxiliary mega-node and auxiliary activation node variables, then we get an *extended factor* $\hat{f}_i$. $\hat{\mathcal{F}} = \left\{\hat{f}_i\right\}_{i=1}^{m}$ is the set of extended factors in the extended factor graph.
- When the activation node O_i equals one, then it activates the indicator constraint in Eq. (8). If this indicator constraint is satisfied, then the extended factor graph $\hat{f}_i$ preserves the same value of f_i for the configuration that is defined over the original input variables defining the factor f_i. Thus, clearly, the following condition holds for each extended factor $\hat{f}_i$ when a configuration, $(x_1, \ldots, x_n)$, of f_i equals to state, y_i, of mega-node, Y_i:

$$\hat{f}_i(X_1 = x_1, \ldots, X_n = x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i = 1} = f_i(X_1 = x_1, \ldots, X_n = x_n). \tag{9}$$

  But if the indicator constraint in Eq. (8) is not satisfied then the extended factor graph $\hat{f}_i$ assigns a value 0. Thus, this condition also holds for each extended factor $\hat{f}_i$ when a configuration $(x_1, \ldots, x_n)$ of f_i is not equal to state y_i of mega-node, Y_i:

$$\hat{f}_i(X_1 = x_1, \ldots, X_n = x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i = 1} = 0. \tag{10}$$

- Setting $O_i = 0$ effectively removes the impact of f_i from the model. That is, when the activation node O_i is not equal to one, then it deactivates the

Table 2 Factor f_1 in the original factor graph (*left*). Its corresponding extended factor $\hat{f}_1$ in the extended factor graph (*right*). When the activation node $O_1 = 1$, the bold values are cases in which the extended factor $\hat{f}_1$ preserves the same value of f_1. Otherwise it assigns a value 0. When the activation node $O_1 \neq 1$, the matches between Y_1 and (X_1, X_2) are cases in which $\hat{f}_1$ assigns a value 1. Otherwise it assigns a value 0.

X_1	X_2	$f_1(X_1, X_2)$
T	T	1
T	F	0
F	T	1
F	F	1

X_1	X_2	Y_1	O_1	$\hat{f}_1(X_1, X_2, Y_1, O_1)$
T	T	TT	1	**1**
T	F	TT	1	0
F	T	TT	1	0
F	F	TT	1	0
T	T	TT	0	1
T	F	TT	0	0
F	T	TT	0	0
F	F	TT	0	0
T	T	TF	1	0
T	F	TF	1	**0**
F	T	TF	1	0
F	F	TF	1	0
T	T	TF	0	0
T	F	TF	0	1
F	T	TF	0	0
F	F	TF	0	0
T	T	FT	1	0
T	F	FT	1	0
F	T	FT	1	**1**
F	F	FT	1	0
T	T	FT	0	0
T	F	FT	0	0
F	T	FT	0	1
F	F	FT	0	0
T	T	FF	1	0
T	F	FF	1	0
F	T	FF	1	0
F	F	FF	1	**1**
T	T	FF	0	0
T	F	FF	0	0
F	T	FF	0	0
F	F	FF	0	1

indicator constraint in Eq. (8). Here, the extended factor $\hat{f}_i$ assigns a value 1 when the possible state of Y_i matches the configuration of variables that are the arguments to the factor f_i. Otherwise it assigns a value 0. Note that by assigning the values in this way, all factors $f_i \in \mathcal{F}$ will have identical values in their corresponding $\hat{f}_i \in \hat{\mathcal{F}}$ when $O_i = 0$. This implies that the deactivation of their indicator constraint has no impact on the distribution from the inclusion of the factors $f_i \in \mathcal{F}$.

Table 2 visualizes the expansion of factor f_1, in the original factor graph, to its corresponding extended factor $\hat{f}_1$ in the extended factor graph.

Proposition 1 *In the extended factor graph $\hat{\mathcal{G}}$, reducing each extended factor $\hat{f}_i$ by evidencing its activation node with one, $\bar{O}_i = 1$, and then eliminating its auxiliary mega-node Y_i by marginalization yields its corresponding original factor f_i in the*

original factor graph $\mathcal{G}$.

$$\sum_{Y_i} \hat{f}_i(X_1, \ldots, X_n, Y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = f_i(X_1, \ldots, X_n),\ \forall \hat{f}_i \in \hat{\mathcal{F}}. \tag{11}$$

Proof see Appendix A. □

Proposition 2 *Any arbitrary factor graph* $\mathcal{G}$ *is equivalent, i.e., defines an identical joint probability over variables* $\mathcal{X}$*, to its extended* $\hat{\mathcal{G}}$ *iff the activation nodes in* $\hat{\mathcal{G}}$ *are evidenced with one:*

$$\mathcal{G} \equiv \hat{\mathcal{G}} \textit{ iff } \bar{O}_i = 1,\ \forall O_i \in \mathcal{O} \textit{ in } \hat{\mathcal{G}}$$

Proof see Appendix A. □

Given this extended factor graph formulation we can now examine the task of performing inference over unobserved quantities given observed quantities through the lens of variational analysis and inference.

Let $\mathcal{O} = \{O_i\}_{i=1}^m$ be the observed variables, represented as a binary vector (of 1's), indicating the observation of the activation node variables $\bar{O}_i = 1,\ \forall O_i \in \mathcal{O}$. Let $\mathcal{H} = \{\mathcal{X}, \mathcal{Y}\}$ be the hidden variables, where $\mathcal{X} = \{X_j\}_{j=1}^n$ is a set of variables (i.e., ground atoms) whose marginals we want to compute, and $\mathcal{Y} = \{Y_i\}_{i=1}^m$ is the set of mega-nodes. Let $q(\mathcal{X}, \mathcal{Y})$ be an auxiliary distribution over the set of hidden variables $\mathcal{H}$, satisfying that $\sum_{\mathcal{X},\mathcal{Y}}\ q(\mathcal{X}, \mathcal{Y}) = 1$. Now using the distribution $q(\mathcal{X}, \mathcal{Y})$, we can leverage Jensens inequality to obtain a lower bound on the log-marginal likelihood of the form $\log P(\mathcal{O}|\mathcal{M})$ as follows:[3]

$$\begin{aligned}
\log P(\mathcal{O}|\mathcal{M}) &= \log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M}) && \text{(12a)}\\
&= \log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\ \frac{q(\mathcal{X}, \mathcal{Y})}{q(\mathcal{X}, \mathcal{Y})} && \text{(12b)}\\
&\geq \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X}, \mathcal{Y})\ \log \frac{P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})}{q(\mathcal{X}, \mathcal{Y})} && \text{(12c)}\\
&= E_{q(\mathcal{X},\mathcal{Y})}\big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\big] + H\big(q(\mathcal{X}, \mathcal{Y})\big) && \text{(12d)}\\
&= \mathcal{F}_{\mathcal{M}}(q(\mathcal{X}, \mathcal{Y})) && \text{(12e)}
\end{aligned}$$

where $\mathcal{F}_{\mathcal{M}}$ in Eq. (12e) is the negative of a quantity that represents the variational free energy functional of the free distribution $q(\mathcal{X}, \mathcal{Y})$. As in Eq. (12d), it is a summation of two terms: $E_{q(\mathcal{X},\mathcal{Y})}$ which is the expected log marginal-likelihood with respect to the distributions, $q(\mathcal{X}, \mathcal{Y})$, and the second term, $H\big(q(\mathcal{X}, \mathcal{Y})\big)$, is the negative entropy of the distribution $q(\mathcal{X}, \mathcal{Y})$ (see Neal and Hinton, 1999, for more details).

We can also easily see that similarly to other traditional settings the free-energy $\mathcal{F}_{\mathcal{M}}$ is smaller than the log-marginal likelihood by an amount equal to the

[3] Note that "log" is a concave function and it can play an important role in maintaining convergences via Jensen's inequality, as will be explained further on.

Kullback-Leibler (KL) divergence between $q(\mathcal{X},\mathcal{Y})$ and the distribution over the hidden variables $P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})$:

$$\mathcal{F}_{\mathcal{M}}(q(\mathcal{X},\mathcal{Y})) = \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X},\mathcal{Y}) \ \log \frac{P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})P(\mathcal{O}|\mathcal{M})}{q(\mathcal{X},\mathcal{Y})} \tag{13a}$$

$$= \log P(\mathcal{O}|\mathcal{M}) - \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X},\mathcal{Y}) \ \log \frac{q(\mathcal{X},\mathcal{Y})}{P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})} \tag{13b}$$

$$= \log P(\mathcal{O}|\mathcal{M}) - KL\big[q(\mathcal{X},\mathcal{Y}) \ || \ P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\big] \tag{13c}$$

Since the $KL\big[q(\mathcal{X},\mathcal{Y}) \ || \ P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\big] \geq 0$ in Eq. (13c) and the log marginal probability under the model is a fixed quantity, then minimizing the KL divergence term is equivalent to maximizing the variational free energy $\mathcal{F}_{\mathcal{M}}$. That is, one could equivalently select either to maximize the lower bound (the variational free energy), or to minimize the KL divergence. Based on that, we now want to infer the distribution $q(\mathcal{X},\mathcal{Y})$ in a class of distributions $\mathcal{Q}$ that maximize the variational free energy:[4]

$$q^*(\mathcal{X},\mathcal{Y}) = \underset{q\in\mathcal{Q}}{\operatorname{argmax}} \ \mathcal{F}_{\mathcal{M}}(q(\mathcal{X},\mathcal{Y})) \tag{14}$$

One problem is that the target of the maximization of the variational free energy $\mathcal{F}_{\mathcal{M}}$ is unwieldy for direct optimization. The variational free energy requires an explicit summation over all possible instantiations of $\mathcal{X}$ and all valid local entries of the factors (i.e., ground clauses) involved in the model for $\mathcal{Y}$, which is an operation that is infeasible in practice.[5] Instead we constrain the auxiliary $q(\mathcal{X},\mathcal{Y})$ distribution to be a factorized (separable) approximation as:

$$q(\mathcal{X},\mathcal{Y}) = q(\mathcal{X};\mathcal{B}_{\mathcal{X}}) \ q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) \tag{15}$$

where $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ is an approximation of the true distribution $P(\mathcal{X}|\mathcal{O},\mathcal{M})$ over hidden variables, $\mathcal{X}$. This distribution is characterized by a set of variational parameters, $\mathcal{B}_{\mathcal{X}} = \big\{\beta_{X_j}\big\}_{j=1}^{n}$. Since we use a fully factored distribution, these approximations are somewhat similar to the approximate marginal probabilities of variables $X_j \in \mathcal{X}$, which one might obtain using standard loopy message-passing inference (e.g., LBP); however, unlike the situation with LBP, here we can subject these approximations to a variational analysis leading to an understanding of message-passing inference in terms of KL divergence minimization. The distribution $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ is an approximation to the true distribution $P(\mathcal{Y}|\mathcal{O},\mathcal{M})$ over hidden mega-nodes, $\mathcal{Y}$, which is characterized by a set of variational parameters, $\mathcal{T}_{\mathcal{Y}} = \{\alpha_{Y_i}\}_{i=1}^{m}$, for adapting the weights associated with the particular states of mega-nodes $Y_i \in \mathcal{Y}$. As a particular formulation of how the $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ distribution is parametrized, these variational parameters $\alpha_{Y_i}(f_i)$ can be defined as:

$$\alpha_{Y_i}(f_i) = \begin{cases} v_s & \text{if the state of } Y_i \text{ satisfies } f_i, \\ v_u & \text{otherwise.} \end{cases} \tag{16}$$

[4] This is equivalent to finding the $q(\mathcal{X},\mathcal{Y})$ that minimizes the KL between the true distribution and its approximation : $q^*(\mathcal{X},\mathcal{Y}) = \underset{q}{\operatorname{argmin}} \ KL\big[q(\mathcal{X},\mathcal{Y}) \ || \ P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\big]$

[5] An additional source of intractability arises in many models (e.g., SRL models) in which the number of hidden variables is very large. For instance, a model with N binary hidden variables generally requires a distribution over all 2^N possible states of those variables. So even for moderately large N this results in computational intractability.

where v_s (and v_u) are the values obtained from f_i when a particular state of Y_i satisfies (or unsatisfies) the factor f_i respectively. Note that v_s and v_u can be adapted using the distributions of the factor f_i's argument variables and the weight associated with f_i (as will be explained in Subsections 4.1 and 4.2 for hard and soft factors respectively). Now, using Eq. (15), we can simply represent the lower bound as follows:

$$\mathcal{F}_{\mathcal{M}}(q(\mathcal{X},\mathcal{Y})) = \mathcal{F}_{\mathcal{M}}(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})\ q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})) \tag{17a}$$

$$= \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X};\mathcal{B}_{\mathcal{X}}) q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\ \log \frac{P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})}{q(\mathcal{X};\mathcal{B}_{\mathcal{X}}) q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} \tag{17b}$$

$$= E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] + H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) \tag{17c}$$

where $E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}$ is the expected log marginal-likelihood with respect to the distributions, $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ and $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$, and the negative of the second term, $-H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big)$, is the entropy. Hence, we can now set up our goal to find the distributions $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ and $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ that maximize the lower bound $\mathcal{F}_{\mathcal{M}}$.

Now the role of the GEM-MP algorithm is to iteratively maximize the lower bound $\mathcal{F}_{\mathcal{M}}$ (or minimize the negative free energy $-\mathcal{F}_{\mathcal{M}}$) with respect to the distributions $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ and $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ by applying two steps. In the first step, $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ is used to maximize $\mathcal{F}_{\mathcal{M}}$ with respect to $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$. Then in the second step, $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ is used to maximize $\mathcal{F}_{\mathcal{M}}$ with respect to $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$. That is, GEM-MP maximizes $\mathcal{F}_{\mathcal{M}}$ by performing two iterative updates

$$\text{- } \mathcal{T}^*_{\mathcal{Y}} \propto \operatorname*{argmax}_{\mathcal{T}_{\mathcal{Y}}} E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] + H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) \tag{18a}$$

$$\text{- } \mathcal{B}^*_{\mathcal{X}} \propto \operatorname*{argmax}_{\mathcal{B}_{\mathcal{X}}} E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] + H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) \tag{18b}$$

Note that the entropy term can be re-written as:

$$H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) = H\big(q(\mathcal{X},\mathcal{Y})\big) = H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})\big) + H\big(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) \tag{19}$$

Now if we substitute the entropy $H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big)$ from Eq. (19) into Eqs. (18a) and (18b), then we will have that $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ does not depend on the entropy $H(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}))$ when maximizing $\mathcal{F}_{\mathcal{M}}$ with respect to the variational parameters $\mathcal{T}_{\mathcal{Y}}$, and $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ does not depend on $H(q(\mathcal{X};\mathcal{B}_{\mathcal{X}}))$ when maximizing $\mathcal{F}_{\mathcal{M}}$ with respect to the variational parameters $B_{\mathcal{X}}$.[6] We thus have

$$\text{- } \mathcal{T}^*_{\mathcal{Y}} \propto \operatorname*{argmax}_{\mathcal{T}_{\mathcal{Y}}} E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] + H\big(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) \tag{20a}$$

$$\text{- } \mathcal{B}^*_{\mathcal{X}} \propto \operatorname*{argmax}_{\mathcal{B}_{\mathcal{X}}} E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] + H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})\big). \tag{20b}$$

Therefore, the goal of GEM-MP can be expressed as that of maximizing a lower bound on the log marginal-likelihood by performing two steps, using superscript (t) to denote the iteration number:

[6] Note that when the optimization is over the parameters $\mathcal{T}_{\mathcal{Y}}$, that affects the function $H(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}))$ and not $H(q(\mathcal{X};\mathcal{B}_{\mathcal{X}}))$, i.e., only $H(q(\mathcal{X};\mathcal{B}_{\mathcal{X}}))$ is dropped.

– GEM-MP "$M_{q(\mathcal{Y})}$-step": (for maximizing mega-nodes' parameters distributions)

$$\overbrace{\mathcal{T}_{\mathcal{Y}}^{(\mathrm{t}+1)}}^{\text{Max. w.r.t } q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}} = \underset{\mathcal{T}_{\mathcal{Y}}}{\operatorname{argmax}} \overbrace{E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}^{(\mathrm{t})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}^{(\mathrm{t})}} \big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big]}^{\text{E-step}} + H\big(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\big) \tag{21}$$

– GEM-MP "$M_{q(\mathcal{X})}$-step": (for maximizing variable-nodes' parameter distributions)

$$\overbrace{\mathcal{B}_{\mathcal{X}}^{(\mathrm{t}+1)}}^{\text{Max. w.r.t } q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}} = \underset{\mathcal{B}_{\mathcal{X}}}{\operatorname{argmax}} \overbrace{E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}^{(\mathrm{t})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}^{(\mathrm{t}+1)}} \big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big]}^{\text{E-step}} + H\big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})\big) \tag{22}$$

where here the arguments to Eqs. (21) and (22) are the $E_{q(\mathcal{X})}$-step and $E_{q(\mathcal{Y})}$-step corresponding to $M_{q(\mathcal{Y})}$-step and $M_{q(\mathcal{X})}$-step, respectively. In Eq. (21), the current value of $q\left(\mathcal{X};\mathcal{B}_{\mathcal{X}}\right)$ and $q\left(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}\right)$ are used to optimize the mega-nodes' variational parameters $\mathcal{T}_{\mathcal{Y}}$. This could result in maximizing the lower bound with respect to $q\left(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}\right)$. Next, in Eq. (22), the new value of $q\left(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}\right)$ and the current value of $q\left(\mathcal{X};\mathcal{B}_{\mathcal{X}}\right)$ are used to optimize the nodes' variational parameters $\mathcal{B}_{\mathcal{X}}$. This could result in maximizing the lower bound once again but this time with respect to $q\left(\mathcal{X};\mathcal{B}_{\mathcal{X}}\right)$. Note that the difficulties in dealing with the expectation in Eqs. (21) and (22) depends on the properties of the distributions $q\left(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}\right)$ and $q\left(\mathcal{X};\mathcal{B}_{\mathcal{X}}\right)$. That is if the inference is easy in these two distributions, then evaluating the expectation should be relatively easily. For the entropy terms, the choice of how to approximate the distributions $q\left(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}\right)$ and $q\left(\mathcal{X};\mathcal{B}_{\mathcal{X}}\right)$ determines whether we can evaluate the entropy terms. As will be shown hereafter, using the variational mean-field for approximating the distributions $q\left(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}\right)$ and $q\left(\mathcal{X};\mathcal{B}_{\mathcal{X}}\right)$ makes evaluating the entropy terms tractable.

Now, using a fully factored variational mean field approximation for $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ implies that we create our approximation from independent distributions over the hidden (mega-node) variables as follows:

$$q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) = \prod_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \tag{23}$$

where $q(Y_i;\alpha_{Y_i})$ is our complete approximation to the true probability distribution $P(Y_i|\mathcal{O},\mathcal{X},\mathcal{M})$ of a randomly chosen valid local entry of mega-node Y_i. Also, the variational mean-field approximation to $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ is similarly defined as a factorization of independent distributions over the hidden variables in $\mathcal{X}$, and can be expressed as follows:

$$q(\mathcal{X};\mathcal{B}_{\mathcal{X}}) = \prod_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}), \tag{24}$$

where $q(X_j;\beta_{X_j})$ is an approximate distribution for the true marginal probability distribution $P(X_j|\mathcal{O},\mathcal{M})$ of variable X_j.

From Eqs. (23) and (24), we can write the expected log marginal-likelihood in Eqs. (21) and (22), as follows:

$$\begin{aligned} &E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}} \big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] \\ &\qquad = \sum_{\mathcal{Y}} \prod_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \Big[\sum_{\mathcal{X}} \prod_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}) \log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\Big] \end{aligned} \tag{25}$$

We now proceed to optimize the lower bound, through the use of Eqs. (21) and (22), using our variational mean-field approximations for both $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ and $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$.

We use Eqs. (25), (24) and (23) in Eq. (21). Hence, we have a maximization of the lower bound on the log marginal-likelihood as

$$\mathcal{T}_{\mathcal{Y}}^{(\mathrm{t}+1)} = \underset{\mathcal{T}_{\mathcal{Y}}}{\operatorname{argmax}} \sum_{\mathcal{Y}} \prod_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \Big[\sum_{\mathcal{X}} \prod_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}) \log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M}) \Big] + \sum_{Y_i \in \mathcal{Y}} H(q(Y_i;\alpha_{Y_i})) \tag{26}$$

One can then separate out the terms related to the updates of the variational parameters for each mega-node Y_i in Eqs (26). In addition, updating the parameter distribution of mega-node Y_i requires considering the distributions $\{q(X_j;\beta_{X_j})\}$ of only the variables in $\mathcal{X}$ that are arguments to the extended factor $\hat{f}_i$ (i.e., $X_j \in \mathcal{X}_{\hat{f}_i}$)- where Y_i is attached to $\hat{f}_i$. That is

$$\alpha_{Y_i}^{(\mathrm{t}+1)} = \underset{\alpha_{Y_i}}{\operatorname{argmax}} \sum_{Y_i} q(Y_i;\alpha_{Y_i}) \Big[\sum_{\mathcal{X}_{\hat{f}_i}} \prod_{X_j \in \mathcal{X}_{\hat{f}_i}} q(X_j;\beta_{X_j}) \log \hat{f}_i(O_i,\mathcal{X}_{\hat{f}_i},Y_i|\mathcal{M}) \Big] + H(q(Y_i;\alpha_{Y_i})) \tag{27}$$

where $\hat{f}_i(O_i,\mathcal{X}_{\hat{f}_i},Y_i|\mathcal{M})$ is the part of $P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})$ in the model with the factor associated with the mega-node Y_i.

This in fact allows optimizing the variational parameters of the distributions of each mega-node Y_i as

$$q(Y_i;\alpha_{Y_i}^*) = \frac{1}{\mathcal{Z}_{Y_i}} \exp\left(\overbrace{E_{\{q^{(t)}(X_j;\beta_{X_j}) \mid X_j \in \mathcal{X}_{\hat{f}_i}\}} \big[\log \hat{f}_i(O_i,\mathcal{X}_{\hat{f}_i},Y_i|\mathcal{M}) \big]}^{E_{q(\mathcal{X})}\text{-step}} \right) \tag{28}$$

where $\mathcal{Z}_{Y_i} = \sum E_{\{q^{(t)}(X_k;\beta_{X_k}) \mid X_k \neq X_j,\ X_j \in \mathcal{X}_{\hat{f}_i}\}} \big[\log \hat{f}_i(O_i,\mathcal{X}_{\hat{f}_i},Y_i|\mathcal{M}) \big]$ is the normalization factor, and the expectation part can be written as

$$\begin{aligned} &E_{\{q^{(t)}(X_j;\beta_{X_j}) \mid X_j \in \mathcal{X}_{\hat{f}_i}\}} \big[\log \hat{f}_i(O_i,\mathcal{X}_{\hat{f}_i},Y_i|\mathcal{M}) \big] \\ &= \sum_{\mathcal{X}_{\hat{f}_i}} \prod_{X_j \in \mathcal{X}_{\hat{f}_i}} q(X_j;\beta_{X_j}) \log \hat{f}_i(O_i,\mathcal{X}_{\hat{f}_i},Y_i|\mathcal{M}) \end{aligned} \tag{29}$$

This update is similar in form to the simpler case of fully factored mean field updates in a model without the additional mega-nodes. See Winn (2004) for more details on the traditional mean field updates. Note that here by using Eqs. (29)

and (28) in Eq. (27), we also have that

$$\alpha_{Y_i}^{(t+1)} = \underset{\alpha_{Y_i}}{\text{argmax}} \sum_{Y_i} q(Y_i;\alpha_{Y_i}) \log q(Y_i;\alpha^*_{Y_i}) + H(q(Y_i;\alpha_{Y_i}) - \log \mathcal{Z}_{Y_i} \quad (30a)$$

$$= \underset{\alpha_{Y_i}}{\text{argmax}} - KL\big[q(Y_i;\alpha_{Y_i}) \,||\, q(Y_i;\alpha^*_{Y_i})\big] + const. \quad (30b)$$

$$= \underset{\alpha_{Y_i}}{\text{argmax}} - KL\big[q(Y_i;\alpha_{Y_i}) \,||\, q(Y_i;\alpha^*_{Y_i})\big], \quad (30c)$$

where $KL\big[q(Y_i;\alpha_{Y_i}) \,||\, q(Y_i;\alpha^*_{Y_i})\big]$ is the Kullback-Leibler divergence. The constant, in Eq. (30b), is simply the logarithm of the normalization factor representing the variables' $\{q(X_j;\beta_{X_j})\}$, that are independent of $q(Y_i;\alpha_{Y_i})$. Note that, from Eq. (30c), we maximize on the lower bound with respect to $q(Y_i;\alpha_{Y_i})$ by minimizing the Kullback-Leibler divergence. This means that the lower bound can be maximized by setting $q(Y_i;\alpha_{Y_i}) = q(Y_i;\alpha^*_{Y_i})$.

Now, likewise, when updating the distribution of each variable X_j we only consider the updated distributions $\{q(Y_i;\alpha_{Y_i})\}$ of mega-nodes attached to the extended factors on which X_j appears (i.e., $\hat{f}_i \in \hat{\mathcal{F}}_{X_j}$). That is

$$q(X_j;\beta^*_{X_j}) = \frac{1}{\mathcal{Z}_{X_j}} \exp\left(\overbrace{E_{\{q^{(t+1)}(Y_i;\alpha_{Y_i}),\, q^{(t)}(X_k;\beta_{X_k}) \,|\, \hat{f}_i \in \hat{\mathcal{F}}_{X_j}\}} \big[\log \hat{F}(O, X_j, \mathcal{Y}|\mathcal{M})\big]}^{E_{q(\mathcal{Y})}\text{-step}} \right) \quad (31)$$

where $\hat{F}(O, X_j, \mathcal{Y}|\mathcal{M})$ is the part of $P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})$ in the model with the factors associated with the node X_j. This part involves only the mega-nodes' qs in the Markov boundary of each X_j, and the qs from the old iteration for the other variables $X_k \neq X_j$ that are arguments to factors in which X_j appears.

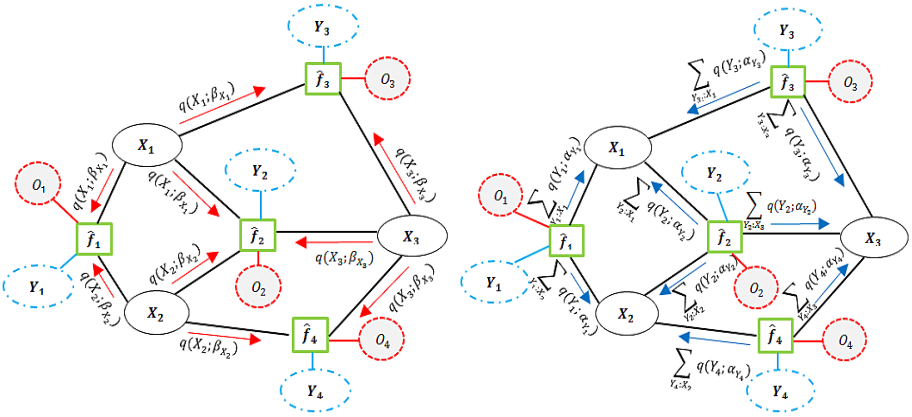


Fig. 3 Illustrating message-passing process of GEM-MP. (*left*) $E_{q(\mathcal{X})}$-step messages from variables-to-factors; (*right*) $E_{q(\mathcal{Y})}$-step messages from factors-to-variables.

At this point, we have paved the way for GEM-MP message-passing inference by transforming the inference task into an instance of an EM style approach often associated with learning tasks. The GEM-MP inference proceeds by iteratively

sending two types of messages on the extended factor graph so as to compute the updated q distributions needed for the M-steps above. The E_q and M_q steps are alternated until converging to a local maximum[7] of $\mathcal{F}_{\mathcal{M}}(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}), q(\mathcal{X};\mathcal{B}_{\mathcal{X}}))$. These messages are different from simply running the standard LBP algorithm, and they are formulated in the form of E (i.e., $E_{q(\mathcal{X})}$, $E_{q(\mathcal{Y})}$) and M (i.e., $M_{q(\mathcal{Y})}$, $M_{q(\mathcal{X})}$) steps outlined in Eqs. (21), (28), (22), and (31), where the E-steps can be computed through message passing procedures as outlined below:

- $E_{q(\mathcal{X})}$**-step messages,** $\{\mu_{X_j \to \hat{f}_i} = q(X_j;\beta_{X_j})\}$, that are sent from variables $\mathcal{X}$ to factors $\hat{\mathcal{F}}$ (as depicted in Figure 3 (*left*)). The aim of sending these messages is to perform the GEM-MP's $M_{q(\mathcal{Y})}$-step in Eq. (21). That is, the setting of the distributions, $\{q(X_j;\beta_{X_j})\}_{\forall X_j \in \mathcal{X}}$, are used for estimating the distributions, $\{q(Y_i;\alpha_{Y_i})\}_{\forall Y_i \in \mathcal{Y}}$, that maximizes the lower bound on the log marginal-likelihood of Eq. (21). To do so, each variable $X_j \in \mathcal{X}$ sends its current marginal probability β_{X_j} as an $E_{q(\mathcal{X})}$-step message, $\mu_{X_j \to \hat{f}_i} = q(X_j;\beta_{X_j})$, to its neighboring extended factors. Then, at the factors level, each extended factor $\hat{f}_i \in \hat{\mathcal{F}}$ uses the relevant marginals from those received incoming messages of its argument variables, i.e., $\{q(X_j;\beta_{X_j})\}_{\forall X_j \in \mathcal{X}_{\hat{f}_i}}$, to perform the computations of the $E_{q(\mathcal{X})}$-step of Eq. (21). This implies updating the distribution $q(Y_i;\alpha_{Y_i})$ of its mega-node Y_i by computing what we call the probabilistic generalized arc consistency (pGAC) (we will discuss pGAC in more detail in section 4).
- $E_{q(\mathcal{Y})}$**-step messages,** $\{\mu_{\hat{f}_i \to X_j} = \sum_{Y_i : \forall y_k(X_j)} q(Y_i;\alpha_{Y_i})\}$, that are sent from factors to variables (as depicted in Figure 3 (*right*)). Sending these messages corresponds to the GEM-MP's $M_{q(\mathcal{X})}$-step in Eq. (22). Here, the approximation of the distributions, $\{q(Y_i;\alpha_{Y_i})\}_{\forall Y_i \in \mathcal{Y}}$, obtained from the GEM-MP's $M_{q(\mathcal{Y})}$-step will be used to update the marginals, i.e., $\{q(X_j;\beta_{X_j})\}_{\forall X_j \in \mathcal{X}}$, that maximizes the lower bound on the log marginal-likelihood in Eq. (22). Characteristically, each extended factor $\hat{f}_i \in \hat{\mathcal{F}}$ sends a corresponding refinement of the pGAC distribution - that approximates the $q(Y_i;\alpha_{Y_i})$ of its mega-node - as an $E_{q(\mathcal{Y})}$-step message, $\mu_{\hat{f}_i \to X_j} = \sum_{Y_i : \forall y_k(X_j)} q(Y_i;\alpha_{Y_i})$, to each of its argument variables, $X_j \in \mathcal{X}_{\hat{f}_i}$. Now, at the variables level, each $X_j \in \mathcal{X}$ uses the relevant refinement of pGAC distributions from those received incoming messages - which are the outgoing messages coming from its extended factors $\hat{f}_i \in \hat{\mathcal{F}}_{X_j}$ - to perform the computations of the $E_{q(\mathcal{Y})}$-step of Eq. (22). This implies updating its distribution $q(X_j;\beta_{X_j})$ by summing these messages (as it will be discussed in more detail in section 4).

Other work has empirically observed that asynchronous belief propagation scheduling often yields faster convergence than synchronous schemes (cf. Elidan et al., 2006). In variational message passing schemes, the mathematical derivations lead to updates that are asynchronous in nature. In Section 4 we will derive a general update-rule for GEM-MP based on variational principles in more detail and we shall see it leads to an asynchronous scheduling of messages. However, messages can be passed in a structured form whereby variables $\mathcal{X}$ are able to send their $E_{q(\mathcal{X})}$-step messages simultaneously to their factors (or mega-node variables

[7] This is equivalent to converging to a local minimum of the negative free energy functional $-\mathcal{F}_{\mathcal{M}}$, which is a stable local minimumwith respect to an inference task on the original factor graph.

$\mathcal{Y}$). At the level of factors, the marginals are updated one at a time, then the factors send back $E_{q(\mathcal{Y})}$-step messages simultaneously to their variables. Moreover, it should be noted that we do the asynchronous updating schedule between variables $\mathcal{X}$ and mega-nodes $\mathcal{Y}$ in a form that allows updates to potentially be computed in parallel. Thus the version of GEM-MP that we present here involves sending messages in parallel from mega-nodes to variables and variables to mega-nodes. Updates to the $q(x_j)$s approximations for variables $X_j \in \mathcal{X}$ could be computed in parallel, and updates to the $q(y_i)$s approximations for mega-nodes $Y_i \in \mathcal{Y}$ could also be performed in parallel.

Theorem 1 (GEM-MP Guarantees Convergence) *At each iteration of updating the marginals (i.e., variational parameters $\mathcal{B}_x$), GEM-MP increases monotonically the lower bound on the model evidence such that it never overshoots the global optimum or until converging naturally to some local optima.*

Proof Using Eq. (15) into Eq. (13c) implies that the maximization of the lower bound $\mathcal{F}_{\mathcal{M}}(q(\mathcal{X};\mathcal{B}_x), q(\mathcal{Y};\mathcal{T}_y))$ is equivalent to the minimization of the Kullback-Leibler (KL) divergence between $q(x;\mathcal{B}_x)$ and $q(\mathcal{Y};\mathcal{T}_y)$ and the true distribution, $P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})$, over hidden variables:

$$\log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M}) - \mathcal{F}_{\mathcal{M}} = \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X};\mathcal{B}_x) q(\mathcal{Y};\mathcal{T}_y) \log \frac{q(\mathcal{X};\mathcal{B}_x) q(\mathcal{Y};\mathcal{T}_y)}{P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})} \tag{32a}$$

$$= KL\Big[q(\mathcal{X};\mathcal{B}_x) q(\mathcal{Y};\mathcal{T}_y) \;||\; P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M}) \Big] \tag{32b}$$

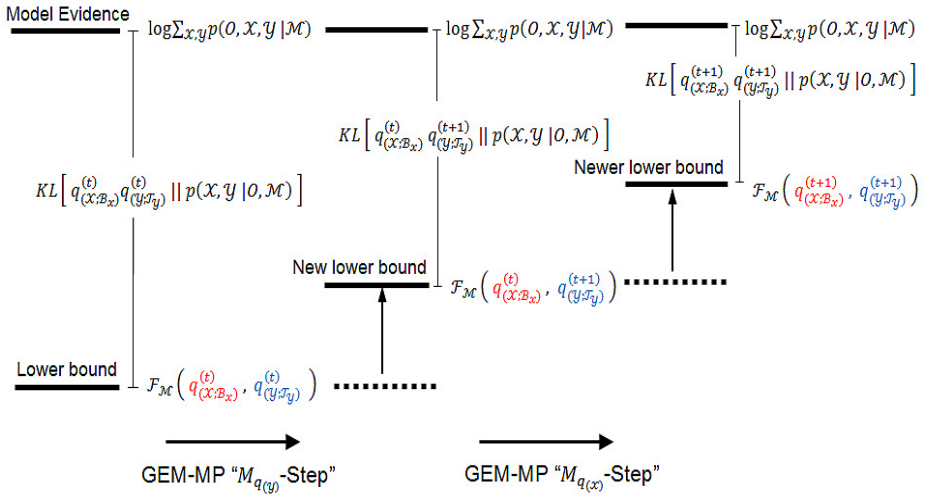


Fig. 4 Illustrating how each step of the GEM-MP algorithm is guaranteed to increase the lower bound on the log marginal-likelihood. In its "$M_{q(\mathcal{Y})}$-step", the variational distribution over hidden mega-node variables is maximized according to Eq. (21). Then, in its "$M_{q(\mathcal{X})}$-step", the variational distribution over hidden $\mathcal{X}$ variables is maximized according to Eq. (22).

Now assume that before and after a given iteration (t), we have $q^{(t)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}$ and $q^{(t+1)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}$ that denote the settings of $\mathcal{B}_\mathcal{X}$ respectively. Likewise for $q^{(t)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})}$ and $q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})}$ with respect to $\mathcal{T}_\mathcal{Y}$, where one iteration is a run of GEM-MP "$M_{q(\mathcal{Y})}$-step" followed by "$M_{q(\mathcal{X})}$-step". By construction, in the $M_{q(\mathcal{Y})}$-step, $q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})}$ is chosen such that it maximizes $\mathcal{F}_\mathcal{M}(q^{(t)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})})$ given $q^{(t)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}$. Then, in the $M_{q(\mathcal{X})}$-step, $q^{(t+1)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}$ is set by maximizing $\mathcal{F}_\mathcal{M}(q^{(t+1)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})})$ given $q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})}$, and we have (as shown in Figure 4):

$$KL\Big[q^{(t)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}\, q^{(t)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})} \mid\mid P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \geq KL\Big[q^{(t)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})} \mid\mid P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \tag{33}$$

and similarly:

$$KL\Big[q^{(t)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})} \mid\mid P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \geq KL\Big[q^{(t+1)}_{(\mathcal{X};\mathcal{B}_\mathcal{X})}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_\mathcal{Y})} \mid\mid P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \tag{34}$$

This implies that GEM-MP increases the lower bound monotonically.
Now since the exact log marginal-likelihood, $\log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})$, is a fixed quantity and the Kullback-Leibler divergence, $KL \geq 0$, is a non-negative quantity then this implies that GEM-MP never overshoots the global optimum of the variational free energy.

Since GEM-MP applies a variational mean-field approximation for $q(\mathcal{Y};\mathcal{T}_\mathcal{Y})$ and $q(\mathcal{X};\mathcal{B}_\mathcal{X})$ distributions (refer to Eqs. (24) and (23)) over both mega-nodes and variables nodes respectively, it inherits the guarantees of mean field to converge to a local minimum of the negative variational free energy free energy or KL divergence. □

Note that the convergence behaviour of GEM-MP for inference task resembles the behaviour of the variational Bayesian expectation maximization approach proposed by Beal and Ghahramani (2003) for the Bayesian learning task. Both of them can be seen as a variational technique (forming a factorial approximation) that minimizes a free-energy-based function for estimating the marginal likelihood of the probabilistic models with hidden variables.

It is worth noting that when reaching the GEM-MP "$M_{q(\mathcal{Y})}$-step", one could select between a local or global approximation to distribution $q(\mathcal{Y};\mathcal{T}_\mathcal{Y})$. However, in this paper, we restricted ourselves to local approximations.[8] Furthermore, although GEM-MP represents a general template framework for applying variational inference to probabilistic graphical models, we concentrate on Markov logic models, where the variables will be ground atoms and the factors will be both hard and soft ground clauses (as will be explained in Section 4) and Ising models (as will be explained in Section 5).

[8] Note that the local approximation means that we handle mega-nodes individually. This appears in the factorization of $q(\mathcal{Y};\mathcal{T}_\mathcal{Y})$ into independent distributions (refer to Eq. (23)).

4 GEM-MP General Update Rule for Markov Logic

By substituting the local approximation for $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ from $M_{q(\mathcal{Y})}$-step into the $M_{q(\mathcal{X})}$-step, we can synthesize update rules that tell us how to set the new marginal in terms of the old one. So, in practice the $M_{q(\mathcal{Y})}$-step and the $E_{q(\mathcal{Y})}$-step messages of GEM-MP can be expressed in the form of one set of messages (from atoms-to-atoms through clauses). This set of messages synthesizes a general update rule for GEM-MP, applicable to Markov logic. However, since the underlying factor graph often contains hard and soft clauses, then within the GEM-MP framework we will intentionally distinguish hard and soft clauses by using two variants of the general update rule (denoted as *Hard-update-rule* and *Soft-update-rule*) for tackling hard and soft clauses, respectively.

4.1 hard update rule

For notational convenience, we explain the derivation of the hard update rule by considering untyped atoms; but extending it to the more general case is straightforward. Also, for clarity, we begin the derivation with the $M_{q(\mathcal{X})}$-step rather than with the usual $M_{q(\mathcal{Y})}$-step. So we assume that we have already constructed $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$. We also assume that all clauses are hard.

1. $M_{q(\mathcal{X})}$**-step:** Recalling Section 3, our basic goal in this step is to use $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ to estimate the marginals (i.e., parameters) $\mathcal{B}_{\mathcal{X}}$ that maximize the expected log-likelihood such that each $\beta_{X_j} \in \mathcal{B}_{\mathcal{X}}$ is a proper probability distribution. Thus we have an optimization problem of the form:[9]

$$\begin{aligned} &\max_{\mathcal{B}_{\mathcal{X}}} E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\big] \\ &\text{s.t. } \left(\beta^{+}_{X_j}+\beta^{-}_{X_j}\right)=1,\ \forall \beta_{X_j}\in\mathcal{B}_{\mathcal{X}} \end{aligned} \tag{35}$$

To perform this optimization, we first express it as the Lagrangian function $\Lambda(\mathcal{B}_{\mathcal{X}})$:

$$\Lambda(\mathcal{B}_{\mathcal{X}}) = E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\big] - T(\mathcal{B}_{\mathcal{X}}) \tag{36}$$

where $T(\mathcal{B}_{\mathcal{X}})$ is a constraint that ensures the marginals, $\mathcal{B}_{\mathcal{X}}$, are sound probability distributions. This constraint can be simply represented as follows:

$$T(\mathcal{B}_{\mathcal{X}}) = \sum_{X_j\in\mathcal{X}} \lambda_{X_j}\left(1-\beta^{+}_{X_j}-\beta^{-}_{X_j}\right) \tag{37}$$

Where λ_{X_j} are Lagrange multipliers that allow a penalty if the marginal distribution β_{X_j} does not marginalize to exactly one.

Now, let us turn to the derivation of the expected log-likelihood. We have that:

$$\begin{aligned} &E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\big] \\ &\qquad = \sum_{Y_i\in\mathcal{Y}} q(Y_i;\alpha_{Y_i}) \sum_{X_j\in\mathcal{X}} q(X_j;\beta_{X_j}) \log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M}) \end{aligned} \tag{38}$$

[9] Note that: $E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\big] \propto E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\big]$.

Based on the (hidden) variables $X_j \in \mathcal{X}$ and mega-nodes $Y_i \in \mathcal{Y}$, we can then decouple the distribution, $\log P(\mathcal{O}, \mathcal{Y}|\mathcal{X}, \mathcal{M})$, into individual distributions corresponding to hard ground clauses, and we have:[10]

$$E_{q(\mathcal{X};\mathcal{B}_\mathcal{X})q(\mathcal{Y};\mathcal{T}_\mathcal{Y})}\big[\log P(\mathcal{O}, \mathcal{Y}|\mathcal{X}, \mathcal{M})\big] \propto \sum_{X_j, Y_i} q(X_j; \beta_{X_j}) q(Y_i; \alpha_{Y_i}) \times \log\Big[\prod_{f_i^h \in \mathcal{F}^h} P(Y_i|\mathcal{X}, \mathcal{M})\Big] \quad (39)$$

Where $P(Y_i|\mathcal{X}, \mathcal{M})$ is the probability of randomly choosing a valid local entry in the mega-node Y_i given the marginal probabilities of the ground atoms, $\mathcal{B}_\mathcal{X}$. Now, we can proceed by decomposing $P(Y_i|\mathcal{X}, \mathcal{M})$ into individual marginals of ground atoms that possess consistent truth values in the valid local entries of Y_i. That is:

$$\log\Big[\prod_{f_i^h \in \mathcal{F}^h} P(Y_i|\mathcal{X}, \mathcal{M})\Big] \approx \log\Big[\prod_{f_i^h \in \mathcal{F}^h} \prod_{X_j \in \mathcal{X}_{f_i^h}} \beta_{X_j}(Y_i(X_j))\Big] \quad (40)$$

Where $\beta_{X_j}(Y_i(X_j))$ is the marginal probability of ground atom X_j at its consistent values with Y_i.

It is important to note that the decomposition in Eq. (40) is a mean field approximation for $P(Y_i|\mathcal{X}, \mathcal{M})$. It implies that the probability of valid local entries of Y_i for the ground clause f_i can be computed using individual marginals of the variables in the scope of f_i at their instantiations over such local entries. For instance, suppose that $f_i(X_1, X_2, X_3)$ is defined over three Boolean variables $\{X_1, X_2, X_3\}$ with marginal probabilities $\{\beta_{X_1}, \beta_{X_2}, \beta_{X_3}\}$. Now let $(0, 0, 1)$ be a valid local entry in the mega-node Y_i of f_i. To compute the probability $P(0, 0, 1|\beta_{X_1}, \beta_{X_2}, \beta_{X_3}, \mathcal{M})$, we can simply multiply the marginals of the three variables at their instantiations over this valid local entry as:

$$P(0, 0, 1|\beta_{X_1}, \beta_{X_2}, \beta_{X_3}, \mathcal{M}) = \beta^-_{X_1} \times \beta^-_{X_2} \times \beta^+_{X_3}$$

Where $\beta^-_{X_1}$, $\beta^-_{X_2}$ and $\beta^+_{X_3}$ are the marginal probabilities of X_1, X_2, and X_3 at values 0, 0, and 1 respectively.

We apply Eq. (40) in Eq. (39), convert logarithms of products into sums of logarithms, exchange summations, and handle each hard ground clause $f_i^h \in \mathcal{F}^h$ separately in a sum.

Subsequently we take the partial derivative of the Lagrangian function in Eq. (36) with respect to an individual ground atom positive marginal $\beta^+_{X_j}$ and

[10] Note that since we marginalize extended factors over their mega-nodes then it is sufficient to work directly with original factors in the original factor graph, which here are the hard ground clauses. In addition, we can eliminate the observed variable $\bar{O} = 1$ from $\log P(\mathcal{O}, \mathcal{Y}|\mathcal{X}, \mathcal{M})$ by explicitly considering only the valid local entries of the hard ground clauses.

equate it to zero:[11]

$$\frac{\partial}{\partial \beta^{+}_{X_j}}\left[\Lambda(\mathcal{B}_{\mathcal{X}})\right] = 0 \Rightarrow \beta^{+}_{X_j} = \frac{1}{\lambda_{X_j}} \underbrace{\left(\sum_{f^h_i \in \mathcal{F}^h_{X_j}} \overbrace{\sum_{Y_i : \forall y_k(X_j) = \text{“+”}} q(Y_i; \alpha_{Y_i})}^{(E_{q(\mathcal{Y})}\text{-step message}) \;\; \mu_{f_i \rightarrow X_j}} \right)}_{Weight^{+}_{X_j}} \tag{41}$$

where $\sum_{Y_i : \forall y_k(X_j) = \text{“+”}} q(Y_i; \alpha_{Y_i})$ is the $E_{q(\mathcal{Y})}$-step message that X_j will receive from each hard ground clause ($f^h_i \in \mathcal{F}^h_{X_j}$) conveying what it believes about X_j's positive marginal. Each $E_{q(\mathcal{Y})}$-step message is computed by adding a term for those valid local entries ($Y_i : \forall y_k(X_j) = \text{“+”}$) which instantiate the current hard ground clause using the positive value “+” for ground atom X_j.

Thus the sum of the $E_{q(\mathcal{Y})}$-step messages that ground atom X_j will receive from its neighboring hard ground clauses represents a weight (i.e., $Weight^{+}_{X_j}$) used to update its positive marginal.

Furthermore an analogous expression can be applied for a negative marginal $\beta^{-}_{X_j}$:

$$\frac{\partial}{\partial \beta^{-}_{X_j}}\left[\Lambda(\mathcal{B}_{\mathcal{X}})\right] = 0 \Rightarrow \beta^{-}_{X_j} = \frac{1}{\lambda_{X_j}} \underbrace{\left(\sum_{f^h_i \in \mathcal{F}^h_{X_j}} \overbrace{\sum_{Y_i : \forall y_k(X_j) = \text{“−”}} q(Y_i; \alpha_{Y_i})}^{(E_{q(\mathcal{Y})}\text{-step message}) \;\; \mu_{f_i(Y_i) \rightarrow X_j}} \right)}_{Weight^{-}_{X_j}} \tag{42}$$

Finally we now move to solving λ_{X_j} as follows:

$$\begin{gathered} \beta^{-}_{X_j} + \beta^{+}_{X_j} = 1 \overset{From\ Eqs.\ (41)\ and\ (42)}{\Rightarrow} \\ \left[\left(Weight^{+}_{X_j} + Weight^{-}_{X_j}\right)/\lambda_{X_j} = 1\right] \Rightarrow \lambda_{X_j} = \left(Weight^{+}_{X_j} + Weight^{-}_{X_j}\right) \end{gathered} \tag{43}$$

which shows that λ_{X_j} serves as a normalizing constant that converts such weights (i.e., $Weight^{+}_{X_j}$, and $Weight^{-}_{X_j}$) into a sound marginal probability (i.e. $\beta_{X_j} = \left[\beta^{-}_{X_j}, \beta^{+}_{X_j}\right]$).

Now to obtain the completed hard update rule, what remains is the $M_{q(\mathcal{Y})}$-step, through which we need to substitute the distribution $q(Y_i; \alpha_{Y_i})$ in Eqs. (41) and (42).

2. $M_{q(\mathcal{Y})}$**-step:** The goal here is to produce the distribution $q(Y_i; \alpha_{Y_i})$ by using the current setting of marginals $\mathcal{B}_{\mathcal{X}}$. However the summation $\sum_{Y_i : \forall y_k(X_j) = \text{“−”}}$ involves enumerating all the valid local entries for each Y_i, which is inefficient. Instead we approximate the distribution $\sum_{Y_i : \forall y_k(X_j) = \text{“−”}} q(Y_i; \alpha_{Y_i})$ for each hard ground clause $f^h_i \in \mathcal{F}^h$ by using a probability $1 - \xi(X_j, f^h_i)$, which we call the *probabilistic generalized arc consistency* (pGAC). At this point, let us pause to elaborate more on pGAC in the next subsection.

[11] This implies taking the derivative of both expected log-likelihoods, which we obtain after substituting the value of Eq. (40) for Eq. (39), and the penalty term in Eq. (37).

4.1.1 Note on the connection between pGAC and variational inference

According to the concept of generalized arc consistency, a necessary (but not sufficient) condition for a ground atom X_j to be assigned a value $d \in \{+, -\}$, is for every other ground atom appearing in the ground clause f_i to be individually consistent in the support of this assignment, i.e., $X_j = d$. Without loss of generality, suppose that X_j appears positively in f_i: there is a probability that $X_j = d$ is not generalized arc consistent with respect to f_i when those other ground atoms appearing in f_i are individually inconsistent with this assignment since $X_j = d$ can belong to an invalid local entry of f_i. This means that there is a probability that $X_j = d$ is unsatisfiable with respect to f_i when all other ground atoms appearing in f_i are set unsatisfyingly. We use $\xi(X_j, f_i)$ to denote this probability, we assume Independence and approximate it as:

$$\xi(X_j, f_i) = \left(\prod_{X_k \in \mathcal{X}_{f_i^+} \setminus \{X_j\}} \left(1 - \beta^+_{X_k}\right) \cdot \prod_{X_k \in \mathcal{X}_{f_i^-} \setminus \{X_j\}} \left(\beta^+_{X_k}\right) \right) \tag{44}$$

As indicated in Eq. (44), $\xi(X_j, f_i)$ is computed by iterating through all the other ground atoms in clause f_i and consulting their marginals toward the opposite truth value of their appearance in f_i. In other words, the $\xi(X_j, f_i)$ forms a product representing the probability that, except X_j, all other ground atoms $\mathcal{X}_{f_i} \setminus \{X_j\}$ in f_i taking on particular values that constitute invalid local entries to f_i. Such invalid local entries support X_j unsatisfying f_i and can be approximated based on the marginal distributions of those ground atoms (i.e., $\mathcal{X}_{f_i} \setminus \{X_j\}$) at these particular values. It should be noted that f_i has those marginal distributions from the incoming $E_{q(\mathcal{X})}$-step messages that are sent from its argument ground atoms $\mathcal{X}_{f_i}$ during the GEM-MP's $M_{q(\mathcal{Y})}$-step.

Hence, if $\xi(X_j, f_i)$ is the probability of $X_j = d$ unsatisfying f_i then $1-\xi(X_j, f_i)$ is directly the probability of $X_j = d$ satisfying the ground clause f_i. It also represents the probability that $X_j = d$ is GAC with respect to f_i because the event of $X_j = d$ satisfying f_i implies that it must be GAC to f_i. This interpretation entails a form of generalized arc consistency, adapted to CNF, in a probabilistic sense; we call it a *Probabilistic Generalized Arc Consistency*.

Definition 2 (Probabilistic Generalized Arc Consistency (pGAC))
Given a ground clause $f_i \in \mathcal{F}$ defined over ground atoms $\mathcal{X}_{f_i}$, and for every $X_j \in \mathcal{X}_{f_i}$, let $D_{X_j} = \{+, -\}$ be the domain of X_j. A ground atom X_j assigned a truth value $d \in D_{X_j}$ is said to be probabilistically generalized arc consistent (pGAC) to ground clause f_i if the probability of $X_j = d$ belonging to a valid local entry of f_i is non-zero. That is to say, if there is a non-zero probability that $X_j = d$ is GAC to f_i. The pGAC probability of $X_j = d$ can be approximated as:

$$0 < 1 - \xi(X_j, f_i) \leq 1 \tag{45}$$

The definition of the traditional GAC in Section 2 corresponds to the particular case of pGAC where $\xi(X_j, f_i) = 0$, meaning that the probability of $X_j = d$ being GAC to f_i definitely occurs, and $\xi(X_j, f_i) = 1$ when it is never GAC to f_i. Based on that, if f_i contains X_j positively then the pGAC probability of $X_j = +$ equals

1 because it is always GAC to f_i. In an analogous way, the pGAC probability is 1 for $X_j = -$ when f_i contains X_j negatively.

From a probabilistic perspective, the pGAC probability of $X_j = d$ represents the probability that $X_j = d$ is involved in a valid local entry of f_i. This is similar to the computation of the solution probability of $X_j = d$ by using the probabilistic arc consistency (pAC) (presented by Horsch and Havens, 2000, and summarized in Section 2). However, it should be noted that our pGAC applies mean-field approximation. This is because when computing $\xi(X_j, f_i)$, as defined in Eq. (44), for each ground atom $X_j \in \mathcal{X}_{f_i}$, we use the marginal probabilities of other ground atoms $X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}$ set unsatisfying in f_i. Thus the main difference between our pGAC and pAC (Horsch and Havens, 2000) appears in the usage of mean-field and BP for computing the probability that $X_j = d$ belongs to valid local entry of f_i in pGAC and pAC, respectively. Furthermore, it should be noted that pAC is restricted to binary constraints whilst pGAC is additionally applicable to non-binary ones.

From the point of view of computational complexity, $\xi(X_j, f_i)$ requires only linear computational time in the arity of the ground clause (as will be shown in Proposition 3). Thus, pGAC is an efficient form of GAC compared to pAC. In addition, pGAC guarantees the convergence of mean-field whereas pAC inherits the possibility of non-convergence from BP.

From a statistical perspective, the pGAC probability of $X_j = +$ is a closed form approximation for a sample from the valid local entries of f_i that involve $X_j = +$. Thus we have that:

$$\big[1 - \xi(X_j, f_i)\big] \Big|_{X_j = \text{“+”}} \propto \sum_{Y_i : \forall y_k (X_j) = \text{“+”}} q(Y_i; \alpha_{Y_i}) \tag{46}$$

And similarly the pGAC probability for $X_j = -$:

$$\big[1 - \xi(X_j, f_i)\big] \Big|_{X_j = \text{“−”}} \propto \sum_{Y_i : \forall y_k (X_j) = \text{“−”}} q(Y_i; \alpha_{Y_i}) \tag{47}$$

Based on Eqs. (46) and (47), we can use pGAC for computing the two components of $E_{q(\mathcal{Y})}$-step message, in Eqs. (41)and (42), that f_i sends to X_j as follows:

- $[1, 1 - \xi(X_j, f_i)]$ if f_i contains X_j positively.
- $[1 - \xi(X_j, f_i), 1]$ if f_i contains X_j negatively.

Note that computing the components of f_i's $E_{q(\mathcal{Y})}$-step message in this way above requires having in hand the marginals of all other ground atoms, $X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}$. Thus, one of the best choices is to simultaneously passing the $E_{q(\mathcal{X})}$-step messages – which convey the marginals – from ground atoms $\mathcal{X}_{f_i}$ to ground clause f_i. Additionally at f_i's level we can sequentially update the marginals as: obtain the marginal of the first ground atom then use its new marginal in the updating process of the second atom's marginal, then use the first and second atoms' new marginals in the updating process of the third atom's marginal, and so on. This sequential updating allows GEM-MP to use the latest available information of the marginals through the updating process. In addition, doing so enables a single update rule that performs both the E- and M- steps at the same time, by directly representing the $M_{q(\mathcal{Y})}$-step within the rule we derived for the $M_{q(\mathcal{X})}$-step.

4.1.2 Using pGAC in the derivation of the hard update rule

We now continue the derivation of the hard update rule by using pGAC to address the task of producing $\sum_{Y_i:\forall y_k(X_j)} q(Y_i;\alpha_{Y_i})$ in Eqs. (41) and (42) as follows:

$$Weight^{+}_{X_j} = \sum_{f^h_i \in \mathcal{F}^h_{X_j}} \sum_{Y_i:\forall y_k(X_j)=\text{“+”}} q(Y_i;\alpha_{Y_i}) \tag{48a}$$

$$= \Bigg[\sum_{f^h_i \in \mathcal{F}^h_{X_j+}} \sum_{Y_i:\forall y_k(X_j)=\text{“+”}} q(Y_i;\alpha_{Y_i})\Bigg] + \Bigg[\sum_{f^h_i \in \mathcal{F}^h_{X_j-}} \sum_{Y_i:\forall y_k(X_j)=\text{“+”}} q(Y_i;\alpha_{Y_i})\Bigg] \tag{48b}$$

$$\approx \sum_{f^h_i \in \mathcal{F}^h_{X_j+}} [1] + \sum_{f^h_i \in \mathcal{F}^h_{X_j-}} (1-\xi(X_j,f^h_i)) \tag{48c}$$

$$= \left|\mathcal{F}^h_{X_j}\right| - \sum_{f^h_i \in \mathcal{F}^h_{X_j-}} \xi(X_j,f^h_i) \tag{48d}$$

where in Eq. (48b) we first separate the summation into X_j's positive and negative hard ground clauses to consider the two distinct situations of whether X_j appears as a positive ground atom versus the other situation where it appears as a negative ground atom. Further in Eq. (48c) in the first positive summation, we replaced the inner summation with the constant 1 (because all other atoms will be generalized arc consistent with $X_j =$ “ + ” for the hard clauses that have a positive appearance of X_j - as explained in subsection 4.1.1).

The end result, as in Eq. (48d), is the $Weight^{+}_{X_j}$ of ground atom X_j computed as the summation of all hard ground clauses that include X_j minus the summation of pGAC of hard ground clauses that involve X_j as a negative atom.

The interpretation of $Weight^{+}_{X_j}$ can be understood as reducing the positive probability of X_j according to the expectation of the probability that X_j is needed by its negative hard ground clauses. Such reductions are taken from a constant that represents the overall number of hard ground clauses that involve X_j (i.e. $\left|\mathcal{F}^h_{X_j}\right|$). Similarly we can obtain:

$$Weight^{-}_{X_j} = \left|\mathcal{F}^h_{X_j}\right| - \sum_{f^h_i \in \mathcal{F}^h_{X_j+}} \xi(X_j,f^h_i) \tag{49}$$

where $Weight^{-}_{X_j}$ has an analogous interpretation of $Weight^{+}_{X_j}$ for the negative probability of X_j.

4.2 soft update rule

To derive the update rule for soft ground clauses, what we need to do is to soften some restrictions on the weight parts (i.e. $Weight^{+}_{X_j}$, $Weight^{-}_{X_j}$) of the hard update rule. This encompasses modifying the distributions, $q(Y_i;\alpha_{Y_i})$, of hard ground clauses for soft ground clauses by applying two consecutive steps: *softening* and *embedding*.

For clarity, let us recall the example of the extended factor graph shown in Figure 2(*right*). In the softening step, we define the variational parameters α_i, of the distributions $q(Y_i;\alpha_{Y_i})$, that are appended to the soft clauses to be different from those appended to hard clauses in a way that renders them suitable to the semantics of soft ground clauses. That is, we discriminate variational parameters of distributions $q(Y_i;\alpha_{Y_i})$ for hard and soft ground clauses respectively as follows:

$$\begin{aligned}\alpha_{Y_i}\left(f_i^h\right) &= \begin{cases}1 & \text{if the state of } Y_i \text{ satisfies } f_i^h \\ 0 & \text{Otherwise.}\end{cases}, \\ \alpha_{Y_i}\left(f_i^s\right) &= \begin{cases}\exp(w_{f_i^s}) & \text{if the state of } Y_i \text{ satisfies } f_i^s, \\ 1 & \text{Otherwise.}\end{cases}\end{aligned} \tag{50}$$

where $w_{f_i^s}$ is the numeric weight associated with soft ground clause f_i^s. Now, the use of variational parameters $\alpha_{Y_i}\left(f_i^s\right)$ (instead of $\alpha_{Y_i}\left(f_i^h\right)$) for the hard update rule in Eq. (48d) implies taking the exponential transformation as follows:

$$\overset{Softening}{\Rightarrow} \beta_{X_j}^{+} = \frac{1}{\lambda_{X_j}}\left(\sum_{Y_i:\forall y_k(X_j)=\text{“+”}} \frac{\exp\left[\sum_{f_i^h\in\mathcal{F}_{X_j}^h} q(Y_i;\alpha_{Y_i}(f_i^h))\right]}{\prod_{f_i^s\in\mathcal{F}_{X_j}^s}\exp\left(q(Y_i;\alpha_{Y_i}(f_i^h))\right)}\right) \tag{51}$$

Note that $\exp\left[\sum_{f_i^h\in\mathcal{F}_{X_j}^h} q(Y_i;\alpha_{Y_i}(f_i^h))\right]$ is converted simply to:

$$\prod_{f_i^s\in\mathcal{F}_{X_j}^s}\exp\left(q(Y_i;\alpha_{Y_i}(f_i^h))\right)$$

where

$$\exp\left(q(Y_i;\alpha_{Y_i}(f_i^h))\right) \approx q(Y_i;\alpha_{Y_i}(f_i^s)).$$

Accordingly in the embedding step we embed the support of invalid local entries. This is because at the dissatisfaction of soft ground clauses we get 1 instead of 0 at the dissatisfaction of hard ground clauses. Thus, we discard the summation over valid local entries (i.e., remove $\sum_{Y_i:\forall y_k(X_j)=\text{“+”}}$ in Eq. (51)) and instead we consider the support of both valid local entries (weighted by $exp(w_{f_i^s})$) and invalid local entries (weighted by 1), ending up with:

$$\overset{Embedding}{\Rightarrow} \beta_{X_j}^{+} = \frac{1}{\lambda_{X_j}}\underbrace{\left(\prod_{f_i^s\in\mathcal{F}_{X_j}^s} q\left(Y_i;\alpha_{Y_i}(f_i^s)\right)\right)}_{Weight_{X_j}^{+}} \tag{52}$$

Likewise adhering to the derivation of the hard update rule, we can obtain the local approximation of $Weight^{+}_{X_j}$ part for the soft update rule as:

$$Weight^{+}_{X_j} = \prod_{f^s_i \in \mathcal{F}^s_{X_j}} q\left(Y_i; \alpha_{Y_i}(f^s_i)\right) \tag{53a}$$

$$= \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j+}} q\left(Y_i; \alpha_{Y_i}(f^s_i)\right)\right] \times \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j-}} q\left(Y_i; \alpha_{Y_i}(f^s_i)\right)\right] \tag{53b}$$

$$\approx \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j+}} \exp(w_{f^s_i})[1]\right] \times \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j-}} \left[(1-\xi(X_j, f^s_i))\exp(w_{f^s_i}) + \xi(X_j, f^s_i)\cdot 1\right]\right] \tag{53c}$$

$$= \left[\exp\Big(\sum_{f^s_i \in \mathcal{F}^s_{X_j}} w_{f^s_i}\Big)\right] - \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j+}} \exp(w_{f^s_i}) \times \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j-}} \xi(X_j, f^s_i)(\exp(w_{f^s_i}) - 1)\right]\right] \tag{53d}$$

Note that comparing Eq. (53c) to its corresponding Eq. (48c) for the update rules of the hard factors, we have an additional term "$\xi(X_j, f^s_i)\cdot 1$" in the second summation. This is because computing the second part of Eq. (53b) implies computing two terms as appeared in the second part of Eq. (53c): the first is $(1-\xi(X_j, f^s_i))$ representing the probability that X_j being positive satisfies the factor f^s_i that include X_j as negative ground atom, and therefore it is multiplied by $\exp(w_{f^s_i})$ since at the satisfaction of soft ground clause f^s_i we obtain $\exp(w_{f^s_i})$. The second term is $\xi(X_j, f^s_i)$ representing the probability that X_j being positive dissatisfies the factor f^s_i, and therefore it is multiplied by 1 since at the dissatisfaction of f^s_i we obtain 1. This is the "$\xi(X_j, f^s_i)\cdot 1$" term that has disappeared from the update rules of the hard factors in Eq. (48c) because $\xi(X_j, f^s_i)$ is multiplied by 0, since at the dissatisfaction of hard ground clauses we get 0 instead of 1 for the dissatisfaction of soft ground clauses.

Similarly, we can obtain the negative weight part $Weight^{-}_{X_j}$ for the soft update rule as:

$$Weight^{-}_{X_j} = \left[\exp\Big(\sum_{f^s_i \in \mathcal{F}^s_{X_j}} w_{f^s_i}\Big)\right] - \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j-}} \exp(w_{f^s_i}) \times \left[\prod_{f^s_i \in \mathcal{F}^s_{X_j+}} \xi(X_j, f^s_i)(\exp(w_{f^s_i}) - 1)\right]\right] \tag{54}$$

Note that the weight parts (in Eqs. (53d) and (54)) used for the soft update rule, are soft versions of previously derived weight parts (in Eqs. (48d) and (49)) used for the hard update rule. Therefore at a high level they have similar interpretations.

At this point, we take the $Weight^{+}_{X_j}$ and $Weight^{-}_{X_j}$ from Eqs. (48d), (49), (53d), and (54) and substitute these for the $Weight^{+}_{X_j}$ and $Weight^{-}_{X_j}$ in Eqs. (41) and (42) to obtain our ultimate set of GEM-MP's rules in order to update the marginals of query ground atoms. This is in Table 3. The main advantage of these update rules is that they capture relationships between ground atoms with each other. Thus, we do not need to pass explicitly the messages from atoms-to-clauses or vice versa.

Note that, on one hand, using a single update rule for updating the marginals is beneficial for the simplicity of implementation. However, on the other hand, using other scheduling than the one used here for the GEM-MP framework requires rederiving GEM-MP's equations to obtain other single update rules that are adopted with the new scheduling, or do not use single update rules and pass explicitly the $M_{q(\mathcal{Y})}$-step and $E_{q(\mathcal{Y})}$-step messages from variables-to-factors and factors-to-variables, respectively.

Table 3 General update rules of GEM-MP inference for Markov logic. These rules capture relationships between ground atoms with each other, and therefore it does not require explicitly passing messages between atoms and clauses.

$\beta^{+}_{X_j} = \frac{Weight^{+}_{X_j}}{\lambda_{X_j}},\ \beta^{-}_{X_j} = \frac{Weight^{-}_{X_j}}{\lambda_{X_j}},\ \lambda_{X_j} = Weight^{+}_{X_j} + Weight^{-}_{X_j}$
Hard-update-rule
$Weight^{+}_{X_j} \leftarrow \left\lvert \mathcal{F}^{h}_{X_j} \right\rvert - \sum_{f^{h}_{i} \in \mathcal{F}^{h}_{X_j -}} \xi(X_j, f^{h}_{i})$
$Weight^{-}_{X_j} \leftarrow \left\lvert \mathcal{F}^{h}_{X_j} \right\rvert - \sum_{f^{h}_{i} \in \mathcal{F}^{h}_{X_j +}} \xi(X_j, f^{h}_{i})$
Soft-update-rule
$Weight^{+}_{X_j} \leftarrow \left[\exp\left(\sum_{f^{s}_{i} \in \mathcal{F}^{s}_{X_j}} w_{f^{s}_{i}}\right)\right] - \left[\prod_{f^{s}_{i} \in \mathcal{F}^{s}_{X_j +}} \exp(w_{f^{s}_{i}}) \times \left[\prod_{f^{s}_{i} \in \mathcal{F}^{s}_{X_j -}} \xi(X_j, f^{s}_{i})(\exp(w_{f^{s}_{i}}) - 1)\right]\right]$
$Weight^{-}_{X_j} \leftarrow \left[\exp\left(\sum_{f^{s}_{i} \in \mathcal{F}^{s}_{X_j}} w_{f^{s}_{i}}\right)\right] - \left[\prod_{f^{s}_{i} \in \mathcal{F}^{s}_{X_j -}} \exp(w_{f^{s}_{i}}) \times \left[\prod_{f^{s}_{i} \in \mathcal{F}^{s}_{X_j +}} \xi(X_j, f^{s}_{i})(\exp(w_{f^{s}_{i}}) - 1)\right]\right]$

4.3 GEM-MP versus LBP

One might contrast GEM-MP and LBP inference. Recall the basic quantities used by GEM-MP in Eqs. (41) and (42) vs. LBP in Eqs. (3) and (4) for updating the marginal of a single variable X_j. Although the marginal update rules of both

algorithms look similar, they are constructed by very different routes, having important differences. The first significant difference is that due to the expectations involved in variational message passing, in GEM-MP we take a summation (i.e. $\sum_{f_i \in \mathcal{F}_{X_j}}$) over the incoming messages to a given node, which are the outgoing messages coming from the factors. This is in contrast to the multiplication (i.e. $\prod_{f_i \in \mathcal{F}_{X_j}}$) associated with standard LBP. In other words GEM-MP handles the incoming message (or, as named, $E_{q(\mathcal{Y})}$-step message) from each factor as a separate term in a sum. This means that when moving toward the local maximum of energy functional $\mathcal{F}_{\mathcal{M}}$ in Eq. (17c), GEM-MP computes a moderate arithmetic average of the incoming $E_{q(\mathcal{Y})}$-step messages to yield the marginal update steps for X_j. Due to the variational underpinnings of GEM-MP these steps update a quantity that is a lower bound on the log marginal likelihood. This is attributable to the use of Jensen's inequality in Eq. (12c) that allows lower bounding the model evidence, and at each update step we minimize the Kullback-Leibler divergence distance. We therefore cannot 'overstep' in our approximation of the true model evidence (refer to Theorem 1). In contrast, LBP computes a (coarse) geometric average of the incoming messages in a setting where there is no such bound.

The second important difference between these algorithms is how they compute their "outgoing messages" from factors to variables based on the previous iteration's incoming messages from variables to factors. In LBP the outgoing message is a partial sum over the product of the factor's probability distribution by its incoming messages from other neighboring variables which naturally arises from the original exact computations which easily fall out of the computations for correctly marginalizing a tree-structured graphical model. However the operations of simply multiplying then taking partial sums do nothing to exploit any local structure of the underlying factor. In contrast GEM-MP leverages the fact that factors (e.g., in Markov logic and Ising models) are represented as logical clauses, and therefore we can take advantage of generalized arc consistency to cleverly convey the local structures' semantics into their outgoing messages. Strictly speaking GEM-MP's outgoing $E_{q(\mathcal{Y})}$-step message is an approximate marginal distribution $q(Y_i; \alpha_{Y_i})$ over the valid local entries $Y_i : \forall y_k(X_j)$ in which the X_j (that will receive the message) is GAC with other variables in the factor; we approximate this distribution by computing the pGAC of X_j using the marginals of other variables in the factor that are GAC with X_j (refer to subsection 4.1.1). This means that the outgoing $E_{q(\mathcal{Y})}$-step message that will be received by X_j ensures that its marginal should be consistent with the marginals of other variables according to the local structure's semantic of the factor. This improves the process of convincing the variables to converge correctly. Hence exploiting the logical structures by pGAC when computing the outgoing messages of factors is what we believe helps GEM-MP alleviate the problems associated with determinism.

4.4 GEM-MP Algorithm

Algorithm 1 gives a pseudo-code for the GEM-MP inference algorithm. The algorithm starts by uniformly initializing (i.e., $\mathcal{U}$) the marginals of all ground atoms that exist in the query set $\mathcal{X}$ (lines 1-3). Then, it distinguishes two subsets of query ground atoms. The first is $\mathcal{X}_h$ that involves query ground atoms involved

Algorithm 1 The GEM-MP inference algorithm for Markov logic.

Input: Clauses $\mathcal{F}$, Ground queries $\mathcal{X}$, Maximum number of iterations $\mathcal{I}_{\max}$.
Output: Marginals $\mathcal{B}_{\mathcal{X}}$.

```
   // Initialization
 1: for each X_j ∈ 𝒳 do
 2:    β_{X_j} ← 𝒰[0,1];
 3: end for
   // discriminate query atoms.
 4: 𝒳_h ← X_j ∈ 𝒳; // involved in hard ground clauses ℱ^h ∈ ℱ.
 5: 𝒳_s ← X_k ∈ 𝒳; // involved in soft ground clauses ℱ^s ∈ ℱ.
   // inferring marginals
 6: repeat
 7:    for each X_j ∈ 𝒳_h do
 8:       β_{X_j} ← Hard-Update-Rule; // as in Table 3
 9:    end for
10:    for each  X_k ∈ 𝒳_s do
11:       β_{X_k} ← Soft-Update-Rule; // as in Table 3
12:    end for
13: until convergence or termination of ℐ_max
14: Return ℬ_𝒳;
```

in hard ground clauses (line 4). The second subset is $\mathcal{X}_s$ for the ones involved in soft ground clauses (line 5). Note that if the query atom is involved in both soft and hard ground clauses, then it will be included in the two subsets. At each step, the algorithm proceeds by updating the marginals for the first subset of query atoms by using the hard update rule (lines 7-9). Then it updates the marginals for query atoms of the second subset by applying the soft update rule (lines 10-12). The algorithm keeps alternating between carrying out the two update-rules until convergence (i.e., $\forall X_j \in \mathcal{X},\ \left|\beta_{X_j}(\mathcal{I}) - \beta_{X_j}(\mathcal{I}-1)\right| < \epsilon$, where ϵ is a specified precision) or reaching the maximum number of iterations (line 13). Although the marginals of the query atoms involved by soft and hard ground clauses (i.e., exist in the two subsets $\mathcal{X}_h$ and $\mathcal{X}_s$) may be affected by swapping from hard to soft update rules, or vice versa, such query atoms' marginals play the role of propagating the information about hard ground clauses to query atoms in $\mathcal{X}_s$ when it is used by the soft update rule, and propagating the information about soft ground clauses to query atoms in $\mathcal{X}_h$ when it is used by the hard update rule. It should be noted that the checks performed by each update-rule are extremely cheap (a fraction of a second, on average) and the subset of ground clauses at each particular step is unlikely to be in the hard critical region.

Proposition 3 (Computational Complexity) *Given an MLN's ground network with n ground atoms, m ground clauses, and a maximum arity of the ground clauses of r, one iteration of computing the marginals of query atoms takes time in $O(nmr)$ in the worst case.*

Proof see Appendix A. □

Note that even though GEM-MP is built on a propositional basis, its computational complexity is quite efficient since the size of the grounded network is proportional to $O(d^r)$, where d is the number of objects (constants) in the domain. Also, in practice, we can improve this computational time by preconditioning some terms. For instance, we do not compute the constant terms (such as $\left|\mathcal{F}^h_{X_j}\right|$ in the

hard update rule) at each iteration, but instead we compute them once and then recall their values.

5 GEM-MP Update Rules for Ising MRFs

In this section we demonstrate how to easily adapt the GEM-MP algorithm to handle inference in the presence of determinism over other typical probabilistic graphical models, rather than over Markov Logic Networks. For simplicity, let us here consider Ising models with arbitrary topology which are a specific subset of the canonical (pairwise) Markov random fields (MRFs)- undirected graphical models that compactly represent a joint distribution over $\mathcal{X}$ by assuming that it is obtained as a product of potentials defined on subsets of variables (see Koller and Friedman, 2009). Although pairwise Markov random fields are commonly used as a benchmark for inference because they have a simple and compact representation, they often pose a challenge for inference. Now assume that $\mathcal{X} = \{X_1, \ldots, X_n\}$ is a set of binary random variables that are Bernoulli distributed. In an Ising model $\mathcal{I} = ((\mathcal{X}, \mathcal{E}); \theta)$ we have an undirected graph consisting of the set of all variables $\mathcal{X}$, a set of edges between variables $\mathcal{E}$, and a set of parameters $\theta = \{\theta_i, \theta_{ij}\}$. The model can be then given as

$$p(\mathcal{X} = x) = \mathcal{Z}_\theta^{-1} \overbrace{e^{\left[\sum_{(X_i,X_j)\in\mathcal{E}} \theta_{ij} \cdot X_i X_j + \sum_{X_i \in \mathcal{X}} \theta_i \cdot X_i\right]}}^{\text{energy function}} \tag{55a}$$

$$= \mathcal{Z}_\theta^{-1} \left[\prod_{(X_i,X_j)\in\mathcal{E}} \overbrace{e^{\theta_{ij} \cdot \mathbb{1}_{(X_i,X_j)}}}^{\phi_{ij}(X_i,X_j)} \right] \times \left[\prod_{X_i \in \mathcal{X}} \overbrace{e^{\theta_i \cdot X_i}}^{\phi_i(X_i)} \right] \tag{55b}$$

where $\{\theta_i\}$ and $\{\theta_{ij}\}$ are the parameters of uni-variate potentials $\{\phi_i(X_i)\}$ and pairwise potentials $\{\phi_{ij}(X_i, X_j)\}$, respectively. Typically it is convenient to represent Ising models in Eq. (55b) as factor graphs, where variables $X_i \in \mathcal{X}$ represented as variable nodes and potentials $\{\phi_i(X_i)\}$ and $\{\phi_{ij}(X_i, X_j)\}$ represented as factor nodes. Traditionally, the parameters $\{\theta_i\}$ are drawn uniformly from $\mathcal{U}[-d_f, d_f]$ where $d_f \in \mathbb{R}$. For pairwise potentials, the parameters $\{\theta_{ij}\}$ are chosen as $\eta \cdot C$ where we sample η in the range $[-d_f, d_f]$ having some nodes to agree and disagree with each other. C is also a chosen constant. Higher values of C impose stronger constraints, leading to a harder inference task.

Hence each univariate potential $\phi_i(X_i)$ can be represented as a unit clause involving only one variable X_i with associated weight θ_i such that it equals e^{θ_i} when it is satisfied and 1 otherwise. Similarly each $\phi_{ij}(X_i, X_j)$ can be formulated as a conjunction of two clauses[12] $\big[(\neg X_i \vee X_j) \wedge (X_i \vee \neg X_j)\big]$, with associated weight $\eta \cdot C$ which equals $e^{\eta \cdot C}$ when $X_i = X_j$ and $e^{-\eta \cdot C}$ otherwise. Hence, the Ising model can be translated into CNF as:

- Unit clauses: (X_i, θ_i), $\forall X_i \in \mathcal{X}$
- Pairwise clauses: $\big[(\neg X_i \vee X_j) \wedge (X_i \vee \neg X_j), \theta_{ij} = \eta \cdot C\big]$, $\forall X_i, X_j \in \mathcal{E}$

[12] Note that $l_1 \Leftrightarrow l_2$ converted into CNF gives two clauses : $(\neg l_1 \vee l_2) \wedge (l_1 \vee \neg l_2)$

Now, without difficulty, we can directly apply the soft update rule from Table 3 for such clauses when computing the marginals on the factor graph. Now assume that we want to present some determinism in the model. We can achieve that by adjusting the parameters in such a way that makes either univariate or pairwise potential produce 0 when it is unsatisfied. For instance, if C is very large (say $C \to \infty$) in the setting of parameters θ_{ij} we obtain that all the valid local entries of $\phi_{ij}(X_i, X_j)$'s clauses equal e^{∞} and all its invalid local entries equal 0 (i.e., $e^{-\infty}$), which can be simply re-cast as $\{0, 1\}$ clauses. Thus in this case we can apply the hard update rule from Table 3 when computing the marginals.

6 Empirical Evaluation

The goal of our experimental evaluation was to investigate the following key questions:

- (**Q1**.) Is GEM-MP's accuracy competitive with state-of-the-art inference algorithms for Markov logic? *This question is important to answer as it examines the soundness of GEM-MP inference.*
- (**Q2**.) In the presence of graphs with problematic cycles, comparing with LBP exhibiting oscillations, does GEM-MP lead to convergence? *We want to explore and emphasize experimentally that GEM-MP inference indeed addresses Limitation 1.*
- (**Q3**.) Is GEM-MP more accurate than LBP in the presence of determinism? *We want to check experimentally the effectiveness of GEM-MP inference to remedy Limitation 2.*
- (**Q4**.) Is GEM-MP scalable compared to other state-of-the-art propositional inference algorithms for Markov logic? *We wish to examine the real-world applicability of GEM-MP inference.*
- (**Q5**.) Is GEM-MP accurate compared to state-of-the-art convergent message-passing algorithms for other probabilistic graphical models such as Markov Random Fields? *We wish to examine the accuracy and convergence behaviour of GEM-MP inference for other related model classes and algorithms.*
- (**Q6**.) Is GEM-MP's accuracy influenced by the initialization of the marginals? *We will examine if the initialization of approximate marginals using random values differs from initializing marginals using a uniform distribution.*

To answer Questions Q1 to Q4, we first selected three real-world datasets: *Cora* for Entity resolution, *Yeast* for Protein-interactions, and *UW-CSE* for Advising relationships. Such datasets[13] and their corresponding MLN formulations contain the problematic properties of determinism and cycles and therefore represent good bases for carrying out our experimental evaluations. The first point to note is that their expressive Markov logic networks have a formidable number of cycles. Besides this, some of their rules can be expressed as hard formulas. Thus, it is highly anticipated that the inference procedure will face the hindrances engendered from determinism and cycles. The second point is that they exemplify important applications: Entity resolution has recently become somewhat of a holy grail sort of

[13] Publicly available: `http://alchemy.cs.washington.edu/data/`

task; Advising relationships and Protein-interactions are instances of Link prediction, an important task that always receives much interest in statistical relational learning (Richardson and Domingos, 2006).

To evaluate our proposed GEM-MP inference algorithm, we compared its results with five prominent state-of-the-art inference algorithms[14] that are built into the Alchemy system (Kok et al., 2007), one of the most powerful tools to perform inference on Markov logic models:[15]

- **MC-SAT** proposed by Poon and Domingos (2006).
- Lazy MC-SAT (**LMCSAT**) proposed by Poon et al. (2008).
- Loopy Belief Propagation (**LBP**) (refer to Yedidia et al., 2005).
- Gibbs sampling (**Gibbs**) (see Richardson and Domingos, 2006).
- Lifted Importance sampling (**L-Im**) proposed by Venugopal and Gogate (2014b) as an improvement of the one proposed by Gogate et al. (2012).

MC-SAT converges rapidly when performing inference in the presence of determinism and L-Im is the recent lifted importance sampling inference algorithm that addresses the evidence problem (see Venugopal and Gogate, 2014a) and as a result improves the scalability and accuracy of reasoning. Therefore to answer Q1 and Q4, our main comparison is with MC-SAT and L-Im. Since our GEM-MP algorithm is a variant of message-passing inference, we shall compare with LBP to answer Q1, Q2, Q3, and Q4. Gibbs, a popular MCMC algorithm, can serve as a good baseline here. Additionally, even though GEM-MP is built on a propositional basis, it may be suitable to compare its scalability with two state-of-the-art approaches for scaling inference such as Lifted in the L-Import algorithm and Lazy in the LMCSAT algorithm. Note that a few other efficient inference methods are not considered in our experiments because they are completely dominated by one of the three considered algorithms (e.g., simulated tempering had shown poor results compared to MC-SAT, as shown by Poon and Domingos (2006)), or they run exact inference (like PTP introduced by Gogate and Domingos (2011)), which is out of reach for the underlying datasets.

6.1 Datasets

Cora. This dataset consists of 1295 citations of 132 different computer science papers.[16]

- **MLN:** We used the MLN model which is similar to the established one of Singla and Domingos (2006). The MLN involves formulas stating regularities such as: if two citations are the same, their fields are the same; if two fields are the same, their citations are the same. It also has formulas representing transitive closure, which are assigned very high weight (i.e. near deterministic clauses). The final knowledge base contains 10 atoms and 32 formulas (adjusted as 4 hard, 3 near-deterministic and 25 soft).

[14] These algorithms run on the original factor graph $\mathcal{G}$.

[15] Alchemy v0.2 is publicly available at: `http://alchemy.cs.washington.edu/`

[16] Primarily labeled by Andrew McCallum (`https://www.cs.umass.edu/~Cmccallum/data/cora-refs.tar.gz`) and recently cleaned up and split into five subsets for cross-validation by Singla and Domingos (2006).

– **Query:** The goal of inference is to predict which pairs of citations refer to the same citation (*SameBib*), and similarly for author, title and venue fields (*SameTitle, SameAuthor and SameVenue*). The other atoms are considered evidence atoms.

Yeast. This dataset captures information about a protein's location, function, phenotype, class, enzymes, and protein-protein interaction for the Comprehensive Yeast Genome.[17] It contains four subsets, each of which contains the information about 450 proteins.

– **MLN:** We used the MLN model described by Davis and Domingos (2009). It involves singleton rules for predicting the interaction relationship, and rules describing how protein functions relate to interactions between proteins (i.e. two interacting proteins tend to have similar functions). The final knowledge base has 7 atoms and 8 first-order formulas (2 hard and 6 soft).
– **Query:** The goal of inference is to predict the interaction relation (*Interaction, Function*). All other atoms (e.g., location, protein-class, enzyme, etc.) are considered evidence atoms.

UW-CSE. This dataset records information about the University of Washington (UW), Computer Science and Engineering Department (CSE). The database consists of five subsets: AI, graphics, programming languages, systems, and theory (which corresponds to five research areas).

– **MLN:** We used the MLN model available from the Alchemy website.[18] It includes formulas such as the following: each student has at most one advisor; if a student is an author of a paper, so is her advisor; advanced students only TA courses taught by their advisors; a formula indicates that it is not allowed for a student to have both temporary and formal advisors at the same time ($\neg TemAdvised(s,p) \vee \neg Advised(s,p)$ which is a true statement at UW-CSE), etc. The final knowledge base contains 22 atoms and 94 formulas (considered as 7 hard and 65 soft and we excluded the 22 unit clauses). Note that ten out of these 22 clauses are equality predicates: Sameperson(person; person), Samecourse(course; course), etc. which always have known, fixed values that are true if the two arguments are the same constant. The rest of them are easily predictable using the unit clause method.
– **Query:** The inference task is to predict advisory relationships (*AdvisedBy*), and all other atoms are evidence (corresponding to the all-information scenario in Richardson and Domingos (2006)).

6.2 Metrics

Since computing exact marginal or joint conditional distributions is not feasible for the underlying domains, we evaluated the quality of inference with our method using two metrics: the average conditional log marginal-likelihood (CLL) and the balanced F_1 score. The CLL, which approximates the KL-divergence between the actual and computed marginals returned by an inference algorithm for

[17] Originally prepared by the Munich Information Center for Protein Sequence.
[18] Available at: `http://alchemy.cs.washington.edu/data/uw-cse/`

query ground atoms, is an intuitive way of measuring the quality of the produced marginal probabilities. After obtaining the marginal probabilities from the inference algorithm, the average CLL of a query atom is computed by averaging the log-marginal probabilities of the true values over all its groundings. For the F_1-score metric, we predict that a query ground atom is true if its marginal probability is at least 0.5; otherwise we predict that it is false (see Huynh and Mooney, 2011, 2009; Papai et al., 2012, for more details about measuring prediction quality on the basis of marginal probabilities). The advantage of F_1-score is its insensitivity to true negatives (TNs), and thus it can demonstrate the quality of an algorithm for predicting the few true positives (TPs).

6.3 Methodology and Results

All the experiments were run on a cluster of nodes with multiprocessors running 2.4 GHz Intel CPUs with 4 GB of RAM under RED HAT Linux 5.5. We used the implementations of both the training algorithm (preconditioned scaled conjugate gradient) and inference algorithms (MC-SAT, LBP, and Gibbs) that exist in the Alchemy system (Kok et al., 2007). In addition, we implemented our GEM-MP algorithm as an extension to Alchemy's inference. All of Alchemy's default parameters were retained (e.g., 100 burn-in iterations to negate the effect of initialization in MC-SAT and Gibbs). We conducted our experimental evaluation through five experiments.

6.3.1 Experiment I.

The first experiment was dedicated to answering Q1 and Q2. We ran our experiments using a five-way cross-validation for both Cora and UW-CSE, and a four-way cross-validation for Yeast. In the training phase we learned the weights of models by running a preconditioned scaled conjugate gradient (PSCG) algorithm (in Lowd and Domingos, 2007, it was shown that PSCG performed the best). In the testing phase, and using the learned models, we carried out inference on the held-out dataset by using each of the four underlying inference algorithms to produce the marginals of all groundings of query atoms being true. Such marginal probabilities were used to compute the F_1 and average CLL metrics.

Although a traditional way to assess the inference algorithms would be to run them until convergence and to compare their running times, it is problematic here because some of the algorithms may never converge in the presence of determinism and cycles (e.g. LBP). Or some may converge very slowly with the existence of near-determinism (e.g., Gibbs). Instead we assigned all inference algorithms an identical running time sufficient to judge the inference behavior. Then, at each time step, we recorded the average CLL over all query atoms by averaging their CLLs on each held-out test set. In addition, we computed the F_1 score based on the results we obtained at the end of the allotted time.

Figure 5 shows the results for the average CLL as a function of time for inference algorithms on the underlying datasets. For each point, we plotted error bars displaying the average standard deviation over the predictions for the groundings of each predicate. Note that when the error bars are tiny, they may not be clearly visible in the plots. Overall GEM-MP is the most accurate of all the algorithms

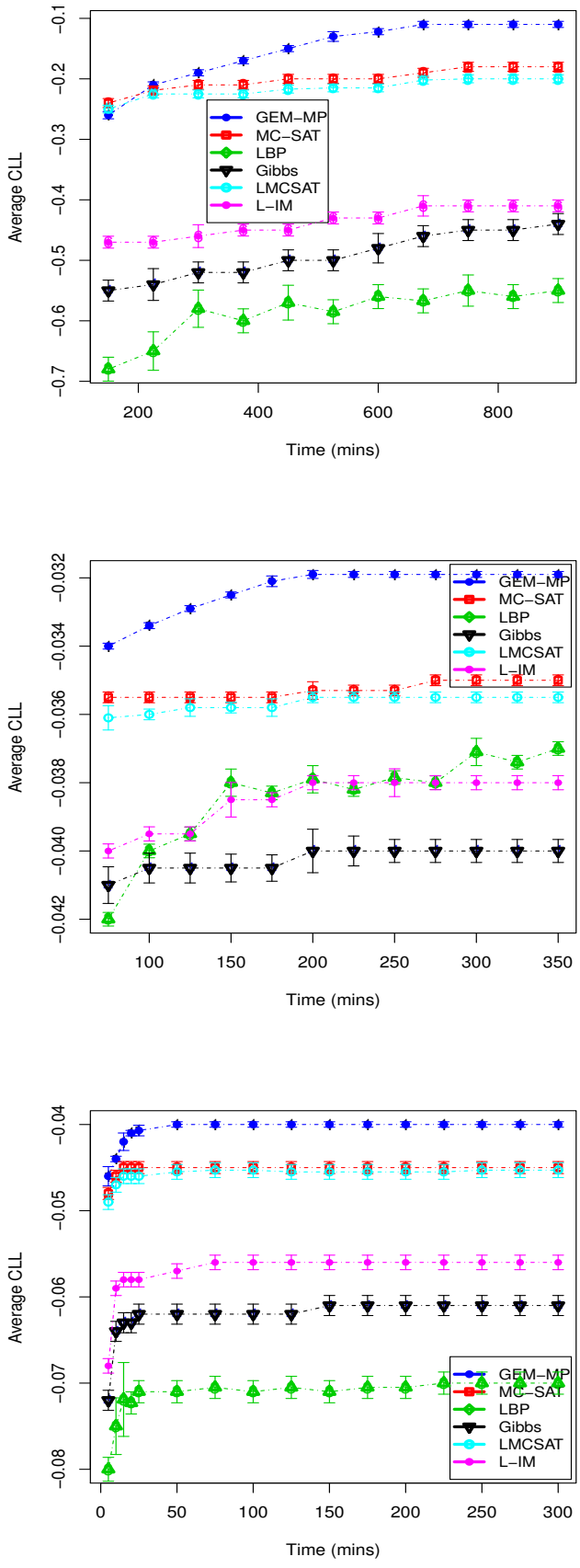


Fig. 5 Average CLL as a function of inference time for GEM-MP, MC-SAT, LBP, Gibbs, LMCSAT, and L-Im algorithms on Cora (*top*), Yeast (*middle*), and UW-CSE (*bottom*).

compared, achieving the best average CLL on Yeast and UW-CSE datasets (*this answers Q1*). For the Cora dataset it took about 225 minutes to dominate all other inference algorithms.[19] MC-SAT came close behind GEM-MP on both Cora and UW-CSE, but considerably further behind on Yeast. LBP was marginally less accurate than Gibbs on both Cora and UW-CSE (which is consistent with the experiments of Singla and Domingos (2008)), but more accurate than Gibbs on Yeast. Remarkably GEM-MP converged quickly on both the Yeast and UW-CSE datasets and converged comparatively fast on Cora as well (*this answers Q2*). By contrast LBP was unable to converge, oscillating on both the Cora and Yeast datasets, and Gibbs converged very slowly on all datasets. L-Im was clearly more accurate than Gibbs on all the tested datasets. In addition its accuracy was better than LBP's accuracy with a large margin on Cora and UW-CSE, and slightly less accurate than LBP on the Yeast dataset. The accuracy of MC-SAT and of its lazy algorithm (LMCSAT) were very close on all the datasets.

Table 4 Average F_1 scores for the GEM-MP, MC-SAT, Gibbs, LBP, LMCSAT, and L-Im inference algorithms on Cora, Yeast, and UW-CSE at the end of the allotted time.

Datasets		Algorithms					
	Query	**GEM-MP**	**MC-SAT**	**Gibbs**	**LBP**	**LMCSAT**	**L-Im**
Cora	*SameBib*	**0.778**	0.695	0.443	0.382	0.690	0.491
	SameAuthor	**0.960**	0.926	0.660	0.657	0.926	0.690
	SameTitle	**0.860**	0.790	0.570	0.515	0.780	0.581
	SameVenue	**0.843**	0.747	0.584	0.504	0.739	0.613
Yeast	*Interacts*	**0.792**	0.669	0.474	0.536	0.651	0.512
	Function	**0.820**	0.691	0.492	0.575	0.679	0.532
UW-CSE	*advisedBy*	**0.762**	0.589	0.483	0.415	0.580	0.504
Overall average		**0.831**	0.730	0.529	0.512	0.720	0.560

Table 4 reports the average F_1 scores for the inference algorithms on the underlying datasets. The results complement those of Figure 5: underscoring the promise of our proposed GEM-MP algorithm to obtain the highest quality among the alternatives for predicting marginals, particularly for the TP query atoms (i.e. query atoms that are true and predicted to be true). GEM-MP substantially outperformed LBP, Gibbs and L-IM on all datasets, achieving 39%, 37%, and 33% greater accuracy respectively (*answer Q2*). MC-SAT was relatively competitive compared with GEM-MP on Cora and UW-CSE, but on the Yeast dataset GEM-MP performed significantly better, attaining 13% greater accuracy than MC-SAT (*conclusive answer to Q1*). Gibbs and LBP rivaled each other on the tested datasets but were both dominated by MC-SAT. LMCSAT was very competitive to its propositional MC-SAT with approximately a 2.2% loss in accuracy.

[19] Note that for the Cora dataset the construction of the grounded network required for inference takes about 185 minutes on average.

6.3.2 Experiment II.

Here we concentrated on Q3. To obtain robust answers we examined the performance of GEM-MP, MC-SAT and LBP at varying amounts of determinism. That is, we re-ran Experiment I at gradual amounts of determinism. We marked each amount of determinism as a *level*, with determinism levels in the range $[0, 50]$. For example the 0-level stands for zero percentage of determinism (i.e., all clauses in the model are considered soft clauses) and the 50-level means 50 percent of determinism (i.e., we considered 50% of the clauses in the model as hard clauses and 50% as soft clauses).

Figure 6 reports the average CLL as a function of time for GEM-MP, LBP, and MC-SAT at different levels of determinism. Overall the results confirm that the amount of determinism in the model has a great impact on both the accuracy and the convergence of GEM-MP and LBP. That is, when increasing the level of determinism, we observe an increase in the accuracy of GEM-MP and a decrease in the accuracy of LBP. At each level of determinism and on all datasets GEM-MP prevailed over the corresponding LBP in terms of accuracy of results (*answering Q3*). In addition the greater the level of determinism, the greater the convergence for GEM-MP, and the greater the non-convergence for LBP (*answering Q2*). Remarkably the 0-level, which has no amount of determinism, exhibits the worst behaviour for GEM-MP.[20] In contrast it is the best level for LBP, though even at this level GEM-MP surpassed LBP on all datasets. For MC-SAT increasing the determinism in the model has a small negative impact on its accuracy.

6.3.3 Experiment III.

This experiment examines Q4. We are interested here in judging the scalability of various inference algorithms. To guarantee a fair comparison, we reran Experiment I while increasing the number of objects in the domain from 100 to 200 by increments of 25, following the methodology previously used by Poon et al. (2008); Shavlik and Natarajan (2009). Then we reported the average running time to achieve convergence or up to a maximum of 5000 and 10000 iterations respectively for the entire inference process.

Figure 7 reports the average inference time as a function of the number of objects in the domain. Overall the results show that both LMCSAT (a lazy-based algorithm) and L-IM (lifted-based algorithm) rivaled each other, and both dominate all other propositional algorithms compared. L-IM was relatively scalable compared with LMCSAT on Yeast and UW-CSE, but on the Cora dataset LMCSAT's scalability was significantly better than L-IM. Aside from Lazy- and Lifted-based algorithms and by considering the propositional ones, the results demonstrate that GEM-MP is scalable compared to the other evaluated inference algorithms. It clearly prevailed over both LBP and Gibbs on the entire range of domain sizes by a significant margin, while saving time by more than a factor of 2 on all datasets. It also rivaled the MC-SAT algorithm overall. Although it came in slightly behind MC-SAT on the Cora dataset, it outperformed MC-SAT in handling all domain sizes on both the Yeast and UW-CSE datasets, whereas MC-SAT ran out of memory with 200 objects on UW-CSE.

[20] With no determinism the hard update rule of GEM-MP is not being used.

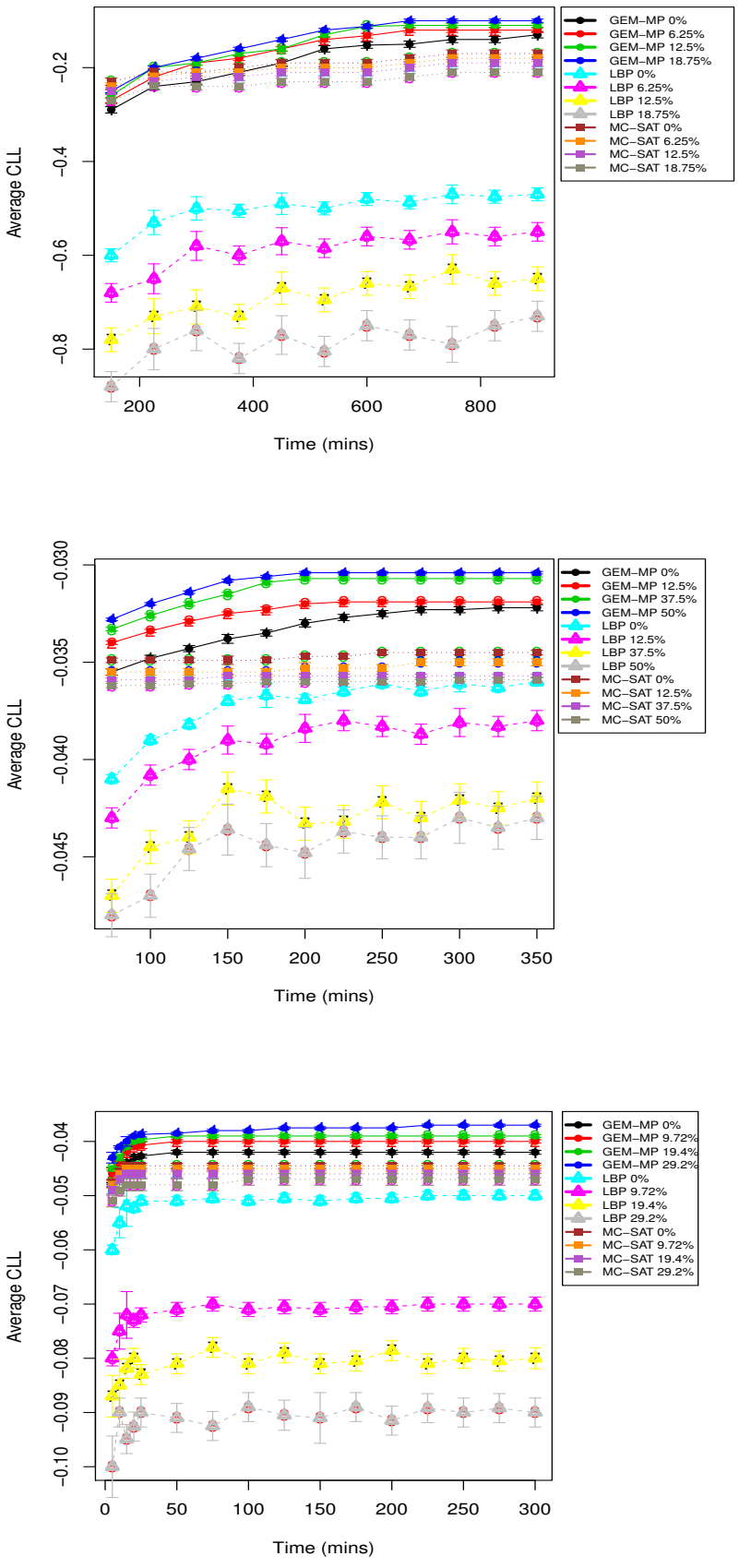
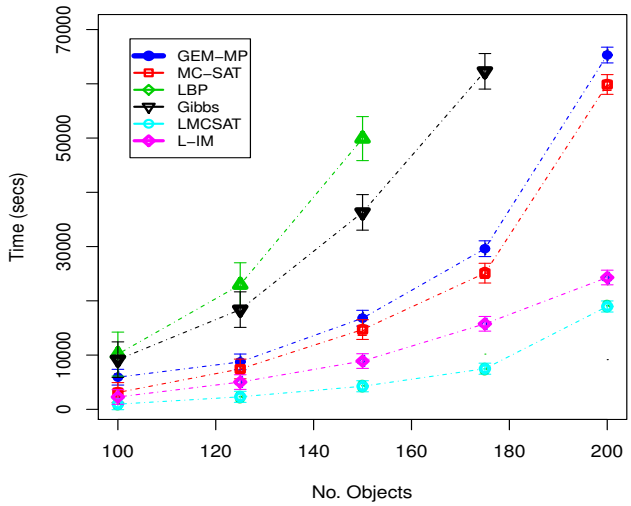
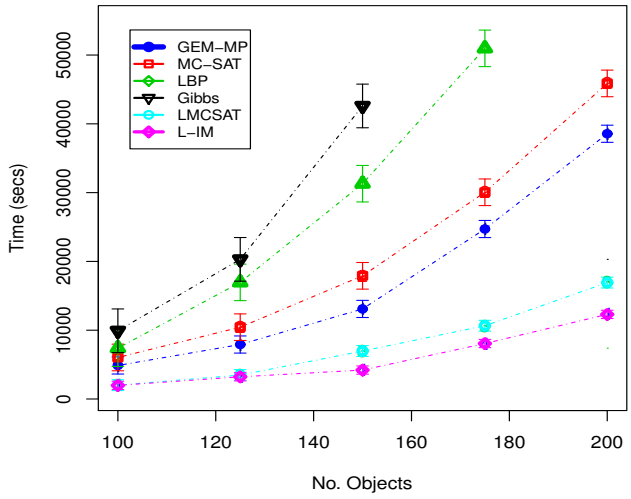


Fig. 6 The impact of determinism on the accuracy of GEM-MP, MC-SAT and LBP for Cora (*top*), Yeast (*middle*), and UW-CSE (*bottom*).





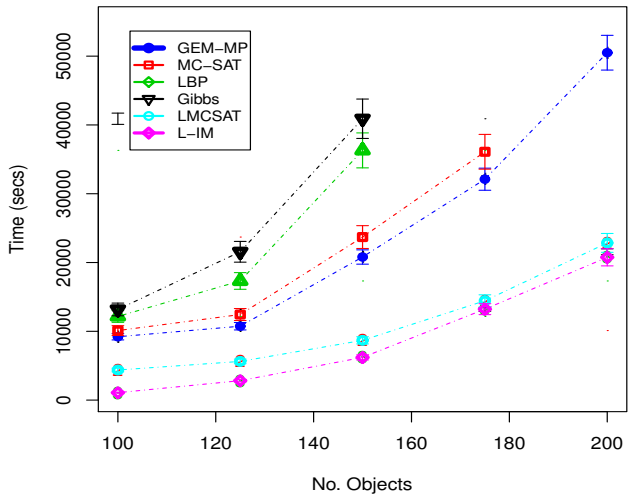


Fig. 7 Inference time vs. number of objects in Cora (*top*), Yeast (*middle*), and UW-CSE (*bottom*).

6.3.4 Experiment IV

This experiment was performed to answer Q5. Here the goal is to compare GEM-MP with three state-of-the art convergent message-passing algorithms:

- *L2-Convex* proposed by Hazan and Shashua (2010, 2008), which runs sequential message passing on the convex-L2 Bethe free energy.
- *RBP* proposed by Elidan et al. (2006), which runs damped Residual BP, a greedy informed schedule for message passing.
- *CCCP* double loop algorithm proposed by Yuille (2001, 2002), which runs message-passing on the convex-concave Bethe free energy.

To evaluate the four underlying message-passing algorithms we apply them to Ising models on a two-dimensional grid network. These model networks are standard benchmarks to evaluate message-passing algorithms as they provide a systematic way to analyze iterative algorithms (see Elidan et al., 2006). Following Hazan and Shashua (2010) and Elidan et al. (2006), we generated 20×20 grids: The distribution has the form $p(x) \propto e^{\sum_{(X_i,X_j)\in\mathcal{E}} \theta_{ij} \cdot X_i X_j + \sum_{X_i} \theta_i \cdot X_i}$, where θ_i, θ_{ij} are parameters (i.e., weights) of the univariate and pairwise potentials respectively. For univariate potentials, the parameters θ_i were drawn uniformly from $\mathcal{U}[-d_f, d_f]$ where $d_f \in \{0.05, 1\}$. For pairwise potentials, we use $e^{\eta \cdot C}$ when $x_i = x_j$ where we sample η in the range $[-0.5, 0.5]$ having some nodes to agree and disagree with each other. C is an agreement factor, so the higher values of C impose stronger clauses (e.g., $C = 200$ and $\eta = 0.5$ yield deterministic potentials since if a state violates a potential with $C = 200$ it becomes 2.69×10^{43} times less probable). Thus to challenge and explore the difficulty of inference in different regimes, we generate the networks with two levels of determinism: Level 1 is $[0\%, 20\%]$ and Level 2 is $[20\%, 40\%]$, with realizations obtained at 10% intervals, 50 graphs at each interval. In each individual realization of the interval, we run the four underlying inference algorithms for the network until convergence or up to 500 iterations. To diagnose the convergence we considered the cumulative percentage of convergence of all algorithms as a function of the number of iterations. To address the quality of results, where exact inference was tractable using the junction tree algorithm, we compute the average KL-divergence (KL) metric between the approximate and exact node marginals for each algorithm on all 20×20 generated Ising grids.

Figure 8 (top) displays the cumulative percentage of convergence as a function of the number of iterations for each algorithm at Level 1 and 2. Overall the results show that GEM-MP converges significantly more often than all other compared convergent message-passing algorithms (*answering Q5*). Also it converges much faster than them. At Level 1 it finishes at 97% convergence rates versus 82% for L2-Convex, 68% for CCCP, and 59% for residual BP. At Level 2 it clearly achieves at least 17.5%, 34.8 %, and 48.4 % better convergence than L2-Convex, CCCP, and residual BP respectively.

Figure 8 (bottom) displays the average KL-divergence (KL) between the approximate and exact node marginals for each algorithm as a function of the number of iterations at the two levels. The results complement those of Figure 8 (top), here again underscoring the promise of GEM-MP for converging to more accurate solutions more rapidly than all other compared algorithms (*answering Q5*). In the two determinism scenarios, it achieves on average 37.8%, 56%, and 61.6 % higher quality marginals in terms of the average KL compared to the L2-Convex,

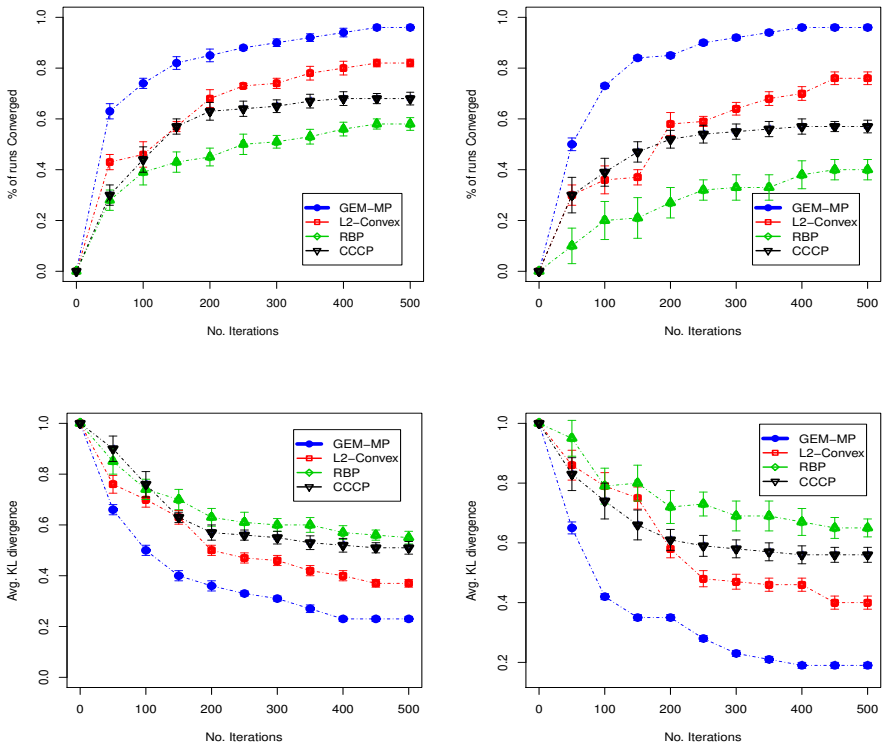


Fig. 8 The results of 20×20 grids of Ising model: (*Top*) The cumulative percentage of convergence (Convergence %) vs. number of iterations, and (*bottom*) the average KL-divergence (KL) metric vs. number of iterations at determinism level 1 $[0\%, 20\%]$ (*left*) and 2 $[20\%, 40\%]$ (*right*).

CCCP, and residual BP methods respectively. Also it finishes at a KL-divergence of 0.23 and 0.19 in the two determinism levels respectively. This shows that the quality of marginals obtained by GEM-MP at Level 2 are more accurate than the ones obtained at Level 1, which is consistent with the results in Experiment II that demonstrate that GEM-MP provides more robust results when there is more determinism in the model.

6.3.5 Experiment V

This experiment attempts to answer Q6. The goal is to compare the quality of solutions returned by GEM-MP at different initialization settings of marginals: GEM-MP with random initialization (GEM-MP-random) and GEM-MP with uniform initialization (GEM-MP-uniform). We re-ran Experiment I for MLNs and recorded the relative correlations of the average CLL between GEM-MP-random and GEM-MP-uniform. In addition, we re-ran Experiment IV for Ising models and report

the relative correlations of the average KL-divergence between GEM-MP-random and GEM-MP-uniform.

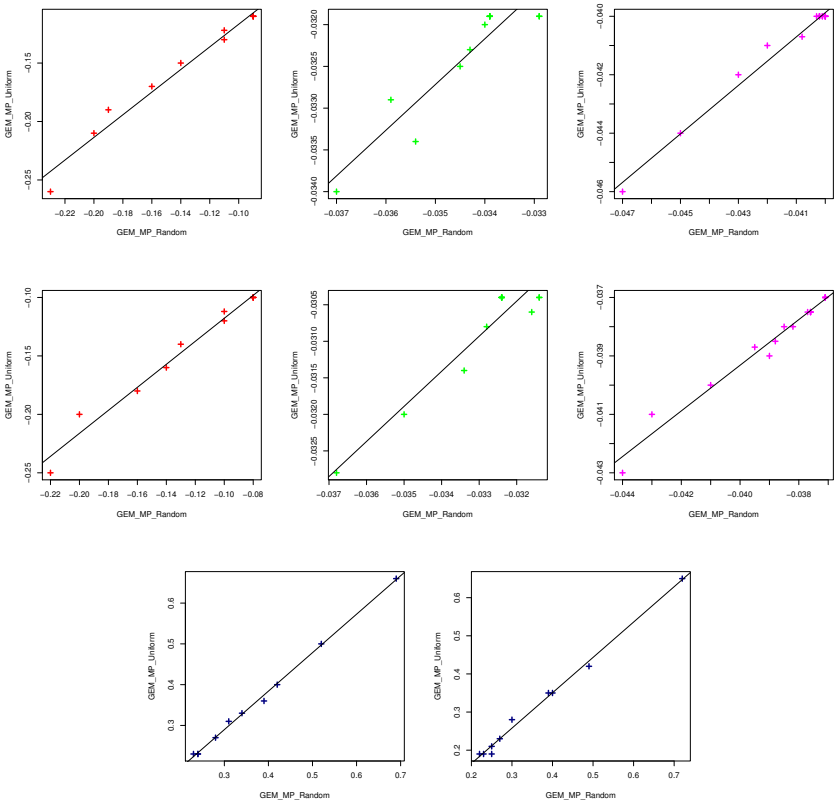


Fig. 9 Top and middle: The average CLL of GEM-MP-random (x-axis) vs. the average CLL of GEM-MP-Uniform (y-axis) for Cora (*left-red*), Yeast (*middle-green*) and UW-CSE (*right-magenta*) at two determinism levels, respectively. Bottom: the average KL-divergence of GEM-MP-random vs. the average KL-divergence of GEM-MP-Uniform for 20×20 grids of Ising model at Level 1 $[0\%, 20\%]$ (*left-blue*) and at Level 2 $[20\%, 40\%]$ (*right-blue*) during iterations.

Figure 9 shows the quality of marginals obtained from GEM-MP-random relative to the quality of marginals of GEM-MP-uniform as a function of the number of iterations at two determinism levels for Cora (red), Yeast (green), UW-CSE (magenta), and Ising (blue). In each scatter plot the line of best fit indicates that both GEM-MP-random and GEM-MP-uniform yield results of nearly identical quality. Any point below the line means that GEM-MP-uniform was more accurate than GEM-MP-random in that iteration, and the contrary is true if the point is above the line. Overall the results show that none of the initialization settings dominates the other (*answering Q6*), and that GEM-MP is not sensitive to the initialization settings.

7 Discussion

The experimental results from the previous section suggest that, in terms of both accuracy and scalability, GEM-MP outperforms LBP inference. It improves message-passing inference in two ways. First, it alleviates the threat of non-convergence in the presence of cycles. This is due to making moderate moves in the marginal likelihood space and the consequences of Jensen's inequality which prevents such moves from overshooting the nearest fixed point. Second, it improves the quality of approximate marginals obtained in the presence of determinism, which we believe is attributable to the virtue of using the concept of generalized arc consistency to leverage the local entries of factors in order to compute more accurate outgoing messages.

Moreover, GEM-MP performs at least as well as the other state-of-the-art sampling-based inference methods (such as MC-SAT and Gibbs). The goal of MC-SAT is to combine a satisfiability-based method (e.g., SampleSAT) with MCMC-based sampling approaches to remedy the challenges engendered by determinism in the setting of MCMC inference. On one hand, GEM-MP achieves a similar goal, but by integrating a satisfiability-based method (i.e., GAC) with message-passing inference, instead of sampling inference. On the other hand, they completely differ in how they use ideas from satisfiability-oriented methods to deal with the issue of determinism.

From the satisfiability perspective, MC-SAT uses SampleSAT (see Wei et al., 2004) to help slice sampling (i.e. MCMC) to near-uniformly sampling a new state given the auxiliary variables. This provides MC-SAT with the ability to rapidly jump between breaking modes, and thus it avoids the local search in MCMC inference from being trapped in isolated modes. Accordingly, one of the limitation of MC-SAT is that it applies a stochastic greedy local search procedure which is unable to make large moves in the state-space between isolated modes. This may affect its capacity to converge to accurate results. Conversely, at a high level, GEM-MP optimizes the setting of parameters with respect to a distribution over hidden variables that captures the relative weights of samples (i.e., the valid local entries) that are generated by individual variables in closed form. Thereby it performs a sort of gradient descent/ascent local search procedure. This gives GEM-MP an advantage in converging to more accurate results than MC-SAT, though MC-SAT is more likely to converge faster than GEM-MP. This could explain the great success of GEM-MP over MC-SAT on most of the experiments (MC-SAT only surpassed GEM-MP on the Cora dataset in experiment III). But we have to remember that, during the training phase, we trained the models by applying a preconditioned scaled conjugate gradient (PSCG) algorithm which uses MC-SAT for its inference step. This in turn gave an advantage to the MC-SAT algorithm when performing inference in the testing phase.

Gibbs is only reliable when neither determinism nor near-determinism are present. LBP for its part also deteriorates in the presence of determinism and near-determinism, but also when cycles are present. Thus if LBP gets stuck in cycles with determinism, it may be lodged there forever. However, if Gibbs hits a local optimum, it would eventually leave, even though it may take considerable time. This could explain the success of Gibbs over LBP. But with the increase of determinism in the model, Gibbs loses out to LBP, as seen in the case of the

Yeast dataset in experiment I. Thus determinism apparently has a stronger effect on Gibbs than on LBP in this experiment.

Furthermore GEM-MP performs better than the other state-of-the-art convergent message-passing inference algorithms such as L2-Convex, CCCP and Damped residual BP. The goal of L2-Convex is to convexify the Bethe free energy to guarantee BP converging to an accurate local minimum. The CCCP algorithm uses a convex-concave Bethe energy to achieve the same purpose. On the one hand, GEM-MP achieves a similar purpose by optimizing a concave variational free energy, which is a lower bound to the model evidence. On the other hand, it additionally leverages the determinism and therefore, while the presence of determinism in a model can hinder the performance and converging behaviour of both L2-Convex and CCCP to reach a local minimum, it increases the possibility that GEM-MP converges to an accurate one.

Overall the experimental results suggest that the initialization of GEM-MP does not significantly matter in practice since the correlation of two initialization settings (i.e., uniform and random) is often moderately positive on average. While we believe that it is important to have a good initialization to ensure that the local minimum that is found is sufficiently close to the global minimum, it seems that a good initialization will depend on the model and data. Therefore in some cases either random or uniform initialization will suffice, whilst in others it may be necessary to use a heuristic. Generally speaking it appears however that GEM-MP is able to reach an accurate result given any initialization, possibly at the expense of a minor increase in computation time.

From the scalability point of view, although Singla (2012) conjectured that lifted inference may subsume lazy, a clear relationship between lifted inference and lazy inference still eludes us. Our experimental results show that neither one was able to dominate the other. On one hand, lazy inference exploits sparseness to ground the network lazily, and therefore greatly reduces the inference memory and time as well. But lazy inference still works at the propositional level, in the sense that the basic units during inference are ground clauses. In contrast, lifted inference exploits a key property of first-order logic to allow answering queries without materializing all the objects in the domain inference. On the other hand, lifted inference requires the network to have a specific symmetric structure, which is not always the case in real-world applications and, in addition, in the presence of evidence most models are not liftable because evidence breaks symmetries. Thus at a high level the structure of the model network plays a significant role in the scalability of inference using different factors: symmetry and sparseness. If the model is extremely sparse then one can expect lazy inference to be more scalable. Lifted inference dominates when the symmetry prevails in the model's structure.

8 Related Work

Belief propagation (BP) was developed by Pearl (1988) as an inference procedure for singly connected belief networks. Pearl was the first to observe that running LBP leads to incorrect results on multi-connected networks. Conversely, other work (such as Mceliece et al., 1998; Frey and MacKay, 1998) has shown success with LBP on loopy networks for turbo code applications. Further, Murphy et al. (1999) reported that LBP can provide good results on graphs with loops. These promising

results shed light on evaluating the performance of BP in other applications and suggest the value of a closer study of its behavior for understanding the reasons for this success. Accordingly, several formulations of LBP have appeared, such as the direct implementation in a factor graph by Kschischang et al. (2001), tree-weighted BP (Wainwright et al., 2003) and the generalized cluster graph method of Mateescu et al. (2010). Most such formulations were influenced by the admirable analysis of Yedidia et al. (2003) who proved a relationship between LBP and Bethe approximation such that the local minima of Bethe free energy are the fixed points of LBP. Complementing this, further analysis has also explored LBPs relationships to variational approximations (Yedidia et al., 2005). This pioneering work outlined new research directions for a deeper understanding of and improvements to LBP.

Studying LBP's convergence. The convergence of LBP has been studied by examining the sufficient conditions to ensure the existence and the uniqueness of local minima. Early on, Heskes (2004) pointed out that if a graph involves a single cycle, then we have a unique local minimum, and the convergence of LBP can be all but guaranteed. Supplementing Heskes (2004), Yedidia et al. (2005) showed that if a factor graph has more than one cycle then the convexity of Bethe free energy is violated and thus the uniqueness of LBP's fixed points is also violated. More recently, Shi et al. (2010) discussed new sufficient conditions for the convergence of LBP by deriving uniform and non-uniform error bounds on the messages. But this research direction ignores an important observation made by Heskes (2002):

> "Still, loopy belief propagation can fail to converge, and apparently for two different reasons. The first rather innocent one is a too large step size, similar to taking a too large "learning parameter" in gradient-descent learning"

In this paper, by relying on a variational formulation, our algorithm optimizes variational bounds on the model evidence and it implicitly guarantees not to overstep a local minimum.

LBP and Bethe free energy. Here, mainstream work attempts to derive new types of LBP for approximate inference by directly optimizing the Bethe energy functional, such as the double loop algorithm (Yuille, 2001). However, the main disadvantage of this algorithm is that it requires solving an optimization problem at each iteration, which results in a slower convergence. Another class of algorithm is known as cluster-graph BP, which runs LBP on sub-trees of the cluster graph. These algorithms exhibit faster convergence and introduce a new way of characterizing the connections between LBP and optimization problems based on the energy functional. Consequently, several works appeared which generalized the class of LBP by introducing variants of the energy functional that improve the convergence of LBP. For instance, Wainwright and Jordan (2003) and Nguyen et al. (2004) proposed a convexified free energy that provides an upper bound on the partition functions. But the algorithms that have been built on this energy functional still cannot guarantee convergence. Recently, alternative algorithms have been introduced to guarantee convergence for such energy functionals (Hazan and Shashua, 2008; Meltzer et al., 2009; Globerson and Jaakkola, 2007; Hazan and Shashua, 2010).

At a high level, our GEM-MP approach resembles previously mentioned approaches in that it is based on variational inference and involves minimizing a free-energy functional.

It remains unclear if there is a relationship between determinism and the uniqueness of local minima of LBP. However, our experiments here support prior work that has also observed that applying LBP on graphical models with determinism and cycles is more likely to oscillate or converge to wrong results.

LBP and Constraint propagation. Horsch and Havens (2000) proposed an algorithm that is a generalization of arc consistency used in constraint reasoning, and a specialization of the LBP used for probabilistic reasoning. The idea was to exploit the relationship between LBP and arc consistency to compute the solution probabilities, which can be then used as a heuristic to guide constructive search algorithms to solve binary CSPs. The bucket-elimination procedure was proposed by Dechter and Mateescu (2003). However it is known that such a procedure has a time and space complexity that is exponential in the induced width of the problem graph, related to the processing order of variables and to how densely these variables are connected to each other. Alternatively, Mateescu et al. (2010) presented approaches that are based on constructing a relationship between LBP and constraint propagation techniques. One idea underlying these approaches is to transform the loopy graph into a tree-like structure to alleviate the presence of cycles, and then to exploit constraint propagation techniques to tackle the determinism. Building on these ideas we explore the second research hypothesis: *constraint satisfaction techniques might be able to help address the challenges resulting from determinism in the graphical models.*

A recent extension of such approaches is the combination of LBP, constraint propagation, and expectation maximization to derive an efficient heuristic search for solving both satisfiability problems (see Hsu et al., 2008, 2007) and constraint satisfaction problems (see Le Bras et al., 2009). Although these algorithms perform well in finding solutions, they apply only to graphical models that have no probabilistic knowledge. In contrast, our GEM-MP method is able to handle probabilistic knowledge.

Damped LBP. Another traditional research area to handle non-convergence has involved dampening the marginals (see Koller and Friedman, 2009) in order to diminish oscillation. However, in many cases, the dampening causes LBP to converge but often yields a poor quality result (Mooij and Kappen, 2005). This is because the correct results are not usually in the average point (Murphy et al., 1999). The second track of this research direction is to alleviate double counting by changing the schedule of updating messages (e.g., sequentially on an Euler path, as per Yeang (2010), residual BP, as per Elidan et al. (2006), among others) besides adapting the initialization of the marginals (e.g., restart with different initializations, as per Koller and Friedman (2009)). However, this cannot guarantee convergence since the algorithm still runs the risk of overshooting the nearest local minimum. Whilst, a key of the approach of GEM-MP is that its iterations are constrained by the variational inequality and therefore updates to distributions over hidden variables are done in a way such that the variational lower bound never exceeds the log marginal likelihood.

Re-parameterized LBP. More recently, Smith and Gogate (2014) introduced a new approach aimed at dealing with determinism more effectively. The idea of this approach is to re-parameterize the Markov network by changing the entry in a factor that has zero to any non-negative real value in such a way that the LBP algorithm converges faster. Our GEM-MP also addresses the problem of determinism by improving message-passing inference to deal with determinism and cycles more effectively, but our approach is different being rooted in both variational techniques and leveraging generalized arc consistency.

LBP and variational methods. Another research area combines message-passing with other variational methods to produce new types of LBP that can guarantee convergence. For example, Winn and Bishop (2005) presented variational message passing as a way to view many variational inference techniques, and it represents a general purpose algorithm for approximate inference. This algorithm shows great performance when it applies to conjugate exponential family models network. Weinman et al. (2008) proposed a sparse variational message passing algorithm to dramatically accelerate the approximate inference needed for parameter optimization related to the problem of stereo vision. Recently, Dauwels et al. (2005) proposed a generic form of structured variational message-passing and investigated a message-passing formulation of EM. Our GEM-MP method can be seen as akin to these message-passing inference methods. But a basic aspect of GEM-MP is the exploitation of ideas from CS to handle the challenges stemming from determinism.

Lifted LBP. Another promising research area that has been recently explored seeks to improve the scalability of LBP on models that feature large networks. Here, mainstream work attempts to exploit some structural properties in the network like symmetry (see Ahmadi et al., 2013), determinism (see Papai et al., 2011; Ibrahim et al., 2015), sparseness (see Poon et al., 2008), and type hierarchy (see Kiddon and Domingos, 2011) to scale LBP inference. For instance, Lifted Inference either directly operates on the first-order structure or uses the symmetry present in the structure of the network to reduce its size (e.g., Ahmadi et al., 2013). In this context, the key idea is to deal with groups of indistinguishable variables rather than individual variables. Poole (2003) was one of the first to show that variable elimination can be lifted to avoid propositionalization. This has been extended with some lifted variants of the algorithm proposed by De Salvo Braz et al. (2005) and Milch et al. (2008). Subsequently, Singla and Domingos (2008) proposed the first lifted version of LBP, which has been extended by Sen et al. (2009), and generalized with the emergence of the color message-passing algorithm introduced by Kersting et al. (2009) for approximating the computational symmetries. Subsequently, it was shown by Gogate and Domingos (2011) that to avoid dissipating the capabilities of first-order theorem proving, we have to take into considerations the logical structure. Based on that, lifted variants of weighted model counting have been proposed by Gogate and Domingos (2011), meanwhile variants of lifted knowledge compilation such as the bisimulation-based algorithm were introduced by Van den Broeck et al. (2011). Later on, it was observed that in some cases the constructed lifted network can itself be quite large, making it very close in size to the fully propositionalized one, and yielding no speedup by lifting the inference. The interesting argument proposed by Kersting (2012) concludes that the evidence

problem could be the reason: symmetries within models easily break down when variables become correlated by virtue of depending asymmetrically on evidence and thus lifting produces models that are often not far from propositionalized ones, diminishing the power of lifted inference. Thus, one can obtain better lifting by performing shattering as needed during BP inference such as anytime BP proposed by De Salvo Braz et al. (2009), or exploit the model's symmetries before we obtain the evidence as demonstrated in (Bui et al., 2012), or shattering a model into local pieces and then iteratively handling the pieces independently and re-combining the parameters from each piece as explained in (Ahmadi et al., 2013). Recently, Gogate et al. (2012) show that the evidence problem with lifting inference can be solved when applied to importance sampling algorithms by using an informed distribution derived from a compressed representation of MLN. Our approach is different from the above lifted-based message passing algorithms being built on a propositional basis, but it can be easily incorporated with their benefits for lifting its inference.

9 Conclusion and Future Work

Our work has targeted the less studied issue of the use of LBP and message passing techniques in probabilistic models possessing both cycles and determinism. To fully exploit determinism as opposed to having determinism posing a problem for inference, we have examined some of the intricacies of message passing algorithms. The novelty of our work lies in the proposal and exploration of an approach which we have named Generalized arc-consistency Expectation-Maximization Message-Passing (GEM-MP), a message-passing algorithm that applies a form of variational approximate inference in an extended form of an underlying graphical model. We have focused our experiments on Markov logic, but our method is easily generalized to other graphical models. To demonstrate the ease of generalizing our approach, we have also presented results using Ising models and we find that our method outperforms a variety of state-of-the-art techniques. The rules of GEM-MP can be viewed as a free energy minimization method whose successive updates form a path of bounded steps to the nearest local minimum in the space of approximate marginals. Using entity resolution and link prediction problems, we have experimentally validated the effectiveness of GEM-MP for converging to more accurate marginals and addressed the limitations of LBP engendered by the presence of cycles and determinism.

As with other variational methods, much of the strength of our method is a consequence of Jensen's inequality which enables variational message-passing inference to estimate marginals - through the optimization of variational parameters - by tightening a lower bound on the model's marginal likelihood at each approximate marginal update, such that we cannot overshoot the underlying true marginal likelihood. We believe this effect alleviates the threat of non-convergence due to cycles. In addition, the effectiveness of generalized arc consistency for handling the logical structures can be used to exploit structure in the problem that is not normally available to a more naive message-passing algorithm. In so doing, our formulation transforms determinism from a limitation into an advantage from the perspective of GEM-MP.

These explorations point to a number of promising directions for future work. We plan to evaluate the use of GEM-MP as an inference subroutine for learning. Also, we intend to investigate the lifted (see Ahmadi et al., 2013; Singla et al., 2010) and the lazy (cf. Poon et al., 2008) versions of GEM-MP to enhance its scalability. Finally, we intend to increase the accuracy of GEM-MP by deriving new update rules that apply a global approximation for $q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})$ in the $M_{q(\mathcal{Y})}$-step of GEM-MP.

Acknowledgements We are grateful to Henry Kautz for helpful discussions. Thanks to Aniruddh Nath for providing help with some datasets and models.

Appendix A: Proofs of Propositions

This section contains the proofs of Propositions 1 and 2 in Section 3, and the proof of Proposition 3 in Section 4.

– Proposition 1.

Proof Assume that we have a non-negative factor $f_i(X_1, \ldots, X_n)$ of n argument variables in the original factor graph $\mathcal{G}$ that is extended to $\hat{f}_i(X_1, \ldots, X_n, O_i, Y_i)$ in the extended factor graph $\hat{\mathcal{G}}$ by attaching to it both an auxiliary activation node O_i and an auxiliary mega-node Y_i such that the extended $\hat{f}_i(X_1, \ldots, X_n, O_i, Y_i)$ never shares either its activation node or its mega-node with other extended factors. Let $\mathcal{Z}$ being the set of all possible local entries of $\hat{f}_i$ that involves $\bar{O}_i = 1$. Each local entry of $\mathcal{Z}$ has the form $(x_1, \ldots, x_n, y_i, 1)$, where $x = (x_1, \ldots, x_n)$ is a configuration to the argument variables of f_i, y_i is a state of auxiliary mega-node Y_i, and value 1 for auxiliary activation node O_i. By construction, for each possible local entry $(x_1, \ldots, x_n, y_i, 1)$ in $\mathcal{Z}$ we have that:

$$\begin{aligned}\sum_{Y_i} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} &= \sum_{Y_i:\mathcal{Z}(y_i=x)} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} \\ &+ \sum_{Y_i:\mathcal{Z}(y_i\neq x)} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1}.\end{aligned} \tag{56}$$

The first and the second parts in the right hand side of Eq. (56) represent the marginalization over the local entries in $\mathcal{Z}$ that involve $x = y_i$ and $x \neq y_i$, respectively. However, we have from Eq. (10) that $\hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = 0$ when $y_i \neq x$. This is because at the dissatisfaction of the indicator constraint, the extended factor $\hat{f}_i$ assigns a value 0. Thus, we now have that:

$$\begin{aligned}\sum_{Y_i} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} &= \sum_{Y_i:\mathcal{Z}(y_i=x)} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} \\ &+ \sum_{Y_i:\mathcal{Z}(y_i\neq x)} \overbrace{\hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1}}^{\text{Zero}}.\end{aligned} \tag{57}$$

Furthermore, there is no need to take the summation $\sum_{Y_i:\mathcal{Z}(y_i=x)}$ since often there is only one possible local entry in $\mathcal{Z}$ on which $y_i = x, \forall x, \forall y_i$. In addition, we have from Eq. (9) that $\hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1}$ preserves the value of f_i when $y_i = x$ (i.e., the case of satisfaction of the indicator constraint). Thus, we have:

$$\sum_{Y_i} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = f_i(x_1, \ldots, x_n) \tag{58}$$

Since Eq. (58) is true for any configuration to the argument variables of $x = (x_1, \ldots, x_n)$ of f_i, then it is also true for any set of configurations:

$$\sum_{Y_i} \hat{f}_i(X_1, \ldots, X_n, Y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = f_i(X_1, \ldots, X_n),\ \forall \hat{f}_i \in \hat{\mathcal{F}} \tag{59}$$

This implies the correctness of the proposition for any factor of n argument variables in the original factor graph. □

– Proposition 2.

Proof The equivalence is proved once we demonstrate that the two factor graphs define an identical joint probability over variables whose marginals we want to compute. Assume that we have an arbitrary factor graph $\mathcal{G}$ that involves N random variables, $\{X_1, \ldots, X_N\}$.[21] It accommodates M factors, $\{f_1(\mathcal{X}_{f_1}), \ldots, f_M(\mathcal{X}_{f_M})\}$, where $\mathcal{X}_{f_a}$ is the subset of variables from $\{X_1, \ldots, X_N\}$ that are adjacent (i.e., argument) variables to f_a.

Then, without loss of generality, the joint probability of $\mathcal{G}$ can be defined as follows:

$$P(X_1, \ldots, X_N) = \prod_{a=1}^{M} f_a(\mathcal{X}_{f_a}) \tag{60}$$

Further, assume that we extend $\mathcal{G}$ to an extended factor graph $\hat{\mathcal{G}}$ by adding both an auxiliary activation node O_a and auxiliary mega-node Y_a for each individual factor $f_a(\mathcal{X}_{f_a})$, obtaining its corresponding extended factor $\hat{f_a}(\mathcal{X}_{f_a}, O_a, Y_a)$. The extended factor graph $\hat{\mathcal{G}}$ now includes the original variables $\{X_1, X_2, \ldots, X_N\}$, activation variables $\{O_1, O_2, \ldots, O_M\}$, and mega-node variables $\{Y_1, Y_2, \ldots, Y_M\}$. It accommodates M extended factors, $\{\hat{f_1}(\mathcal{X}_{f_1}, O_1, Y_1), \ldots, \hat{f_M}(\mathcal{X}_{f_M}, O_M, Y_M)\}$.

Thus, the joint probability of $\hat{\mathcal{G}}$ is defined as follows:

$$P(X_1, \ldots, X_N, Y_1, \ldots, Y_M, O_1, \ldots, O_M) = \prod_{a=1}^{M} \hat{f_a}(\mathcal{X}_{f_a}, O_a, Y_a) \tag{61}$$

Since it is a condition that all the activation variables must be evidenced with one to get the equivalence, then we can reduce each extended factor $\hat{f_a}$ with $\bar{O_a} = 1$:

$$P(X_1, .., X_N, Y_1, .., Y_M, \bar{O_1}, .., \bar{O_M}) \Big|_{\bar{O_a}=1, \forall a \in \{1,..,M\}} = \prod_{a=1}^{M} \hat{f_a}(\mathcal{X}_{f_a}, \bar{O_a}, Y_a) \Big|_{\bar{O_a}=1} \tag{62}$$

At this point, we can eliminate the auxiliary mega-node variables from Eq. (62) by marginalizing each extended factor $\hat{f_a}$ over its mega-node Y_a, we then have:

$$\begin{aligned} P(X_1, \ldots, X_N) &= \textstyle\sum_{Y_1, \ldots, Y_M} P(X_1, .., X_N, Y_1, .., Y_M, \bar{O_1}, .., \bar{O_M}) \Big|_{\bar{O_a}=1, \forall a \in \{1,..,M\}} \\ &= \textstyle\sum_{Y_1, \ldots, Y_M} \prod_{a=1}^{M} \hat{f_a}(\mathcal{X}_{f_a}, \bar{O_a}, Y_a) \Big|_{\bar{O_a}=1} \end{aligned} \tag{63}$$

However since all auxiliary mega-nodes are connected independently to individual factors, then we can distribute the summation over the product with respect to individual extended factors in Eq. (63), and we obtain:

$$P(X_1, \ldots, X_N) = \prod_{a=1}^{M} \sum_{Y_a} \hat{f_a}(\mathcal{X}_{f_a}, \bar{O_a}, Y_a) \Big|_{\bar{O_a}=1} \tag{64}$$

However, from Proposition 1 we have that:

$$\sum_{Y_a} \hat{f_i}(\mathcal{X}_{f_a}, \bar{O_i}, Y_i) \Big|_{\bar{O_i}=1} = f_i(\mathcal{X}_{f_a}) \tag{65}$$

By applying Eq. (65) for each reduced extended factor in Eq. (64), we obtain the joint probability of the extended factor graph $\hat{\mathcal{G}}$ over the variables $\{X_1, , \ldots, X_N\}$ as follows:

$$P(X_1, , \ldots, X_N) = \prod_{a=1}^{M} f_a(\mathcal{X}_{f_a}) \tag{66}$$

[21] For simplicity, suppose that we want to compute their marginal probability for all variables $\{X_1, \ldots, X_N\}$.

This joint distribution we obtained for the extended factor graph $\hat{\mathcal{G}}$ is identical to the joint distribution that is defined by the original factor graph $\mathcal{G}$ in Eq. (60), which implies the equivalence between the two factor graphs. □

– Proposition 3.

Proof We introduce a complexity bound of the algorithm that is based on the efficiency of functions and tools implemented in Alchemy (Kok et al., 2007) that are used by the algorithm. Assume that n is the number of ground atoms and m_h, m_s are the number of hard and soft ground clauses respectively, where $m = m_h + m_s$ is the total number of ground clauses. Let T_l be the time required for computing the pGAC probability, $1 - \xi(X_j, f_i)$, of a ground atom X_j with respect to a ground clause f_i. Also assume that T_h, T_s are the time required to perform hard and soft update rules respectively. The algorithm consists of three stages:

– Initialization stage $\mathcal{S}_1$: requires $\mathcal{S}_1 \in \Theta(n)$ to initialize the marginals of n ground atoms.
– Discrimination stage $\mathcal{S}_2$: requires $\mathcal{S}_2 \in \Theta(nm)$ since for each ground atom we iterate through its ground clauses to decide whether it is involved in hard clauses or soft clauses or both.
– Inference stage $\mathcal{S}_3$: here for each ground atom in $\mathcal{X}_h$ we run the hard update rule, then for each ground atom in $\mathcal{X}_s$ we run the soft update rule, thus we have:

$$\mathcal{S}_3 = |\mathcal{X}_h| \times T_h + |\mathcal{X}_s| \times T_s \tag{67}$$

To compute the computational complexity required by both T_h and T_s. For T_h we first compute $\left|\mathcal{F}^h_{X_j}\right|$ once which requires $\Theta(m_h)$ and then compute T_l for the set of hard ground clauses that involve X_j as positive and negative respectively, which requires $\Theta(m_h T_l)$. To compute the pGAC probability T_l for each $X_j \in \mathcal{X}_h$ with respect to one ground clause f_i we iterate through other ground atoms in f_i except X_j to multiply their marginals at the opposite value. This requires linear time in the arity of the clause, therefore, we have that T_l is bounded above by r, the maximum arity of the ground clauses. Hence, the time required by each hard update rule can be obtained as follows:

$$T_h \in \Theta(m_h) + \Theta(m_h T_l) \in O(m_h r) \tag{68}$$

In an analogous way, the time required by each soft update rule:

$$T_s \in \Theta(m_s) + \Theta(m_s T_l) \in O(m_s r) \tag{69}$$

We take Eqs. (68) and (69) and substitute for Eq. (67) to obtain the computational complexity of $\mathcal{S}_3$:

$$\mathcal{S}_3 \in |\mathcal{X}_h| \times O(m_h r) + |\mathcal{X}_s| \times O(m_s r) \tag{70}$$

Since $|\mathcal{X}_h|$ and $|\mathcal{X}_s|$ are less than n, and m_h and m_s are less than m

$$\mathcal{S}_3 \in O(nmr) \tag{71}$$

Using Eq. (71), the total complexity of the three stages of the algorithm can be bounded as:

$$\mathcal{S}_1 + \mathcal{S}_2 + \mathcal{S}_3 \in O(\max[n, nm, nmr]) = O(nmr) \tag{72}$$

This implies that the worst case complexity of the algorithm is $O(nmr)$. □

References

Ahmadi B., Kersting K., Mladenov M., Natarajan S. (2013). Exploiting symmetries for scaling loopy belief propagation and relational training. *Machine learning* 92(1):91–132.

Bach F. R., Jordan M. I. (2001). Thin junction trees. In: Proceedings of the 14th Conference on Neural Information Processing Systems: Advances in Neural Information Processing Systems 14 (NIPS-2001), MIT Press. pp. 569–576.

Beal M. J., Ghahramani Z. (2003). The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics* 7:453–464.

Van den Broeck G., Taghipour N., Meert W., Davis J., De Raedt L. (2011). Lifted probabilistic inference by first-order knowledge compilation. In: Proceedings of the Twenty-Second international joint conference on Artificial Intelligence, Barcelona, Catalonia, Spain, 1622 July, AAAI Press. pp. 2178–2185.

Bui H. B., Huynh T. N., de Salvo Braz R. (2012). Exact lifted inference with distinct soft evidence on every object. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, Toronto, Ontario, Canada, AAAI Press. pp. 1875–1881.

Dauwels J., Korl S., Loeliger H.-A. (2005). Expectation maximization as message passing. In: Proceedings of IEEE International Symposium on Information Theory (ISIT 2005), Adelaide Convention Centre Adelaide, Australia, IEEE computer society. pp. 583–586.

Davis J., Domingos P. (2009). Deep transfer via second-order markov logic. In: Proceedings of the 26th annual International Conference on Machine Learning (ICML-09), Montreal, Quebec, Canada, ACM, Montreal, Canada. pp. 217–224.

De Salvo Braz R., Amir E., Roth D. (2005). Lifted first-order probabilistic inference. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, AAAI Press. pp. 1319–1325.

De Salvo Braz R., Natarajan S., Bui H., Shavlik J., Russell S. (2009). Anytime lifted belief propagation. In: Proceedings of 6th International Workshop on Statistical Relational Learning, Leuven, Belgium, vol. 9, pp. 1–3.

Dechter R., Mateescu R. (2003). A simple insight into iterative belief propagation's success. In: Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence, Acapulco, Mexico, Morgan Kaufmann Publishers Inc.. pp. 175–183.

Elidan G., McGraw I., Koller D. (2006). Residual belief propagation: Informed scheduling for asynchronous message passing. In: Proceedings of the Twenty-Second Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-06), AUAI Press, Arlington, Virginia. pp. 165–173.

Flach P. A. (2010). First-order logic. In: Encyclopedia of Machine Learning, Springer, pp. 410–415.

Frey B. J., MacKay D. J. (1998). A revolution: Belief propagation in graphs with cycles. In: Proceedings of the 11th Conference on Neural Information Processing Systems: Advances in neural information processing systems 11 (NIPS-1998), Morgan Kaufmann. pp. 479–485.

Getoor L., Taskar B. (2007). Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press.

Globerson A., Jaakkola T. (2007). Convergent propagation algorithms via oriented trees. In: Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22, AUAI Press. pp. 133–140.

Gogate V., Domingos P. (2011). Probabilistic theorem proving. In: Proceedings of the Twenty-Seventh Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-11), AUAI Press, Corvallis, Oregon. pp. 256–265.

Gogate V., Jha A. K., Venugopal D. (2012). Advances in lifted importance sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada, AAAI Press. pp. 1910–1916.

Hazan T., Shashua A. (2008). Convergent message-passing algorithms for inference over general graphs with convex free energies. In: Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, Helsinki, Finland, July 9-12, pp. 264–273.

Hazan T., Shashua A. (2010). Norm-product belief propagation: Primal-dual message-passing for approximate inference. *IEEE Transactions on Information Theory* 56(12):6294–6316.

Heskes T. (2002). Stable fixed points of loopy belief propagation are local minima of the bethe free energy. In: Proceedings of the 15th Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 9-14: Advances in neural information processing systems 15 (NIPS-2002), Curran Associates Inc.. pp. 343–350.

Heskes T. (2004). On the uniqueness of loopy belief propagation fixed points. *Neural Computation* 16(11):2379–2413.

van Hoeve W. J., Pesant G., Rousseau L.-M. (2006). On global warming: Flow-based soft global constraints. *Journal of Heuristics* 12(4-5):347–373.

Horsch M. C., Havens W. S. (2000). Probabilistic arc consistency: A connection between constraint reasoning and probabilistic reasoning. In: Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., pp. 282–290.

Hsu E. I., Kitching M., Bacchus F., McIlraith S. A. (2007). Using expectation maximization to find likely assignments for solving csp's. In: Proceedings of 22nd National Conference on Artificial Intelligence (AAAI '07), Vancouver, Canada, AAAI Press. vol. 22, pp. 224–232.

Hsu E. I., Muise C., Beck J. C., McIlraith S. A. (2008). Probabilistically estimating backbones and variable bias. In: Proceedings of 14th International Conference on Principles and Practice of Constraint Programming (CP '08), Sydney, Australia, Springer. pp. 613–617.

Huynh T. N., Mooney R. J. (2009). Max-margin weight learning for markov logic networks. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Part 1, Bled, Slovenia, September 7-11, Springer. vol. 5781, pp. 564–579.

Huynh T. N., Mooney R. J. (2011). Online max-margin weight learning for markov logic networks. In: Proceedings of SIAM-11 International conference on Data Mining, Mesa, Arizona, USA, SIAM / Omnipress. pp. 642–651.

Ibrahim M.-H., Pal C., Pesant G. (2015). Exploiting determinism to scale relational inference. In: Proceedings of the Twenty-Ninth National Conference on Artificial Intelligence (AAAI'15), January 25-30, 2015, Austin, Texas, USA, AAAI Press. pp. 1756–1762.

Kersting K. (2012). Lifted probabilistic inference. In: Proceedings of 20th European Conference on Artificial Intelligence (ECAI–2012), August 27-31, Montpellier, France, IOS Press: ECCAI. pp. 33–38.

Kersting K., Ahmadi B., Natarajan S. (2009). Counting belief propagation. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, Quebec, June 18-21, AUAI Press, pp. 277–284.

Kiddon C., Domingos P. (2011). Coarse-to-fine inference and learning for first-order probabilistic models. In: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, San Francisco, California, USA, August 7-11, AAAI Press. pp. 1049–1056.

Kok S., Singla P., Richardson M., Domingos P., Sumner M., Poon H., Lowd D. (2007). The alchemy system for statistical relational ai. In: Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA, http://alchemy.cs.washington.edu. .

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

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

Lauritzen S. L., Spiegelhalter D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society Series B (Methodological)* 50:157–224.

Le Bras R., Zanarini A., Pesant G. (2009). Efficient generic search heuristics within the embp framework. In: Proceedings of the 15th international conference on Principles and practice of constraint programming (CP'09), Lisbon, Portugal, Springer-Verlag, Berlin, Heidelberg. pp. 539–553.

Lowd D., Domingos P. (2007). Efficient weight learning for markov logic networks. In: Proceedings of 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007), Warsaw, Poland, September 17-21, Springer. pp. 200–211.

Mateescu R., Kask K., Gogate V., Dechter R. (2010). Join-graph propagation algorithms. *Journal of Artificial Intelligence Research* 37:279–328.

Mceliece R. J., Mackay D. J. C., Cheng J.-f. (1998). Turbo decoding as an instance of pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communications* 16:140–152.

Meltzer T., Globerson A., Weiss Y. (2009). Convergent message passing algorithms - a unifying view. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, AUAI Press. pp. 393–401.

Milch B., Zettlemoyer L. S., Kersting K., Haimes M., Kaelbling L. P. (2008). Lifted probabilistic inference with counting formulas. In: Proceedings of the Twenty Third Conference on Artificial Intelligence, Chicago, Illinois, USA, AAAI Press. vol. 8, pp. 1062–1068.

Mooij J. M., Kappen H. J. (2005). Sufficient conditions for convergence of loopy belief propagation. In: Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI-05), Edinburgh, Scotland, July 26-29, AUAI Press. pp. 396–403.

Murphy K., Weiss Y., Jordan M. (1999). Loopy belief propagation for approximate inference: An empirical study. In: Proceedings of the Fifteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-99), Stockholm, Sweden, Morgan Kaufmann. pp.

467–476.
Neal R. M., Hinton G. E. (1999). Learning in Graphical Models, MIT Press, chap. A View of the EM Algorithm That Justifies Incremental, Sparse, and Other Variants, pp. 355–368.
Nguyen X., Wainwright M. J., Jordan M. I. (2004). Decentralized detection and classification using kernel methods. In: Proceedings of the twenty-first international conference on Machine learning, (ICML), Banff, Canada, ACM. vol. 69, pp. 80–88.
Papai T., Singla P., Kautz H. (2011). Constraint propagation for efficient inference in markov logic. In: Proceedings of 17th International Conference on Principles and Practice of Constraint Programming (CP 2011), Perugia, Italy, September 12-16, springer. pp. 691–705.
Papai T., Kautz H. A., Stefankovic D. (2012). Slice normalized dynamic markov logic networks. In: Proceedings of 26th Conference on Neural Information Processing Systems, December 3-8 Harrahs and Harveys, Lake Tahoe: Advances In Neural Information Processing Systems 25, Curran Associates Inc.. pp. 1916–1924.
Pearl J. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann.
Poole D. (2003). First-order probabilistic inference. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence IJCAI'03, Acapulco, Mexico, Morgan Kaufmann Publishers Inc.. vol. 3, pp. 985–991.
Poon H., Domingos P. (2006). Sound and efficient inference with probabilistic and deterministic dependencies. In: Proceedings of the 21st national conference on Artificial intelligence, July 16-20, Boston, Massachusetts, AAAI Press. vol. 1, pp. 458–463.
Poon H., Domingos P., Sumner M. (2008). A general method for reducing the complexity of relational inference and its application to mcmc. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, Chicago, Illinois, July 13-17, AAAI Press. pp. 1075–1080.
Potetz B. (2007). Efficient belief propagation for vision using linear constraint nodes. In: Proceeding of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07), Minneapolis, MN, USA, IEEE computer society. pp. 1–8.
Richardson M., Domingos P. (2006). Markov logic networks. *Machine Learning* 62(1-2):107–136.
Roosta T., Wainwright M. J., Sastry S. S. (2008). Convergence analysis of reweighted sum-product algorithms. *IEEE Transactions on Signal Processing* 56(9):4293–4305.
Rossi F., Van Beek P., Walsh T. (2006). Handbook of constraint programming. Elsevier.
Sen P., Deshpande A., Getoor L. (2009). Bisimulation-based approximate lifted inference. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, Canada, June 18-21, AUAI Press, pp. 496–505.
Shavlik J., Natarajan S. (2009). Speeding up inference in markov logic networks by preprocessing to reduce the size of the resulting grounded network. In: Proceedings of the 21 International Joint Conference on Artificial Intelligence, Pasadena, California, USA, IJCAI Organization. pp. 1951–1956.
Shi X., Schonfeld D., Tuninetti D. (2010). Message error analysis of loopy belief propagation. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP 2010), March 14-19, Dallas, Texas, USA, IEEE computer society. pp. 2078–2081.
Singla P. (2012). Markov logic networks: theory, algorithms and applications. In: Proceedings of the 18th International Conference on Management of Data, Computer Society of India, pp. 15–150.
Singla P., Domingos P. (2006). Entity resolution with markov logic. In: Proceedings of the Sixth International Conference on Data Mining, ICDM'06, Hong Kong, China, 1822 December, IEEE computer society. pp. 572–582.
Singla P., Domingos P. (2008). Lifted first-order belief propagation. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, Chicago, Illinois, July 1317, AAAI Press. pp. 1094–1099.
Singla P., Nath A., Domingos P. (2010). Approximate lifted belief propagation. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, Georgia, USA, July 11-15, 2010, AAAI Press. pp. 92–97.
Smith D., Gogate V. (2014). Loopy belief propagation in the presence of determinism. In: Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, April 22-25, Reykjavik, Iceland, JMLR: W & CP. vol. 33, pp. 895–903.

Venugopal D., Gogate V. (2014a). Evidence-based clustering for scalable inference in markov logic. In: Proceedings of the 7th European Conference on machine learning and data mining conference (ECML PKDD 2014), Nancy, France, September 15-19, Springer. pp. 258–273.

Venugopal D., Gogate V. G. (2014b). Scaling-up importance sampling for markov logic networks. In: Proceedings of the 28th Conference on Neural Information Processing Systems, 8-13 December, Montreal, Canada: Advances In Neural Information Processing Systems 27 (NIPS 2014), Curran Associates Inc.. pp. 2978–2986.

Wainwright M., Jordan M. (2003). Semidefinite relaxations for approximate inference on graphs with cycles. In: Proceedings of the 17th conference on Neural Information Processing Systems: Advances in neural information processing systems 16 (NIPS-2003), MIT Press. pp. 369–376.

Wainwright M., Jaakkola T., Willsky A. (2003). Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory* 49(5):1120–1146.

Wei W., Erenrich J., Selman B. (2004). Towards efficient sampling: Exploiting random walk strategies. In: Proceedings of the Nineteenth National Conference On Artificial Intelligence, July 25-29, San Jose, California, USA, AAAI Press. vol. 4, pp. 670–676.

Weinman J. J., Tran L. C., Pal C. J. (2008). Efficiently learning random fields fo stereo vision with sparse message passing. In: Proceedings of the 10th European Conference on Computer Vision, Marseille, France, Springer. pp. 617–630.

Winn J. M. (2004). Variational message passing and its applications. PhD thesis, University of Cambridge.

Winn J. M., Bishop C. M. (2005). Variational message passing. *Journal of Machine Learning Research* 6:661–694.

Yeang C.-H. (2010). Exact loopy belief propagation on euler graphs. In: Proceedings of the 12th International Conference on Artificial Intelligence, Las Vegas, Nevada, USA, July 12-15, CSREA Press. pp. 471–477.

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

Yedidia J. S., Freeman W. T., Weiss Y. (2003). Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium* 8:236–239.

Yuille A. L. (2001). A double-loop algorithm to minimize the bethe free energy. In: Proceedings of the Third International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, NRIA Sophia-Antipolis, France, September 3-5, Springer-Verlag. pp. 3–18.

Yuille A. L. (2002). Cccp algorithms to minimize the bethe and kikuchi free energies: Convergent alternatives to belief propagation. *Neural computation* 14(7):1691–1722.