Titre: Title:	Using Constraint Satisfaction Techniques and Variational Methods for Probabilistic Reasoning
Auteur: Author:	Mohamed Ibrahim
Date:	2015
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Ibrahim, M. (2015). Using Constraint Satisfaction Techniques and Variational Methods for Probabilistic Reasoning [Ph.D. thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/1864/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/1864/
Directeurs de recherche: Advisors:	Gilles Pesant, & Christopher J. Pal
Programme: Program:	Génie informatique

UNIVERSITÉ DE MONTRÉAL

USING CONSTRAINT SATISFACTION TECHNIQUES AND VARIATIONAL METHODS FOR PROBABILISTIC REASONING

MOHAMED IBRAHIM
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INFORMATIQUE)
AOÛT 2015

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

USING CONSTRAINT SATISFACTION TECHNIQUES AND VARIATIONAL METHODS FOR PROBABILISTIC REASONING

présentée par: IBRAHIM Mohamed
en vue de l'obtention du diplôme de: Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de:

M. MULLINS John, Ph. D., président
M. PESANT Gilles, Ph. D., membre et directeur de recherche
M. PAL Christopher J., Ph. D., membre et codirecteur de recherche
M. DESMARAIS Michel C., Ph. D., membre
M. VAN BEEK Peter, Ph. D., membre externe

DEDICATION

To my beloved mother, the memory of my father and to my lovely wife and children, with endless love and gratitude

ACKNOWLEDGEMENTS

First and foremost, I sincerely offer praise to ALLAH for providing me the health, patience, and knowledge to complete my thesis.

I would like to express my deepest gratitude to my supervisors, Prof. Gilles Pesant and Prof. Christopher Pal, whose expertise, understanding, and patience, added considerably to my graduate experience. I am grateful for everything I learned from them, for their continuous support throughout my study and research, and for providing me with invaluable insights which helped me solve many of the problems I encountered in my research. It was simply impossible producing this thesis without their excellent guidance.

I would like to take this as an opportunity to thank Prof. John Mullins, Prof. Michel Desmarais, and Prof. Peter Van Beek for assigning part of their time to review this dissertation and to serve on my thesis committee.

I would like to acknowledge the Ministry of Higher Education, Egypt and my supervisors for respectively supporting me financially during my Ph.D. which gave me opportunities to complete this thesis.

I also thank all my colleagues, friends, and all the selfless and active workers serving society as well.

Last but certainly not least, I am very thankful to my family: To my mother, Hameeda Ali-Eldin, who somehow convinced me that ‘failure’ is not a dictionary word; to my father, Hamza Ibrahim to whom I never managed to explain what I do but who is always happy for my success, my sister, Marwa Ibrahim, my wife, Zolfa Selmi, and my children, Youssef, Seif and Habiba for their support and encouragement. I love you all so much!

RÉSUMÉ

Cette thèse présente un certain nombre de contributions à la recherche pour la création de systèmes efficaces de raisonnement probabiliste sur les modèles graphiques de problèmes issus d'une variété d'applications scientifiques et d'ingénierie. Ce thème touche plusieurs sous-disciplines de l'intelligence artificielle. Généralement, la plupart de ces problèmes ont des modèles graphiques expressifs qui se traduisent par de grands réseaux impliquant déterminisme et des cycles, ce qui représente souvent un goulot d'étranglement pour tout système d'inférence probabiliste et affaiblit son exactitude ainsi que son évolutivité.

Conceptuellement, notre recherche confirme les hypothèses suivantes. D'abord, les techniques de satisfaction de contraintes et méthodes variationnelles peuvent être exploitées pour obtenir des algorithmes précis et évolutifs pour l'inférence probabiliste en présence de cycles et de déterminisme. Deuxièmement, certaines parties intrinsèques de la structure du modèle graphique peuvent se révéler bénéfiques pour l'inférence probabiliste sur les grands modèles graphiques, au lieu de poser un défi important pour elle. Troisièmement, le re-paramétrage du modèle graphique permet d'ajouter à sa structure des caractéristiques puissantes qu'on peut utiliser pour améliorer l'inférence probabiliste.

La première contribution majeure de cette thèse est la formulation d'une nouvelle approche de passage de messages (message-passing) pour inférer dans un graphe de facteurs étendu qui combine des techniques de satisfaction de contraintes et des méthodes variationnelles. Contrairement au message-passing standard, il formule sa structure sous forme d'étapes de maximisation de l'espérance variationnelle. Ainsi, on a de nouvelles règles de mise à jour des marginaux qui augmentent une borne inférieure à chaque mise à jour de manière à éviter le dépassement d'un point fixe. De plus, lors de l'étape d'espérance, nous mettons à profit les structures locales dans le graphe de facteurs en utilisant la cohérence d'arc généralisée pour effectuer une approximation de champ moyen variationnel.

La deuxième contribution majeure est la formulation d'une stratégie en deux étapes qui utilise le déterminisme présent dans la structure du modèle graphique pour améliorer l'évolutivité du problème d'inférence probabiliste. Dans cette stratégie, nous prenons en compte le fait que si le modèle sous-jacent implique des contraintes inviolables en plus des préférences, alors c'est potentiellement un gaspillage d'allouer de la mémoire pour toutes les contraintes à l'avance lors de l'exécution de l'inférence. Pour éviter cela, nous commençons par la relaxation des préférences et effectuons l'inférence uniquement avec les contraintes inviolables. Cela permet d'éviter les calculs inutiles impliquant les préférences et de réduire la taille effective du réseau graphique.

Enfin, nous développons une nouvelle famille d'algorithmes d'inférence par le passage de messages dans un graphe de facteurs étendus, paramétrées par un facteur de lissage (smoothing parameter). Cette famille permet d'identifier les épines dorsales (backbones) d'une grappe qui contient des solutions potentiellement optimales. Ces épines dorsales ne sont pas seulement des parties des solutions optimales, mais elles peuvent également être exploitées pour intensifier l'inférence MAP en les fixant de manière itérative afin de réduire les parties complexes jusqu'à ce que le réseau se réduise à un seul qui peut être résolu avec précision en utilisant une méthode MAP d'inférence classique. Nous décrivons ensuite des variantes paresseuses de cette famille d'algorithmes.

Expérimentalement, une évaluation empirique approfondie utilisant des applications du monde réel démontre la précision, la convergence et l'évolutivité de l'ensemble de nos algorithmes et stratégies par rapport aux algorithmes d'inférence existants de l'état de l'art.

ABSTRACT

This thesis presents a number of research contributions pertaining to the theme of creating efficient probabilistic reasoning systems based on graphical models of real-world problems from relational domains. These models arise in a variety of scientific and engineering applications. Thus, the theme impacts several sub-disciplines of Artificial Intelligence. Commonly, most of these problems have expressive graphical models that translate into large probabilistic networks involving determinism and cycles. Such graphical models frequently represent a bottleneck for any probabilistic inference system and weaken its accuracy and scalability.

Conceptually, our research here hypothesizes and confirms that: First, constraint satisfaction techniques and variational methods can be exploited to yield accurate and scalable algorithms for probabilistic inference in the presence of cycles and determinism. Second, some intrinsic parts of the structure of the graphical model can turn out to be beneficial to probabilistic inference on large networks, instead of posing a significant challenge to it. Third, the proper re-parameterization of the graphical model can provide its structure with characteristics that we can use to improve probabilistic inference.

The first major contribution of this thesis is the formulation of a novel message-passing approach to inference in an extended factor graph that combines constraint satisfaction techniques with variational methods. In contrast to standard message-passing, it formulates the Message-Passing structure as steps of variational expectation maximization. Thus it has new marginal update rules that increase a lower bound at each marginal update in a way that avoids overshooting a fixed point. Moreover, in its expectation step, we leverage the local structures in the factor graph by using generalized arc consistency to perform a variational mean-field approximation.

The second major contribution is the formulation of a novel two-stage strategy that uses the determinism present in the graphical model's structure to improve the scalability of probabilistic inference. In this strategy, we take into account the fact that if the underlying model involves mandatory constraints as well as preferences then it is potentially wasteful to allocate memory for all constraints in advance when performing inference. To avoid this, we start by relaxing preferences and performing inference with hard constraints only. This helps avoid irrelevant computations involving preferences, and reduces the effective size of the graphical network.

Finally, we develop a novel family of message-passing algorithms for inference in an extended factor graph, parameterized by a smoothing parameter. This family allows one to find the "backbones" of a cluster that involves potentially optimal solutions. The cluster's backbones are not only portions of the optimal solutions, but they also can be exploited for scaling MAP inference by iteratively fixing them to reduce the complex parts until the network is simplified into one that can be solved accurately using any conventional MAP inference method. We then describe lazy variants of this family of algorithms. One limiting case of our approach corresponds to lazy survey propagation, which in itself is novel method which can yield state of the art performance.

We provide a thorough empirical evaluation using real-world applications. Our experiments demonstrate improvements to the accuracy, convergence and scalability of all our proposed algorithms and strategies over existing state-of-the-art inference algorithms.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

BP	Belief Propagation
CCC	Double loop (Concave-Convex) Procedure
CLL	average Conditional Log marginal-Likelihood
CS	Constraint Satisfaction
GAC	Generalized arc Consistency
GEM-MP	Generalized arc consistency Expectation Maximization Message Passing
Gibbs	MCMC (Gibbs Sampling)
KL	The average Kullback-Leibler divergence
LBP	Loopy Belief Propagation
L-IM	Lifted MCMC (Importance Sampling)
L2-Convex	Sequential Message Passing on the Convex-L2 Bethe Free Energy
MAP	Maximum-A-Posteriori Inference
MAR	Marginal Inference
MAX-SAT	Maximum Satisfiablility Problem
MRF	Markov Random Field
MF	Mean Field Approximation
MLN	Markov Logic Network
ML	Machine Learning
pAc	Probabilistic Arc-Consistency
PR	Preference Relaxation
PGAC	Probabilistic Generalized Arc-Consistency
PGM	Probabilistic Graphical Model
PG	Propositional Grounding
SAT	Propositional Satisfiability Problem
SP	Survey Propagation
SRL	Statistical Relational Learning
RBP	Residual Belief Propagation
VEM	Variational Expectation Maximization
WSP-χ	Family of Weighted Survey Propagation Algorithms
WSP-Dec	WSP-χ Inspired Decimation

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

We provide here an overview of our research and the motivation behind it. Next, we explain our research problems and formalize our objectives through some concise research questions. We then summarize our contributions and conclude with an outline of this thesis.

1.1 Overview and Motivations

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 scalable inference within such graphical models is thus an important challenge 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) is a powerful formalism for graphical models that has made significant progress towards the goal of combining the powers of both first-order logic and probability. This allows one to address relational and uncertain dependencies in data. However, for the second part of the goal, and in practice, inference always tends to give poor results in the presence of both cycles and determinism and can be problematic for learning when using it as a subroutine.

Furthermore, commonly in SRL models it has been convenient to convert formulas to conjunctive normal form (CNF) and to propositionalize the theory to a grounded network of clauses wherein any probabilistic inference can be applied for reasoning about uncertain queries. In such cases, the logical structure within the network can be problematic for certain inference procedures and often represents a major bottleneck when faced with instances close to the satisfiability threshold. This situation in fact frequently occurs in SRL instances of real-world applications. In addition, this approach can be very time consuming: the grounded network is typically large, which slows down inference, and this also can be problematic, especially for the scalability of inference.

1.2 Problem Statement and Limitations

Our dissertations here revolve around improving the accuracy and scalability of inference for three problems related to computing marginal probabilities (marginals for short) and MAP assignments on graphical models that involve cycles and determinism within an underlying probabilistic network of a size such that inference in the network poses problems for all known methods.

1.2.1 Problem 1

To compute marginals, 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.*

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. It is known that the local optima of the Bethe free energy correspond to fixed points of LBP, and it has been proven that violating the uniqueness condition for the Bethe free energy generates several fixed points in the space of LBP's marginal distributions (Heskes, 2004; Yedidia *et al.*, 2005). A graph involving a single cycle has a unique fixed point and usually guarantees the convergence of LBP (Heskes, 2004). From the viewpoint of local search, LBP performs a gradient-descent/ascent search over the marginal space, endeavoring to converge to a fixed point (i.e., 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 (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 (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). The solid black curves in Figure 1.1 are empirical examples that depict the non-convergence of LBP due to cycles.

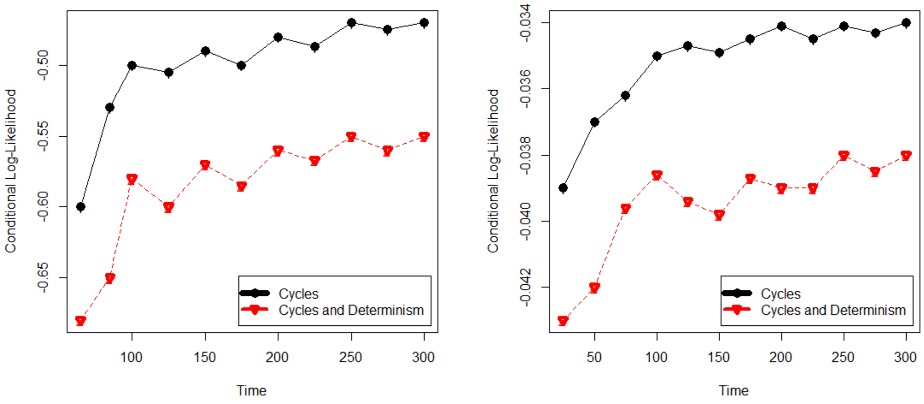

Figure 1.1 A preliminary experiment illustrates the effects of cycles and determinism on the convergence behaviour of LBP in Cora dataset (Singla and Domingos, 2006a) (*left*) and Yeast dataset (Singla and Domingos, 2006a) (*right*). These datasets and their models are publicly available at the alchemy website: http://alchemy.cs.washington.edu/data/

Determinism also plays a substantial role in reducing the effectiveness of LBP (Heskes, 2004). 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 fixed point (see pages 413-429 of Koller and Friedman, 2009, for more details). The solid red curves in Figure 1.1 illustrate empirically the increase in the non-convergence of LBP due to cycles and determinism. 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).

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.

1.2.2 Problem 2

It is common for many real-world problems to have expressive graphical models that combine deterministic and probabilistic dependencies. In the world of SRL, the former often appear in the form of mandatory (i.e. hard) constraints that must be satisfied in any world with a non-zero probability. The latter are typically formulated as preferences (i.e. soft constraints), and dissatisfying them is not impossible, but less probable. Thus, if a query atom X which is involved with a set of constraints $\mathcal{C} = \{\mathcal{H}, \mathcal{S}\}$ (where $\mathcal{H}$ and $\mathcal{S}$ are its subsets of hard and soft constraints respectively) violates one of the hard constraints h in $\mathcal{H}$, then its marginal probability will be zero, even if it satisfies its other hard and soft constraints in $\mathcal{C} - \{h\}$. A variable violates a hard constraint if there is a truth value for that variable such that the constraint is violated in all possible worlds consistent with that truth value assignment to the variable and the evidence. Current inference approaches do not exploit the fact that there is no need to consider irrelevant computations (and memory usage) with soft constraints $\mathcal{S}$, since using only hard constraints $\mathcal{H}$ should be sufficient to compute that its marginal probability $P(X)$ is zero.

Limitation 3. *In relational domains, where we have millions of query atoms each involved with thousands of constraints, such irrelevant computations greatly weaken the scalability of the inference, especially if most query atoms have a tendency to violate hard constraints.*

Inspired by the sparseness property of relational domains, the marginal probability of the vast majority of query atoms being true is frequently zero. Potentially this is due to the violation of hard constraints that have at least one false precondition query atom.

1.2.3 Problem 3

To compute MAP assignments, it has been proven that maximum-a-posteriori (MAP) inference is equivalent to solving a weighted MAX-SAT problem (i.e., where one finds the most probable truth assignment or "MAP" solution that maximizes the total weight of satisfied clauses) (Park, 2002). A simple approach to tackle MAP inference is to use off-the-shelf local search algorithms that are designed to efficiently solve weighted MAX-SAT instances. For example MaxWalkSAT (Kautz *et al.*, 1997; Selman *et al.*, 1993) was recently applied as a conventional MAP inference method in probabilistic graphical models (Richardson and Domingos, 2006). However, it is known that for real world applications, and in relational domains, models are normally translated into grounded networks featuring high densities.[1] From the point of view of satisfiability, if the density of the grounded network is close to a satisfiability threshold, then the (MAP) assignments in the solution space are clustered (Mann and Hartmann, 2010). This clustering means that the (MAP) assignments belonging to the same cluster are close to each other (e.g., in terms of Hamming distance). It has been shown that such clustering in the solution space exists not only for uniform random satisfiability, but also for some structured satisfiability instances that follow realistic and natural distributions (Hartmann and Weigt, 2006; Zhang, 2004; Gomes *et al.*, 2002; Parkes, 1997), similar to real-world problems that are modeled as MLNs (e.g., social networks, as shown by Kambhampati and Liu (2013)). In general, using local search algorithms for MAP inference frequently has the following limitations due to the clustering of the solution space:

Limitation 4. *Usually, the existence of many clusters is an indication of a rugged energy landscape, which then also gives rise to many local optima. This often hinders the performance of most local search algorithms because they can get stuck in a local optimum (Montanari et al., 2004).*

Limitation 5. *Another possible consequence of the clustering of the solution space is that the search space fractures dramatically with a proliferation of 'metastable' clusters (Chavas et al., 2005). This acts as a dynamic trap for local search algorithms, including MaxWalkSAT, since they can get stuck in one such 'metastable' cluster at a local optimum (Kilby et al., 2005; Chavas et al., 2005). On the flip side of the issue, if the clauses capture complex structures (e.g., relational dependencies) then the clusters tend to decompose into an exponentially small fraction of (MAP) assignments (Mann and Hartmann, 2010). So there is not even any guarantee of getting into a cluster that contains an optimal solution. Clearly, this all greatly weakens the possibility that a local search can converge to an optimal solution, particularly for SRL applications (Riedel, 2008).*

It is well known that techniques such as marginalization-decimation based on the max-product BP algorithm (Wainwright *et al.*, 2004; Weiss and Freeman, 2001), can be used to help local search find an optimal MAP solution, but it is also believed that BP does not

[1] The density is the ratio of ground clauses to ground atoms.

converge due to strong attraction in many directions (Kroc *et al.*, 2009; Khosla *et al.*, 2009). That is, max-product BP fails because its local computations may obtain locally optimal assignments corresponding to different clusters, but these cannot be combined to find a global MAP solution (Kroc *et al.*, 2009).

Fortunately, it is likely that in each cluster there is a certain fraction of 'frozen' ground atoms (Achlioptas and Ricci-Tersenghi, 2009) that are fixed in all (MAP) solutions within the cluster, while the others can be varied subject to some ground clauses. These frozen ground atoms are known as 'cluster backbones' (Kroc *et al.*, 2008). Thus, one relatively crude but useful way to obtain a portion of a solution in the cluster is by finding the cluster backbones.

1.3 Research Questions and Objectives

We formalize our objectives through some concise research questions as follows:

- To address Limitations 1 and 2 of LBP discussed above, our first objective is to answer the following research questions:
    - **(RQ1)** Does updating the marginals such that we do not overshoot the nearest fixed point diminish the threat of non-convergence of LBP?
    - **(RQ2)** Are constraint satisfaction techniques able to help address the challenges resulting from determinism in the graphical models?
- To address Limitation 3, and avoid irrelevant computations involving preferences when performing inference, our second objective is to answer the following research question:
    - **(RQ3)** Can relaxing soft constraints or preferences be helpful to avoid irrelevant computations involving preferences? That is to say, can performing the inference by using only hard constraints (determinism) be exploited to reduce problem size?
- To address Limitations 4 and 5 of local search algorithms, which are due to search space clustering, our third objective is to answer the following research questions:
    - **(RQ4)** Can we derive a method to identify backbones in a cluster involving potentially optimal MAP solutions?
    - **(RQ5)** Can the cluster backbones be used to scale MAP inference?
    - **(RQ6)** Can the cluster backbones be helpful to reach optimal MAP solutions?

1.4 Summary of the Contributions

Through the research presented in this thesis, we achieve our objectives by answering the research questions above. This has led to our main contributions summarized as follows:

- **Generalized arc-consistency Expectation-Maximization Message-Passing (GEM-MP), a novel message-passing algorithm for applying variational approximate inference to graphical models.**

  In GEM-MP, we first re-parameterized the factor graph in such a way that the inference task (that could be performed by LBP inference on the original factor graph) is equivalent to a variational EM procedure. Then, we take advantage of the fact that LBP and variational EM can be viewed in terms of different types of free energy minimization equations. We formulate our Message-Passing structure as the E and M steps of a variational EM procedure (Beal and Ghahramani, 2003; Neal and Hinton, 1999). This variational formulation leads to the synthesis of new rules that update marginals by maximizing a lower bound of the model evidence such that we never overshoot the model evidence (*Answering research question RQ1*). In addition, in the corresponding Expectation step of GEM-MP, the constructed expected log marginal-likelihood has been defined according to the posterior distribution over local entries of the logical clauses that define factors. This enables us to exploit their logical structures by applying a generalized arc-consistency concept (Rossi *et al.*, 2006), and to use that to perform a variational mean-field approximation when updating the marginals. This significantly amends smoothing out the marginals to converge correctly to a stable convergent fixed point in the presence of determinism (*Answering research question RQ2*). Our experiments on real-world problems demonstrate the increased accuracy and convergence of GEM-MP compared to existing state-of-the-art inference algorithms such as MC-SAT, LBP, and Gibbs sampling, and convergent message passing algorithms such as the Concave-Convex Procedure (CCCP), Residual BP, and the L2-Convex method.

- **Preference Relaxation (PR), a new two-stage strategy that uses the determinism (i.e., hard constraints) present in the underlying model to improve the scalability of relational inference.**

  The basic idea of PR is to diminish irrelevant computational time and memory, which are due to preference constraints. To do so, in a first stage PR starts by relaxing preferences and performing inference with hard constraints only in order to obtain the zero marginals for the query ground atoms that violate the hard constraints. It then filters these query atoms (i.e., it removes them from the query set) and uses them to enlarge the evidence database. Then in a second stage preferences are reinstated and inference is performed on a grounded network that is constructed based on both a filtered query set and an expanded evidence database obtained in the first stage. PR substantially reduces the effective size of the constructed grounded network, potentially with a loss of accuracy. (*Answering research question RQ3*). Experiments on real-world applications show how this strategy substantially scales inference with a minor impact on accuracy.

- **Weighted Survey Propagation-inspired Decimation (WSP-Dec), a novel family of message passing algorithms for applying MAP inference to graphical models.**

  In WSP-Dec, we first describe a novel family of extended factor graphs, specified by the parameter $\chi \in [0, 1]$. These factor graphs are re-parameterized in such a way that they define positive joint probabilities over max-cores. These max-cores are natural interpretations of core assignments (Maneva *et al.*, 2007) that satisfy a set of clauses with maximal weights. We then show that applying BP message-passing to this family recovers a family of Weighted SP algorithms (WSP-χ), ranging from pure Weighted SP (WSP-1) to standard max-product BP (WSP-0). The magnetization of the marginals computed by any WSP-χ algorithm can be used to obtain backbones (i.e., frozen ground atoms) of a cluster involving potentially optimal MAP solutions (*Answering research question RQ4*). The cluster backbones are not only a portion of the optimal solutions in the cluster, but they can also be used to enlarge the evidence database and shrink the query set. Therefore, iteratively fixing them results in a reduction of the complex parts of the grounded network, which afterwards can be simplified to a scalable one that is then solved accurately using any conventional MAP method (*Answering research question RQ5*). Hence, integrating WSP-χ as pre-processing with a MaxWalkSAT algorithm in a decimation procedure produces a family of Weighted Survey Propagation-inspired decimation (WSP-Dec) algorithms for applying MAP inference to SRL models (*Answering research question RQ6*). Our experiments on real-world applications show the promise of WSP-χ to improve the accuracy and scalability of MAP inference when integrated with local search algorithms such as MaxWalkSAT.

- **Lazy-WSP-χ, a novel lazy variant of weighted Survey Propagation for efficient relational MAP inference.**

  In Lazy-WSP-χ, we start by grounding the network lazily, and maintaining only active ground clauses and their active ground atoms that are sufficient to answer the queries. We then call WSP-χ to scale the lazy ground network, which was built using those active clauses and atoms, by fixing the frozen active atoms. Thus, Lazy-WSP-χ mainly differs from WSP-χ in both the initial set of underlying query atoms and clauses. Our experiments on real-world applications show how using the lazy variants of WSP-χ greatly improves the scalability of MAP inference.

- **Lazy-PR-based inference, a hybrid inference approach that combines PR-based inference with lazy inference to greatly improve the scalability of marginal inference.**

  One key advantage of our proposed PR strategy is that it can be combined with other state-of-the-art approaches which improve the scalability of inference, such as Lazy and Lifted. In Lazy-PR-based inference, we start by maintaining only the active-awake hard constraints and their atoms. We then call the relational inference on the ground network, which was built using those maintained clauses and atoms. After reaching a fixed point, we filter active-awake query atoms and enlarge the evidence database. The experimental evaluations on real-world applications show that hybridizing Lazy-PR-based inference with relational inference algorithms improves their efficiency for computing marginals.

Through our experimental evaluations in this thesis, we have focused on Markov logic and Ising models, but it is worth noting that all the main proposed techniques (GEM-MP, PR and WSP-χ) and their lazy variants are applicable to other representations of SRL models, including standard graphical models defined in terms of factor graphs.

1.5 Organization of the Dissertation

The remainder of this thesis is organized as follows.

Part I is about the state of the art, and it consists of two chapters:

- **Chapter 2** presents the necessary background of the techniques and methods useful to understand this thesis.
- **Chapter 3** presents a thorough discussion for the related work relevant to this thesis.

Part II consists of three chapters in which we present our contributions:

- **Chapter 4** demonstrates our **G**eneralized arc-consistency **E**xpectation-**M**aximization **M**essage-**P**assing (GEM-MP) for applying variational approximate inference to graphical models in the presence of cycles and determinism. The contents of this chapter are largely extracted from our paper:

Mohamed-Hamza Ibrahim, Christopher Pal and Gilles Pesant. "Improving Message-Passing Inference in the Presence of Determinism and Cycles". Currently under review in the Machine Learning Journal (MLJ).

- **Chapter 5** explains our **P**reference **R**elaxation (PR) strategy that uses the determinism present in the underlying model to improve the scalability of relational inference. It also demonstrates how to combine PR-based inference with lazy inference. The contents of this chapter are based on our paper:

  Mohamed-Hamza Ibrahim, Christopher Pal and Gilles Pesant. "Exploiting determinism to scale relational inference". Published in Proceedings of the Twenty-Ninth National Conference on Artificial Intelligence (AAAI'15) in 2015 (Ibrahim *et al.*, 2015).

- **Chapter 6** demonstrates our **W**eighted **S**urvey **P**ropagation-inspired Decimation (WSP-Dec), a family of algorithms to handle MAP inference in the presence of clustered search space. It also explains lazy variants of this family of algorithms. The contents of this chapter are largely extracted from our paper:

  Mohamed-Hamza Ibrahim, Christopher Pal and Gilles Pesant. "Exploiting Cluster Backbones to Improve MAP Inference in Relational Domains". Submitted to Journal of Artificial Intelligence Research (JAIR).

Chapter 7 concludes the thesis by revisiting our major contributions and sketching our possible opportunities for future research directions.

Appendix A presents the proofs of Theorems and Propositions.

CHAPTER 2 BACKGROUND

To set the stage for this thesis, in this chapter we review basic concepts and methods that are necessary to understand our research by using an explanatory example, presented in Table 2.1. This example is an excerpt of the knowledge base for the Cora dataset (Singla and Domingos, 2006a). Suppose that we are given a citation database in which each citation has author, title, and venue fields. We wish 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, suppose that our goal will be to predict the *SameBib* ground atoms' marginals and compute their most probable truth assignment (i.e., MAP solution). The first part of the goal refers to the marginal inference problem and the second refers to the maximum-a-posteriori (MAP) inference problem. At this point, let us first express our prerequisite basic definitions.

Table 2.1 An excerpt of the knowledge base 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 represent authors and r_1, r_2, r_3 represent citations.

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

2.1 Basic Notation and Definitions

A first-order knowledge base (KB) is a set of formulas in first-order logic (FOL).

Definition 1 (First-Order Logic (FOL))**.** *The set of terms of FOL (also known as first-order predicate calculus) is defined by the following rules:*

- *A variable is a term*
- *If h is a function symbol with $t_1, \ldots, t_n$ terms and $n \geq 0$, then $h(t_1, \ldots, t_n)$ is a term.*

- *If P is a predicate symbol with $t_1, \ldots, t_n$ terms and $n \geq 0$, then $P(t_1, \ldots, t_n)$ is an atomic statement.*

In FOL, each formula is a sentential formula (e.g., $\forall X \ \exists Z \ f(X, Z)$), where $\forall$ is the universal quantifier, and $\exists$ is the existential quantifier. $f(X, Z)$ is the scope of the respective quantifier, and any occurrence of variable X or Z in the scope of a quantifier is bound by the closest $\forall X$ or $\exists Z$.

The set of sentential formulas of first-order predicate calculus is defined by the following rules:

- *Any atomic statement is a sentential formula*
- *If f_1 and f_2 are sentential formulas, then $\neg f_1$, $f_1 \wedge f_2$, $f_1 \vee f_2$, and $f_1 \Rightarrow f_2$ (f_1 implies f_2) are sentential formulas.*

In formulas of first-order predicate calculus, all variables are object variables serving as arguments of functions and predicates.

Definition 2 (Conjunctive Normal Form (CNF))**.** *A formula is in conjunctive normal form (or clausal normal form) if it is represented as a conjunction of clauses, where a clause is a disjunction of literals-* ***a literal*** *represents either a variable or its negation.*

For the purpose of probabilistic (and automated) inference, it is often convenient to convert FOL formulas to a clausal form (CNF).[1] For instance, as shown in Table 2.1, we convert FOL formulas of the explanatory example into a clausal form (CNF).[2] This is known as propositional grounding.

Definition 3 (Propositional grounding (PG))**.** *PG is the process of replacing a first-order Knowledge Base (KB) by an equivalent propositional one (Richardson and Domingos, 2006). In finite domains, inference over a first-order KB can be performed by propositional grounding followed by satisfiability testing. But in order to apply a satisfiability solver we need to create a Boolean variable for every possible grounding of every predicate in the domain and a propositional clause for every grounding of every first-order clause.*

[1] This includes the removal of the existential quantiers by Skolemization, which is not sound in general. However, in finite domains an existentially quantied formula can simply be replaced by a disjunction of its groundings.

[2] Note that the weights that are associated with the FOL formulas in Table 2.1 will be used later to extend the FOL formulation with Markov Logic– one of the most powerful graphical model representations – as will be discussed in Section 2.2.2

After propositional grounding, we get a formula $\mathcal{F}$, which is a conjunction of m ground clauses. We use $f_i \in \mathcal{F}$ 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. Each f_i is associated with the pair $(\hat{w}_i, \mathcal{X}_{f_i})$, where $\hat{w}_i$ is its weight and the set $\mathcal{X}_{f_i} \subseteq \mathcal{X}$ corresponds to the variables appearing in its scope. Both "+" and "−" will be used to denote the positive (true) and negative (false) appearance of the ground atoms. A joker "$*$" will be used to denote the "don't care" state of a variable X_j. That is, the assignment $X_j = *$ means that the variable X_j is free to take either "+" or "−" since the clauses containing it should be already satisfied by other variables, and the value of X_j does not matter. 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 $\left|\mathcal{F}_{X_j}\right|$. 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. $\mathcal{B}_{\mathcal{X}}$ denotes the whole set of $\mathcal{X}$'s marginals.

Definition 4. *For a ground clause f_i, we define $s_{i,j}$ (resp. $u_{i,j}$) as the value of a ground atom $X_j \in \{1, 0\}$ that satisfies (resp. violates) f_i. Therefore, f_i is satisfied if and only if at least one of its ground atoms X_j is equal to $s_{i,j}$.*

$\mathcal{X}^s_{f_i}$ and $\mathcal{X}^u_{f_i}$ are used to denote the two disjoint subsets of the ground atoms that are satisfying and violating f_i, respectively. Now from the above definition, we have the following sets:

$$\mathcal{F}_{X_j+} = \{f_i \in \mathcal{F}_{X_j}; s_{i,j} = 1\},\ \mathcal{F}_{X_j-} = \{f_i \in \mathcal{F}_{X_j-}; s_{i,j} = 0\} \tag{2.1a}$$

$$\mathcal{F}^s(j) = \{f_i \in \mathcal{F}_{X_j}; X_j = s_{i,j}\},\ \mathcal{F}^u(j) = \{f_i \in \mathcal{F}_{X_j}; X_j = u_{i,j}\} \tag{2.1b}$$

$$\mathcal{F}^s_{f_i}(j) = \{f_k \in \mathcal{F}_{X_j} \setminus \{f_i\}; s_{i,j} = s_{k,j}\},\ \mathcal{F}^u_{f_i}(j) = \{f_k \in \mathcal{F}_{X_j} \setminus \{f_i\}; s_{i,j} \neq s_{k,j}\} \tag{2.1c}$$

where we use $\mathcal{F}^s(j)$, $\mathcal{F}^u(j)$ to denote the two disjoint subsets of the ground clauses that are satisfied and violated by X_j, respectively. We use $\mathcal{F}^s_{f_i}(j)$ (resp. $\mathcal{F}^u_{f_i}(j)$) to denote the subset of ground clauses that agree (resp. disagree) with f_i about X_j.

Definition 5. *(Maneva et al., 2007) We say that a variable X_j is the unique satisfying variable for a clause $f_i \in \mathcal{F}$ if it is assigned $s_{i,j}$ whereas all other variables in the clause are assigned $u_{i,j}$. A variable X_j is constrained by the clause f_i if it is the unique satisfying* *variable for f_i. That is to say, it satisfies an indicator function as follows:*

$$CON_{i,j}(X_{f_i}) = \mathit{Ind}(X_j \text{ is the unique variable satisfying } f_i) \tag{2.2}$$

Where Ind(Predicate) returns 1 *if the Predicate is true and* 0 *otherwise. The variable X_j is said to be unconstrained if it has* 0 *or* 1 *value and it violates $CON_{i,j}(\mathcal{X}_{f_j})$ for each clause $f_i \in \mathcal{F}_{X_j}$ that involves it, or it has a joker $*$ value.*

Definition 6. *(Chieu and Lee, 2009) A complete assignment X:*

- *Satisfies a clause f_i if and only if (i) f_i contains a constrained variable X_j that is set to $s_{i,j}$, or (ii) f_i contains at least two unconstrained joker variables.*
- *Violates a clause f_i if and only if all its variables $X_j \in \mathcal{X}_{f_i}$ are set to $u_{a,i}$.*
- *Is invalid for a clause f_i if and only if exactly one variable X_j takes a joker value $*$ and all its other variables $X_k \in \mathcal{X}_{f_i} \setminus X_j$ are set to $u_{i,k}$.*

The complete assignment X is valid for a $\mathcal{F}$ if it is valid for all of its clauses. Note that the above definition of the invalid complete assignment reflects the interpretation that the joker value "$*$" is a "don't care" state for variable X_j: clauses involving a variable $X_j = *$ should be already satisfied by other variables, and the value of X_j does not matter. This in fact means that $X_j = *$ cannot be the last remaining possibility of satisfying any clause. So, if the other variables violate the clause and the remaining variable X_j is a joker, then the complete assignment is neither satisfying nor violating the clause because in this case we have two possibilities for $X_j = *$: one makes the complete assignment satisfying and the other makes it violating. Thus we say that the complete assignment is invalid. In the case where a clause contains two variables set to joker, the clause can be satisfied by either one of these two variables, so the other variable can take the "don't care" value.

2.2 Probabilistic Graphical Models

Now after expressing the basic notation and definitions, let us recall our goal of solving the marginal and MAP inference problems in the explanatory example. The first step, before carrying out the inference, is to represent the problem as a probabilistic graphical model (PGM). PGM is a graphical model that efficiently defines and describes a probabilistic model (e.g., joint distribution of the variables). PGMs are often classified into two major classes: Bayesian networks (BNs) and Markov random fields (MRFs).

Markov random fields (MRFs), also known as undirected graphical models, compactly represent a joint distribution over $\mathcal{X}$ as a product of potentials defined on subsets of variables (see Koller and Friedman, 2009). In order to motivate the use of MRFs, below we describe briefly two MRFs' models that will be used frequently throughout the thesis:

- Ising Models (Ising)
- Markov Logic Networks (MLNs)

2.2.1 Ising Models

The Ising model is an example of an MRF that arose from statistical physics. It was originally used for modeling the behavior of magnets. In particular, let $X_i \in \{1, 0\}$ represents the spin of an atom, which can either be spin down (i.e., $0 \downarrow$) or up (i.e., $1 \uparrow$). In some magnets, called ferro-magnets, neighboring spins tend to line up in the same direction, whereas in other kinds of magnets, called anti-ferromagnets, the spins want to be different from their neighbors. We can model this as an MRF by creating a graph in the form of a 2D or 3D lattice, and connect neighboring variables, as in Figure 2.1, by using a set of pairwise clique potentials:

$$\phi_{ij}(X_i, X_j) = \begin{cases} e^{\theta_{ij}} & \text{if } X_i = X_j \\ e^{-\theta_{ij}} & \text{Otherwise} \end{cases} \tag{2.3}$$

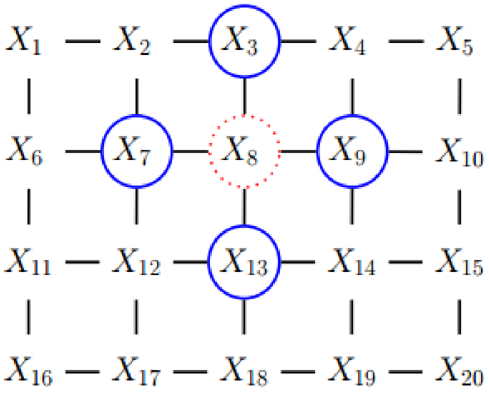

Figure 2.1 A 2D lattice represented as undirected graphical model. The red node X_8 is independent of the other black nodes given its neighbors (blue nodes)

The 2D Ising models with arbitrary topology are a specific subset of the canonical (pairwise) MRFs. Assume that $\mathcal{X} = \{X_1, \ldots, X_n\}$ is a set of binary random variables (representing the spins) that are Bernoulli distributed. In a 2D 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 given as

$$p(\mathcal{X} = x) = \mathcal{Z}_\theta^{-1}\, e^{\overbrace{\left[\sum_{(X_i,X_j)\in\mathcal{E}} \theta_{ij}\cdot X_i X_j + \sum_{X_i\in\mathcal{X}} \theta_i\cdot X_i\right]}^{\text{energy function}}} \tag{2.4a}$$

$$= \mathcal{Z}_\theta^{-1}\left[\prod_{(X_i,X_j)\in\mathcal{E}} \overbrace{e^{\theta_{ij}\cdot\mathbb{1}_{(X_i,X_j)}}}^{\phi_{ij}(X_i,X_j)}\right] \times \left[\prod_{X_i\in\mathcal{X}} \overbrace{e^{\theta_i\cdot X_i}}^{\phi_i(X_i)}\right] \tag{2.4b}$$

where $\mathcal{Z}_\theta$ is the normalizing constant. $\mathbb{1}_{(X_i,X_j)}$ is an indicator function that equals one when $X_i = X_j$. Otherwise it is equal to zero. Both $\{\theta_i\}$ and $\{\theta_{ij}\}$ are the parameters of univariate potentials $\{\phi_i(X_i)\}$ and pairwise potentials $\{\phi_{ij}(X_i, X_j)\}$, respectively. Typically, 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.

2.2.2 Markov Logic Networks

A Markov Logic Network (MLN) (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. With finite sets of constants that are domains to atoms, we can build a Markov logic network that has the following:

- One binary node for each possible grounding of each atom appearing in each clause. The value of the node is 1 if the ground atom is true, and 0 otherwise.

- One feature for each possible ground clause f_i. The value of this feature is 1 if the ground formula is true, and 0 otherwise. The weight of the feature is the weight w_i associated with f_i, where the weight attached to each ground clause reflects its strength dependency.

The power of MLNs appears in their 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*.

To understand the semantics of Markov logic, recall the explanatory example in Table 2.1. 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); and (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.[3]

2.2.3 Factor Graphs

For the purpose of solving inference problems, it is often convenient to convert MRFs into a different representation called factor graph.

Definition 7 (Factor Graphs (FG)). *A factor graph is a bipartite graph involving two types of nodes: variables and factors (i.e., constraints or clauses). Associated with each such node is a variable or a factor over variables, respectively. Typically, we will denote a factor graph $\mathcal{G} = (\mathcal{X}; \mathcal{F})$ in terms of its variables $\mathcal{X}$ and factors $\mathcal{F}$.*

Note that FG introduce additional nodes for representing the potentials of MRFs. This allows the model to be more explicit about the factorization when defining its joint distribution, which can be written as a product of factors as follows:

$$P(X_1, \cdots, X_n) = \frac{1}{Z} \prod_{f_i \in \mathcal{F}} f_i(\mathcal{X}_{f_i}) \tag{2.5}$$

Where Z is a normalization constant that enforces the factorization to represent a probability distribution that sums to one. Factor graphs unify directed and undirected graphs with the same representation. Thus they can be used to represent undirected graphs such as MRFs, where the factors will be the potential functions. If the factor graphs are being used to represent directed graphs like BNs, then the factors will be the local conditional distributions of each node, and here the extra normalization term in the joint distribution is not needed.

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

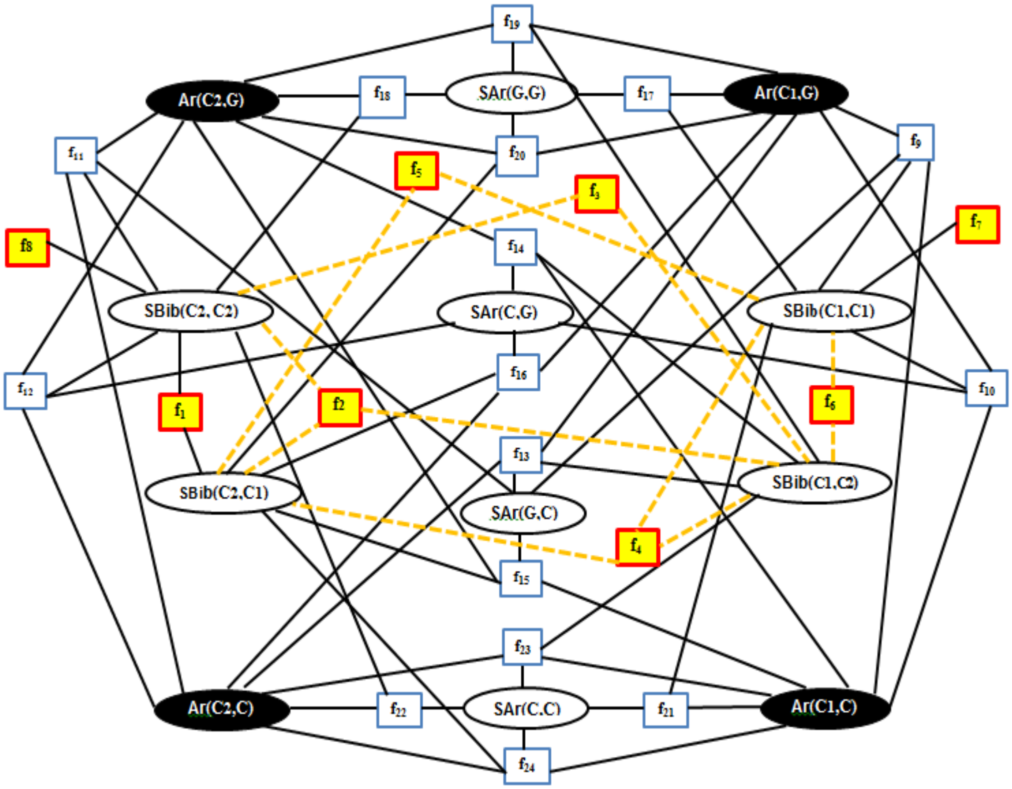

Figure 2.2 Grounded factor graph obtained by applying clauses in Table 2.1 to the constants: {Gilles(G), Chris(C)} for a_1 and a_2; $\{C_1, C_2\}$ for r_1, r_2, and r_3. The factor graph involves: 12 ground atoms in which 4 are evidence (dark ovals) and 8 are non-evidence (non-dark ovals); 24 ground clauses wherein 8 are hard ($\mathcal{F}^h = \{f_1, \cdots, f_8\}$) and 16 are soft ($\mathcal{F}^s = \{f_9, \cdots, f_{24}\}$).

For instance, the Ising models in Eq. (2.4b) can be represented as factor graphs, where variables $X_i \in \mathcal{X}$ are represented as variable nodes and potentials $\{\phi_i(X_i)\}$ and $\{\phi_{ij}(X_i, X_j)\}$ are represented as factor nodes. Also, Markov logic can be represented as a factor graph. For clarity, let us represent Markov logic for the explanatory example, in Table 2.1, as a factor graph after grounding it using a small set of typed constants (say, for example, $a_1, a_2 \in \{\text{Gilles(G), Chris(C)}\}$, 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 2.2, which is a bipartite graph $(\mathcal{X}, \mathcal{F})$, where $\mathcal{F} = \left\{\mathcal{F}^h, \mathcal{F}^s\right\}$. This factor graph has a variable node (oval) for each ground atom $X_j \in \mathcal{X}$ (here $\mathcal{X}$ includes the ground atoms: $\text{SBib}(C_1, C_1)$, $\text{SBib}(C_2, C_1)$, $\text{SBib}(C_2, C_2)$, $\text{SBib}(C_1, C_2)$, $\text{Ar}(C_1, G)$, $\text{Ar}(C_2, G)$, $\text{Ar}(C_1, C)$, $\text{Ar}(C_2, C)$, $\text{SAr}(C, C)$, $\text{SAr}(C, G)$, $\text{SAr}(G, C)$, and $\text{SAr}(G, G)$). 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 square) and each soft ground clause $f_i^s \in \mathcal{F}^s$ (non-bold 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:

$$P(X_1, \cdots, X_n) = \frac{1}{\lambda} \prod_{i=1}^{|\mathcal{F}^h|} f_i^h(\mathcal{X}_{f_i^h}) \cdot \prod_{i=1}^{|\mathcal{F}^s|} f_i^s(\mathcal{X}_{f_i^s}) \tag{2.6}$$

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.

2.3 Probabilistic Reasoning over Graphical Models

We now turn to the problem of inference in factor graphs, in which some of the variables are observed (ex. with evidence), and we wish to compute the posterior distributions of one or more subset of other unobserved variables, called query or non-evidence. Hence, the objective of the inference task can be classified into two categories:

- Computing the marginal probability of the query given others as evidence. This is known as a *Marginal inference problem.* For instance, given the factor graph represented in Figure 2.2, the marginal inference problem can be how to compute the marginal probability of the query atoms $\mathcal{X} = \{SameBib\}$ given some others as evidence $E = \{Author\}$:

$$P(\mathcal{X}) = \sum_{E} P(\mathcal{X}, E) \tag{2.7}$$

- Computing the maximum a-posteriori probability (MAP) assignment (or the most probable explanation (MPE) as a special case of MAP) for the query given others as evidence. This is known as a *MAP inference problem.* For example, given the factor graph represented in Figure 2.2, the MAP inference problem could be how to find the most probable truth assignment of $\mathcal{X} = \{SameBib\}$ that maximizes the sum of the weights of satisfied clauses, given evidence $E = \{Author\}$:

$$\mathcal{X}^{MAP} = \operatorname*{argmax}_{\mathcal{X}} P(\mathcal{X}|E) \tag{2.8}$$

In the following subsections, we explain briefly some state-of-the art methods for solving each of these inference categories.

2.3.1 Message-Passing Methods

Loopy Belief Propagation

One widely used approximate inference is loopy belief propagation (LBP) (Yedidia *et al.*, 2005). LBP is a message-passing method that provides exact marginals of query atoms conditional on evidence atoms 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 (as shown in Figure 2.3).

The message from a variable X_j to a factor f_i is:

$$\mu_{X_j \to f_i} = \prod_{f_k \in \mathcal{F}_{X_j} \setminus \{f_i\}} \mu_{f_k \to X_j} \tag{2.9}$$

The message from a factor f_i to variable X_j is:

$$\mu_{f_i \to X_j} = \sum_{X_1} .. \sum_{X_{j-1}} \sum_{X_{j+1}} .. \sum_{X_l} \left(f_i(X_1, .., X_{j-1}, X_j, X_{j+1}, ..X_l) \prod_{X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}} \mu_{X_k \to f_i} \right) \tag{2.10}$$

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*

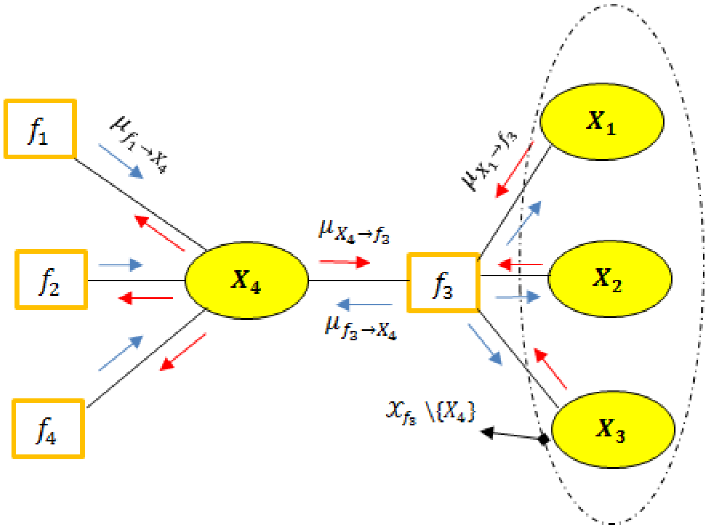

Figure 2.3 A depiction of the LBP's message-passing process on a simple factor graph consists of four variables $\{X_1, \ldots, X_4\}$ and four factors $\{f_1, \ldots, f_4\}$. It shows how LBP passes two types of messages: from variables to factors (in red) and from factors to variables (in blue).

messages[4]:

$$\beta_{X_j} \propto \prod_{f_i \in \mathcal{F}_{X_j}} \mu_{f_i \to X_j} \tag{2.11}$$

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 (SBib(C_1, C_1), SBib(C_2, C_1), SBib(C_1, C_2), and SBib(C_2, C_2)) on the factor graph represented in Figure 2.2. Remarkably, these query atoms are involved in many cycles. This emphasizes, at least theoretically, the existence of more than one local minimum (or fixed point) 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 evidence (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). These two limitations of LBP can be addressed by using our GEM-MP inference approach (as will be explained in Chapter 4).

[4] i.e. 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.

2.3.2 Markov Chain Monte Carlo Methods

Gibbs sampling

Gibbs sampling (Gibbs) (Geman and Geman, 1984) is a commonly used Markov Chain Monte Carlo (MCMC) algorithm for computing marginals. The applicability of Gibbs relies on the ease with which samples can be drawn from the conditional distributions. Each conditional distribution is a function defined over states of the nodes in the Markov blanket. In MRFs, the Markov blanket compromises the set of neighboring nodes. Starting from a random initial state for the Markov chain, the Gibbs procedure proceeds by applying a set of sampling steps for computing marginals until convergence or exceeding the maximum number of samples. At each step, Gibbs replaces the value of one of the variables by a value drawn from the distribution of that variable conditioned on the values of the other remaining variables. We repeat this step either by cycling through the variables in a specific order or by selecting them at each step at random from some distribution.

Consequently, the basic idea of Gibbs is that after creating a series of samples according to the decomposed joint probability distribution, the proportion of such samples containing a specific variable X_j assigned a value can be used to estimate the variable's marginal probability. Note that as the series length approaches infinity, the central limit theorem can guarantee that our estimate approaches the true marginal distribution. However, it is known that MCMC techniques, including Gibbs, require an irreducible sample space guaranteeing "ergodicity", which is non-trivial (Andrieu *et al.*, 2003) and could be limited in the presence of determinism (Poon and Domingos, 2006). In addition, MCMC methods essentially can take random walks biased toward the most likely portions of the search space over samples, and this, generally speaking, makes them very similar to local search algorithms when solving inference problems.

MC-SAT

MC-SAT (Poon and Domingos, 2006) is designed to overcome the limitation of irreducible sampling in MCMC due to determinism. By adding an auxiliary variable u_k for each factor f_k, MC-SAT applies slice sampling to Markov logic by using SampleSAT (Wei *et al.*, 2004) to sample a new state given the auxiliary variables.

Algorithm 1 gives the pseudo-code for MC-SAT. The algorithm starts from an initial state that is found by applying a satisfiability solver to the set of all hard clauses in the network (if this set is unsatisfiable, the output of MC-SAT is undefined). Then it applies a set of iterative sampling steps until convergence or exceeding the maximum number of samples. At each iteration, if f_k is not satisfied by the current state, u_k is drawn uniformly from $[0, 1]$, and there is no requirement that it will be satisfied in the next state. If f_k is satisfied, u_k is drawn uniformly from $[0, e^{w_k}]$, and with probability $1 - e^{-w_k}$ it will be greater than 1, in which case the next state must satisfy f_k. Thus, sampling all the auxiliary variables determines a random subset (say M) of the currently satisfied clauses that must also be satisfied in the next state. We then take as the next state a uniform sample from the set of states that satisfy M. Notice that the set (M) is never empty, because it always contains at least the current state.

Algorithm 1 MC-SAT

```
$\mathcal{X}^{(0)} \leftarrow$ Satisfy (Hard clauses);
**for** $i \leftarrow 1$ to $num\_samples$ **do**
    $M \leftarrow \emptyset$;
    **for all** $f_k \in$ hard clauses satisfied by $\mathcal{X}^{(i-1)}$ **do**
        with probability $1 - e^{-w_k}$ add $f_k$ to $M$;
    **end for**
    Sample $\mathcal{X}^{(i-1)} \sim \mathcal{U}_{\mathrm{SAT(M)}}$; //call SampleSAT
**end for**
```

MC-SAT provides accurate marginals in the presence of determinism compared to Gibbs. However, MC-SAT relies on SampleSAT to obtain nearly uniform samples, and it is known that SampleSAT may fail to find a satisfying solution (Wei *et al.*, 2004), when in fact there is always one. Thus, in practice, MC-SAT is still not guaranteed to be sound in the presence of determinism.

2.3.3 Local Search Methods

A major class of methods that have been applied to MAP inference are methods that explore the space of assignments, searching for a high-weight (or low-cost) assignment. In this context, local search became the predominant method for solving MAP inference.

Furthermore, Park (2002) showed that MAP inference corresponds to an instance of solving the weighted maximum satisfiability problem. Hence, taking into account the logical structures of PGMs such as those instantiated as MLNs, MAP inference can be carried out efficiently using a weighted satisfiability solver like MaxWalkSAT (Kautz *et al.*, 1997), which is currently the state-of-the-art MAP inference algorithm for MLNs (Sarkhel *et al.*, 2014).

MaxWalkSAT

The MaxWalkSAT (Kautz *et al.*, 1997) algorithm extends WalkSAT (Selman *et al.*, 1993) to the weighted satisfiability problem, where each clause has a weight and the goal is to maximize the sum of the weights of satisfied clauses. (Systematic solvers have also been extended to weighted satisfiability, but tend to work less well.)

Algorithm 2 gives a pseudo-code for MaxWalkSAT. The algorithm starts from a random initial state, then it repeatedly performs a stochastic local search by picking an unsatisfied clause at random and flipping the truth value of one of the atoms in it. With a certain probability, the atom is chosen randomly; otherwise, the atom is chosen to maximize the sum of satisfied clauses' weights when flipped. MaxWalkSAT can solve problems with hundreds of thousands of ground atoms in a fraction of a second, and hard ones in minutes. However, it cannot distinguish between an unsatisfiable CNF and one that takes too long to solve.

Algorithm 2 MaxWalkSAT.

Input: weighted clauses, max flips, max tries, target, p
Output: MAP solution $\mathcal{X}^{MAP}$

```
vars ← variables in weighted clauses
for $i \leftarrow 1$ to max tries do
    $\mathcal{X} \leftarrow$ a random truth assignment to vars
    cost ← sum of weights of unsatisfied clauses in $\mathcal{X}$
    for $i \leftarrow 1$ to max flips do
        if cost ≤ target then
            Return $\mathcal{X}^{MAP}$
        end if
        $f \leftarrow$ a randomly chosen unsatisfied clause
        if $\mathcal{U}(0,1) < p$ then
            $X \leftarrow$ a randomly chosen variable from $f$
        else
            for each variable $X$ in $f$ do
                compute DeltaCost($X$)
            end for
            $X^{MAP} \leftarrow X$ with lowest DeltaCost($X$)
        end if
        $\mathcal{X} \leftarrow \mathcal{X}$ with $X^{MAP}$ flipped
        cost ← cost + DeltaCost($X$)
    end for
end for
```

2.4 Constraint Satisfaction Techniques for Analyzing Constraint Problems

Starting from describing the constraint satisfaction problem (CSP) and Boolean satisfiability problem (SAT), our goal here is to state the conjecture of the clustering phenomenon — which is also known as phase transition — and illustrate its effects on the solution space of SAT problems within the transition area. Taking this as a motivation, we will next explain survey propagation as a successful application for SAT in the presence of the clustering phenomenon.

2.4.1 Constraint Satisfaction Problems

Definition 8. *(Rossi et al., 2006, Chapter 2)[Constraint Satisfaction Problem (CSP)] A CSP is a triple* $\left\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \right\rangle$ *where* $\mathcal{X}$ *is an n-tuple of* variables $\mathcal{X} = \left\langle X_1, \ldots, X_n \right\rangle$, $\mathcal{D}$ *is a corresponding n-tuple of* domains $\mathcal{D} = \left\langle D_1, \ldots, D_n \right\rangle$ *such that* $X_j \in D_j$*, and* $\mathcal{C}$ *is a m-tuple of* constraints $\mathcal{C} = \left\langle c_1, \ldots, c_m \right\rangle$*. A constraint* c_i *is a pair* $\left\langle \mathcal{R}_{\mathcal{X}_{c_i}}, \mathcal{X}_{c_i} \right\rangle$ *where* $\mathcal{R}_{\mathcal{X}_{c_i}}$ *is a relation on the variables* $\mathcal{X}_{c_i} = scope(c_i)$*. A solution to the CSP is a complete assignment (or a possible world)* $s = \left\langle v_1, \ldots, v_n \right\rangle$ *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 Programming (CP) is an appropriate paradigm to model and solve CSPs. In CP, solving a CSP requires a combination of constraint propagation (or constraint filtering) and search. Constraint propagation (see Rossi *et al.*, 2006, Chapter 3) 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 9 (Generalized arc Consistency (GAC))**.** *A constraint* $c \in \mathcal{C}$ *which is defined over the subset of variables* $\mathcal{X}_c$*, is* generalized arc consistent *(GAC) if and only if 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 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 2.2, applying GAC to variable

(SBib$(C_1, C_1) = true$) with a hard constraint (or clause) $f_6 : \neg\text{SBib}(C_1, C_1) \vee \neg\text{SBib}(C_1, C_2)$ 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.

Now, as aforementioned, in SRL models, we convert formulas to a typical clausal form (CNF). This indeed makes the Boolean satisfiability (SAT) problem of interest to our work in this thesis. SAT problems can be thought of as a certain form of the CSP (i.e., a special case of constraint satisfaction).

Definition 10 (Boolean Satisfiability (SAT))**.** *A general Boolean satisfiability (SAT) problem is a constraint satisfaction problem where all variables have the Boolean domain $\mathcal{D} = \{-, +\}$. A SAT problem in conjunctive normal form (CNF) represents a conjunction of clauses, with each clause representing a disjunction of literals, and a literal represents either a variable or its negation. A SAT can also be called a "theory", as suggested by its interpretation as a set of rules constraining any model of some system (Hopcroft et al., 2006). Given a CNF formula, the SAT is satisfiable if we determine a solution (i.e., a truth assignment that satisfy the CNF formula).*

Further, an important property of any problem in CNF is its density. In the world of satisfiability, the density is determined by computing the clause-to-variable ratio. It is commonly used to measure, intuitively, the constrainedness of a problem. As it will be explained in the next subsection, when the density is high there are probably few solutions that are apart from each other, and when it is low the solutions are likely numerous and easy to find.

2.4.2 Clustering Phenomenon and Geometry of the Solution Space

For clarity, we will explain the clustering phenomenon for random SAT problems. The clustering phenomenon of these problems has been widely explored in the literature (Semerjian and Monasson, 2003; Biroli *et al.*, 2002; Altarelli *et al.*, 2009). Let us use $\Gamma = m/n$ to denote the density (or clause-to-variable ratio) of a SAT instance, where m and n are the number of clauses and variables defining the problem, respectively.

Hence, consider the likelihood that a random SAT instance is satisfiable, given varying values of Γ. If the clause-to-variable ratio is extremely low, then the problem is under-constrained and very likely to be satisfiable; further, it is presumable that a solver of almost any design will be able to find a solution easily. Likewise, if Γ is extremely high then we can presume not only that the problem is likely to be unsatisfiable, but also that solvers will be able to make this determination efficiently. Specifically, as Γ increases so does the probability that the problem is unsatisfiable–but, the transition is not gradual. Rather, much as water abruptly becomes ice at a certain temperature point, random SAT instances undergo a phase transition in satisfiability across values of Γ. It was observed that there is a jump in the probability of producing an unsatisfiable problem, around the neighborhood of $\Gamma = 4.267$ for random k-SAT problems (Friedgut, 2005) – a k-SAT problem restricts SAT such that each clause disjoins exactly k literals. A standing conjecture in theoretical computer science and statistical physics is that for a given k, this phase transition occurs at a fixed clause-to-variable ratio Γ, for sufficiently large n.

The clustering phenomenon has attracted interest as an explanation for the performance limits of various SAT-solving approaches. The explanation is in terms of the geometry of solution space, where we consider the set of solutions to a satisfiable problem in terms of the Hamming distances between the complete assignments.

As depicted in Figure 2.4, we can see various thresholds occurring just below the unsatisfiability threshold Γ_u, corresponding to abrupt changes in the geometry of the solution set, along with limits on the applicability of various solving methodologies. For extremely low values of Γ, almost any complete assignment is a solution or near-solution — thus solutions are likely to cover the whole space of assignments and to be tightly interconnected, forming a single cluster. Accordingly, any conceivable and reasonable solving technique should find a solution relatively quickly. As Γ rises beyond a "submodularity" threshold Γ_{Sub}, the space of solutions narrows; but all solutions are still likely lie, for the most part, in a single large cluster of solutions and near-solutions. Thus, local search algorithms can continue to do well in this region because near-solutions are very likely to be near actual solutions. And as for marginal estimation, BP is still likely to do well — it is unlikely to build its estimates from

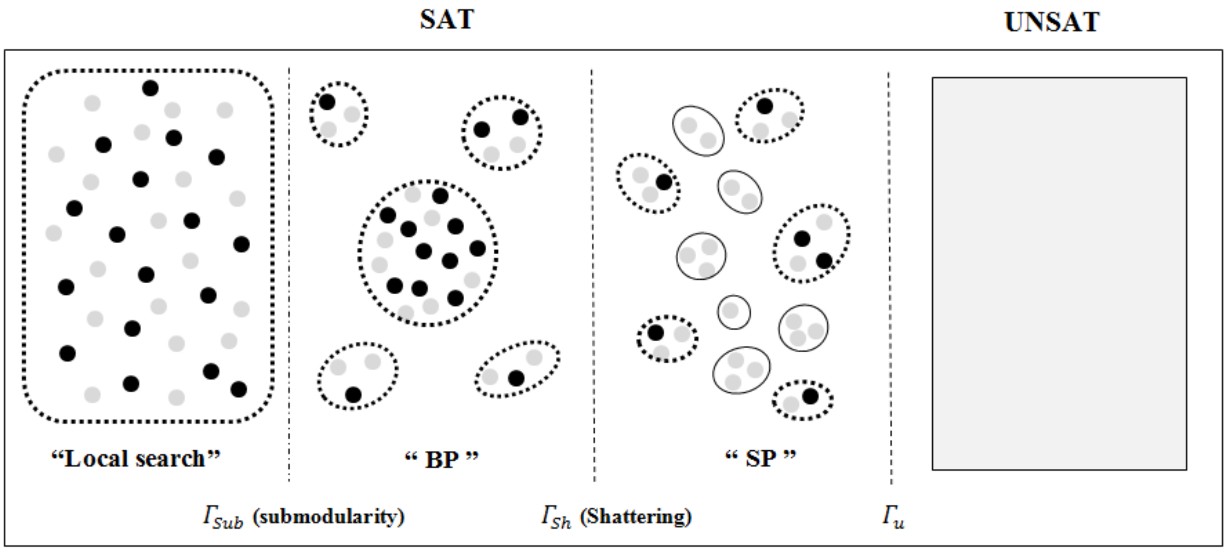

Figure 2.4 A notional depiction of the clustering phenomenon. It shows how the space between solutions varies as Γ increases. Solutions are depicted as solid circles, while unsatisfying assignment or near-solution (which satisfy almost all, but not all, of the clauses) appear as fainter circles. Within the limitations of a two-dimensional representation, the placement of assignments represents their Hamming distances. Thus, two assignments are considered adjacent if they have very small Hamming distance (e.g., they differ by a single variable). In addition, dotted outlines group adjacent assignments into arbitrary clusters of interest, while solid outlines group assignments into metastable clusters (Kilby *et al.*, 2005; Chavas *et al.*, 2005), which have no solutions. Finally, under each phase transition of the solution space, the solving technique that is widely used in the literature to find a solution relatively quickly, is indicated.

solution-impoverished regions of the space. The approximate location of the submodularity threshold for k-SAT is $\Gamma = 3.86$ (Braunstein *et al.*, 2005).

Importantly, as Γ continues to increase, but still before reaching unsatisfiability, the space of solutions has been observed to shatter into an exponentially large number of solution and near-solution clusters, where each such cluster is in turn exponentially small and where clusters that contain exclusively near-solutions exponentially outnumber those that contain solutions (Semerjian and Monasson, 2003; Chavas *et al.*, 2005; Mann and Hartmann, 2010). This impedes both local search and BP alike. In the first case, there are exponentially many local optima to entrap the search process, and in the latter, marginals will include information about exponentially many near-solution clusters, instead of solution clusters.

At this time, there is no known way to calculate Γ for arbitrary CSP instances, or to even express the threshold in general. However, there exists a proof for a strong version of the general conjecture that proves the existence of a clustering phenomenon (Mann and Hartmann, 2010; Friedgut *et al.*, 1999; Hartmann and Weigt, 2006).

The survey propagation (SP) model described in the next section was designed to extend the feasibility of marginal estimation into the shattering region, and has done so with good success (Braunstein *et al.*, 2005). The main idea of SP is to compute marginals over generalized assignments that represent solution clusters. In this context, Maneva *et al.* (2007) proposed the concept of *core* as a combinatorial representative of valid complete assignments of a cluster.

Definition 11. *(Maneva et al., 2007) A core is a valid complete assignment* $X \in \{0, 1, *\}^n$ *that satisfies all the clauses, and contains no unconstrained variables equal to* 0 *or* 1*.*

When a variable takes a fixed value 0 or 1 in the core X, then it is said to be *frozen* with respect to the core X (i.e., fixed in all the valid assignments of the cluster represented by core X) (Achlioptas and Ricci-Tersenghi, 2009). Otherwise it is *unfrozen.* A frozen variable is also called a *backbone* variable (Kilby *et al.*, 2005), and it can be extended for optimization problems (Slaney and Walsh, 2001).

Definition 12. *(Achlioptas and Ricci-Tersenghi, 2009) Cluster backbones refer to the set of frozen variables in a core* X*. In other words, it is the set of backbone variables in all valid complete assignments (i.e., solutions) within the cluster represented by* X *(Kroc et al., 2008).*

While a number of incomplete accounts attempt to explain why SP continues to work well beyond the shattering threshold (Battaglia *et al.*, 2004; Maneva *et al.*, 2007; Chieu and Lee, 2009), perhaps the most compelling explanation is that solution clusters in this region tend to have many frozen variables (or cluster backbones), and the more sophisticated survey propagation model is more sensitive to this fact.

2.4.3 The Survey Propagation Model of Satisfiability

From the message-passing perspective, SP (Maneva *et al.*, 2007) can be viewed as a LBP algorithm on a factor graph that defines non-zero probabilities over cores (Maneva *et al.*, 2007) representing solution clusters. Unlike standard BP, SP passes the messages as follows:

- Each factor $f_i \in \mathcal{F}$ passes a survey message containing a real number $\eta_{f_i \to X_j}$ to each of its neighboring variables X_j in the factor graph. This survey is the probability that the factor f_i warns the variable X_j against violating it. That is, if the survey message is close to 1, then the factor f_i is warning the variable X_j against taking a value that will violate the factor; if the survey is close to 0, then the factor f_i does not care about the value taken by X_j since it is satisfied by other variables in $\mathcal{X}_{f_i} \setminus \{X_j\}$.

- Each variable X_j sends to a neighboring factor f_i, a message $[\mu^s_{X_j \to f_i}, \mu^u_{X_j \to f_i}, \mu^*_{X_j \to f_i}]$, where the three components of the message correspond to the probability that X_j will be warned by other factors to take a value that will satisfy f_i, violate f_i, or freely take any value (i.e., joker).

2.4.4 Decimation Based on Survey Propagation

As was shown above in section 2.4.3, the variables' marginals obtained from SP correspond to surveys over the clusters in the solution space. These marginals in fact provide information about the fraction of clusters in which each variable is free or frozen (Chieu *et al.*, 2007; Braunstein *et al.*, 2005; Battaglia *et al.*, 2004). Thus one efficient way to exploit the SP's marginals (also called biases) is to apply a marginalization-decimation algorithm based on SP (Braunstein *et al.*, 2005; Kroc *et al.*, 2009), which can be summarized in the following steps:

1. Run SP on the underlying model.
2. Extract the fraction of frozen variables with the largest biased marginals, and fix them to their most likely values.
3. Simplify the model, and return to Step 1.
4. Once we obtain all the biased variables based on a pre-specified threshold, the Walk-SAT algorithm is applied to find the remainder of the assignment (if there are still some variables not assigned).

In this process, the results of the initial phases of the decimation are partially assigned to variables, which determine a particular solution cluster. So the goal of the initial phases is really to use SP to find a cluster. Once the cluster is found, the problem is relatively easy to solve. Thus an assignment to the rest of the variables can be found using any conventional algorithm that works well within a given cluster such as the Walk-SAT algorithm (Kautz *et al.*, 1997; Selman *et al.*, 1993). This mechanism has shown to be one of the best incomplete solvers in solving hard k-SAT instances efficiently (Chieu *et al.*, 2007; Braunstein *et al.*, 2005; Battaglia *et al.*, 2004; Kroc *et al.*, 2009).

2.5 Variational Approximation Methods

To derive a method with enhanced algorithmic behavior and theoretical semantics for LBP, we shall be interested in both variational EM (Beal and Ghahramani, 2003; Neal and Hinton, 1999) and variational mean-field (Saul *et al.*, 1996; Koller and Friedman, 2009). Thus, let us review these techniques in a setting which is suitable for the purpose of our GEM-MP presented in chapter 4.

2.5.1 Variational Expectation Maximization

Suppose that we have a model structure $\mathcal{M}$ with parameters θ, observed data $\mathcal{O} = \{O_1, \ldots, O_n\}$ and hidden variables $\mathcal{H} = \{H_1, \ldots, H_n\}$. If we assume a prior distribution over parameters $P(\theta|\mathcal{M})$ conditional on $\mathcal{M}$ then the model defines a log marginal likelihood of the form $P(\mathcal{H}|\mathcal{M})$. By introducing distributions over both hidden variables, $q_{\mathcal{H}}(\mathcal{H})$ and parameters, $q_\theta(\theta)$, we can leverage Jensen's inequality to obtain a lower bound $\mathcal{F}_{\mathcal{M}}$ on the log marginal likelihood with the following form:

$$\mathcal{F}_{\mathcal{M}} = \sum_{\theta} q_\theta(\theta) q_{\mathcal{H}}(\mathcal{H}) \log \frac{P(\mathcal{O}, \mathcal{H}, \theta|\mathcal{M})}{q_\theta(\theta) q_{\mathcal{H}}(\mathcal{H})} \tag{2.12}$$

The role of the variational Expectation Maximization (EM) is that it serves as an iterative optimizing algorithm for the lower bound $\mathcal{F}_{\mathcal{M}}$ in Eq. (2.12) (i.e., estimating the parameters from the observed data in the presence of hidden variables) by performing the following updates:

VBE-step:

$$q_{\mathcal{H}_i}^{(t+1)}(\mathcal{H}_i) = \mathcal{Z}_{H_i}^{-1} \exp\left[\sum_{\theta} q_\theta^{(t)}(\theta) \log P(\mathcal{O}_i, \mathcal{H}_i, \theta|\mathcal{M})\right] \tag{2.13}$$

where $q_{\mathcal{H}}^{(t+1)}(\mathcal{H})$ is approximated as:

$$q_{\mathcal{H}}^{(t+1)}(\mathcal{H}) = \prod_i q_{\mathcal{H}_i}^{(t+1)}(\mathcal{H}_i) \tag{2.14}$$

VBM-step:

$$q_\theta^{(t+1)}(\theta) = \mathcal{Z}_\theta^{-1} P(\theta|\mathcal{M}) \exp\left[\sum_{\mathcal{H}} q_{\mathcal{H}}^{(t+1)}(\mathcal{H}) \log P(\mathcal{O}, \mathcal{H}|\theta, \mathcal{M})\right] \tag{2.15}$$

The derivations and proofs for Eqs. (2.12), (2.13), (2.14), and (2.15) are given in more detail in (Beal and Ghahramani, 2003). Now, at a high level one can see variational EM as two iterative steps for maximizing the lower bound $\mathcal{F}_{\mathcal{M}}$: the VBE-step uses information about parameters to maximize the expectation of the log marginal likelihood with respect to $q_{\mathcal{H}}^{(t+1)}(\mathcal{H})$, and the VBM-step takes advantage of the new obtained information about hidden variables to maximize the expectation of the log marginal likelihood with respect to $q_{\theta}^{(t+1)}(\theta)$. In addition, one can simply reduce variational EM to the ordinary EM by restricting the parameter estimation in the M-step to be a point estimation that involves re-estimating θ (see Beal and Ghahramani, 2003, for more detail).

2.5.2 Variational Mean Field approximation

Variational Mean Field (MF) (Koller and Friedman, 2009) is a variational algorithm that approximates an intractable distribution $q(\mathcal{X})$ with a fully factorized distribution as follows:

$$q(\mathcal{X}) = \prod_{X_j \in \mathcal{X}} q(X_j) \tag{2.16}$$

where the $q(\mathcal{X})$ distribution is selected to minimize the distance to $P(\mathcal{X})$, which is the true (or exact) joint probability. The distance is measured as the Kullback-Leibler divergence $KL(q(\mathcal{X})||P(\mathcal{X}))$. To reach a local minimum, MF updates the marginal distributions $\{q(X_j)\}_{X_j \in \mathcal{X}}$ as follows (see Koller and Friedman (2009), for more details):

$$q(X_j) = \frac{1}{\lambda_j} \exp\left(\sum_{f_i \in \mathcal{F}: X_j \in \mathcal{X}_f} E_{\sim q(X_j)}[\log q(\mathcal{X}_{f_i} \setminus \{X_j\}, X_j)] \right) \tag{2.17}$$

Where λ_j is normalizing constant and $E_{\sim q(X_j)}$ is the expectation of factor f in which X_j appears. According to MF's updating rule in Eq. (2.17), the marginal distribution $q(X_j)$ of each variable X_j has to be consistent with the expectation of the factors $f_i \in \mathcal{F}$ that involve X_j, and the resulting value of the marginal $q(X_j)$ is the optimal value given the choice of all other marginals of its consistent variables values in f_i. This intuitively means that MF updates the marginal distribution $q(X_j)$ using the set of marginals of other variables' values that are locally consistent with respect to the factors that involve X_j. At a high level, this could be seen as a kind of application of local consistency enforcing (such as GAC) to obtain those consistent variables' values required for updating the marginal of X_j. In chapter 4, we take advantage of this relationship between MF and GAC to develop a new probabilistic GAC, denoted as (pGAC).

CHAPTER 3 LITERATURE REVIEW

In this chapter, we discuss relevant work to our research in this thesis. Three main topics were selected for the literature review. First, in Section 3.1, we examine various significant inference methodologies for computing marginals using message-passing. Second, in Section 3.2, we discuss previous important work that integrates constraint satisfaction techniques with message-passing to improve the accuracy and scalability of marginal inference. Finally, in Section 3.3, we survey some worthy work related to improving MAP inference.

3.1 Message-Passing Techniques for Computing Marginals

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, several works (such as Mceliece *et al.*, 1998; Frey and MacKay, 1998) have 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 to understand 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). All such formulations were inspired 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 demonstrated that LBP is related to variational approximations (Yedidia *et al.*, 2005). This pioneering work outlined new research directions for a deeper understanding of and improvements to LBP.

3.1.1 Studying Message-passing's convergence

The first area of research is the investigation of the convergence of LBP by studying the sufficient conditions to ensure the existence and the uniqueness of its fixed point. Early on, Heskes (2004) pointed out that if a graph involves a single cycle, then we have a unique fixed point, 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. 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"

Drawing on this observation, we put forth our first research hypothesis which is: updating the marginals such that we do not cross the nearest local minimum (or fixed point) could alleviate the threat of non-convergence of LBP.

Another mainstream work attempts to derive new types of LBP for approximating 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 the convergence. Recently, alternative algorithms have been introduced to guarantee the 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 shares with the previously mentioned approaches the derivation of new variational inference that minimizes a free-energy functional. But GEM-MP has different characteristics that enable it to guarantee convergence by minimizing a KL divergence between an approximation to the true posterior distribution of the model and an approximation at each step when updating marginals such that it never overshoots the local minimum.

3.1.2 Damped Message-passing

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.

A key of the approach of GEM-MP is to avoid such risks by compelling LBP to never overshoot the nearest local minimum in the marginal space.

3.1.3 Re-parameterized Message-passing

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.

3.1.4 Message passing 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. Later on, 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 the structure 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.

3.1.5 Scalable Message Passing

Another promising research area that has been recently explored seeks to improve the scalability of inference on large probabilistic 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 Message-passing 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 by 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 GEM-MP approach is different from the above lifted-based message passing algorithms being built on a propositional basis for improving the inference's accuracy, but it can be easily incorporated with their benefits for lifting its inference. In addition, our PR scaling strategy (Ibrahim et al., 2015) is also different from those aforementioned algorithms because it leverages determinism (i.e., hard constraints) presented in the structure of the network to reduce its size, but it also can be used in conjunction with other prominent methods for scaling inference (as it will be shown in Chapter 6, where we use PR with lazy inference).

3.2 Integrating Constraint Satisfaction techniques with Message Passing

It remains unclear if there is a relationship between determinism and the uniqueness of fixed points of LBP. However, it is observable that applying LBP on graphical models with determinism and cycles is more likely to oscillate or converge to wrong results.

3.2.1 Constraint Propagation Based Methods

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, such a procedure has a time and space complexity that is exponential in the tree-width of the graph, which makes it inapplicable to large domains. 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.

3.2.2 Survey Propagation Based Methods

In a complementary context, another fruitful area of research is how to relate Boolean satisfiability and CSP methods to local search for marginal inference, most notably survey propagation (Braunstein *et al.*, 2005). The SP algorithm has shown a surprising ability to accurately solve extremely large and hard SAT instances close to the satisfiability threshold (Braunstein *et al.*, 2005; Kroc *et al.*, 2009). In an attempt to more fully understand the success of SP, it was first shown that SP can be viewed as belief propagation (BP) on a factor graph defined over solution clusters represented by covers having positive probability (Braunstein and Zecchina, 2004). Subsequently, SP has been generalized (Maneva *et al.*, 2007) to work on extended MRFs, where solution clusters can be elegantly represented as cores having positive probability. It has also been extended to an algorithm called SP-y (Battaglia *et al.*, 2004) to handle the maximum satisfiability (Max-SAT) problem. Recently, relaxed SP (Chieu *et al.*, 2007) has been introduced as an extension to the SP and SP-y for weighted satisfiability. However, the applicability of these approaches to complex relational problems is still limited.

Our WSP-χ approach can be seen as akin to these SP-based inference methods. But WSP-χ is different in being built on different re-parameterized extended factor graphs that are rooted in relational techniques and used to address relational MAP inference.

3.3 Solving Maximum-A-Posteriori Inference Problems

Developing accurate and scalable MAP inference in probabilistic graphical models (PGMs) is a key challenge that would affect a wide range of applications in machine learning, constraint satisfaction, information theory, computational biology, and other sub-disciplines of artificial intelligence. The iterbi algorithm (Forney, 1973) was first proposed to address MAP inference — in hidden Markov models — for decoding applications. A generalization was then proposed to other types of PGMs in other applications (Pearl, 1988), and extended by the appearance of a clique tree algorithm (Lauritzen and Spiegelhalter, 1988). Since then, several pioneering research directions in AI have been explored for improving MAP inference.

3.3.1 Systematic and Non-Systematic Search Methods

The first research area uses the fact that MAP inference can be cast as the task of minimizing an energy function — in computer vision applications (Szeliski, 2006), the energy function defined over terms enforces some kind of spatial coherence; another one penalizes solutions that are inconsistent with the observed data. Under this line of analysis, hill-climbing methods such as iterated conditional modes (ICM) (Besag, 1986), or simulated annealing (SA) (Granville *et al.*, 1994) become directly applicable to solve the MAP inference problem. However, in practice these algorithms were proven to be inefficient. For example, in computer vision applications (Szeliski, 2006), this is due to the slowdown that comes from the considerable amount of computation in their early adaptations. Other systematic search methods such as branch-and-bound have also been applied to solve MAP inference (Marinescu and Dechter, 2005). However, it is known that the efficiency of these algorithms heavily relies on both the tightness of the bound used and the branching strategy's effectiveness; and unfortunately, there is no general bounding and branching strategy that performs well for all MAP problems.

Furthermore, when the PGM features logical structures like those instantiated as SRL models, finding the MAP solution of the model can be obtained by solving its weighted satisfiability problem using any satisfiability solver (Huynh and Mooney, 2009). This in fact opens the way to the use of local search algorithms for MAP inference. For instance, MaxWalkSAT (Selman *et al.*, 1993) was proposed to tackle the MAP problem in MLNs (Richardson and Domingos, 2006). However, local search algorithms are non-systematic, and they often come with no performance guarantees. Additionally, it was observed that if the density of the underlying graphical model was close to a certain threshold then the search space of MAP solutions would become clustered (Hartmann and Weigt, 2006; Zhang, 2004; Gomes *et al.*, 2002; Parkes, 1997; Kambhampati and Liu, 2013). However, none of those approaches took into consideration the clustering that occurred in the search space when solving the MAP inference problem, and this makes them vulnerable to getting stuck in a local maximum at one of the metastable clusters (Kilby *et al.*, 2005; Chavas *et al.*, 2005).

Our WSP-χ approach can serve as a pre-processing step to handle the clustered search space when solving the MAP problem. Thus it can be used in conjunction with all the above search algorithms increase the possibility of finding the optimal MAP solution.

3.3.2 Message-Passing-Based Methods

Another traditional research area involves the use of message-passing algorithms to tackle MAP inference. Here the mainstream interest is in max-product loopy BP. From a theoretical perspective, it was shown that the marginals obtained after the convergence of max-product BP can be used to derive a MAP solution that is globally optimum for an acyclic network, and locally optimum for a loopy network (Weiss and Freeman, 2001). Accordingly, parameterized message passing (Wainwright *et al.*, 2004) has been proposed to obtain beliefs corresponding to max-marginals. This work then empirically extended by developing a convexified message-passing algorithm for MAP inference called Tree-reweighted max-product message-passing (TRW) algorithm (Wainwright *et al.*, 2005). However, TRW does not monotonically increase its objective function and therefore does not converge in general to accurate results. As a remedy, a variant of TRW, called TRW-S (Kolmogorov, 2006) — which involved passing asynchronously — has been proposed to improve the objective monotonically. However, TRW-S still cannot guarantee convergence to a global optimum since it can get stuck in local optima. From the optimization perspective, TRW and its variant TRW-S can be in fact considered the first connection between LP relaxation and message passing algorithms. This stimulates some other works to extend TRW by using sophisticated LP relaxation methods. For instance, a general TRW-based approach (Ravikumar and Lafferty, 2006) has been proposed to iteratively improve the solution by searching the nearest improvements using a proximal optimization method. However, it was subsequently proven that the use of a simple LP relaxation (Kumar and Torr, 2008) would produce better (i.e., tighter) relaxation than the complicated one that is based on a proximal optimization method (Ravikumar and Lafferty, 2006). Other work has extended TRW to Tree-Reweighted Belief Propagation (TRBP) that uses the marginals derived from a convex-BP to obtain optimal MAP solutions (Yanover *et al.*, 2006).

At a high level, our WSP-χ approach shares with the previously mentioned approaches the derivation of a parameterized BP message passing procedure such that the obtained beliefs correspond to marginals. However, WSP-χ is different, since its marginals are defined over max-cores, which are representations of clusters of candidate MAP solutions. In addition, in WSP-χ, there is a cooling parameter y that plays a similar role to the temperature parameter in simulated annealing (Granville et al., 1994). This cooling parameter can be tuned to reduce the strengths of factors. We show empirically how this greatly improves the convergence of WSP-χ on the associated extended factor graph. This gives WSP-χ different characteristics that enable it to avoid getting stuck in local optima and be more likely to convergence to a global optimum.

3.3.3 Scalable MAP Methods

In the context of scaling MAP inference, FOVE-P (De Salvo Braz *et al.*, 2006) was proposed as the first lifted variable elimination, which is a modification of FOVE (De Salvo Braz *et al.*, 2005) that incorporates partial inversion for lifting MAP inference in SRL models. Recently, a lifted MaxWalkSAT algorithm (Sarkhel and Gogate, 2013) has been proposed for MAP inference in Markov logic networks (MLNs). Subsequently, a generalization of lifted MaxWalkSAT (Sarkhel *et al.*, 2014) has been introduced for lifting MAP inference in MLNs. However, we could repeal the merit of lifting inference if symmetries break when the variables become correlated by the virtue of depending asymmetrically on evidence (Kersting, 2012), since in this case lifted inference becomes close to propositionalized inference. Lazy Inference is another efficient way to scale MAP inference. Lazy MaxWalkSAT (Singla and Domingos, 2006b) was proposed as the first lazy algorithm for MAP inference in Markov logic networks, where the sparseness was used to ground the theory lazily for yielding gains in memory and time. Other work proposed a Cutting Plane Inference (Riedel, 2008) as an efficient and accurate algorithm for MAP inference. Although CPI avoids grounding the whole MLN, it works well for some types of MLNs where the separation step of the CPI method returns a small set of constraints.

Our WSP-χ approach is different from the above scaling algorithms, since it is based on the use of backbones (or frozen variables) to fix complex parts of the network by enlarging the evidence database and reducing the set of queries, and thereby simplifying the network. But it is also orthogonal to their benefits, and thus can be combined with them (as it will be shown in Section 6.3, where we combine WSP-χ with lazy inference).

CHAPTER 4 IMPROVING INFERENCE IN THE PRESENCE OF DETERMINISM AND CYCLES

In this chapter, our key objective 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, which are due to the presence of cycles and determinism as discussed in Chapter 1, we introduce **G**eneralized arc-consistency **E**xpectation-**M**aximization **M**essage-**P**assing (GEM-MP), a novel message-passing algorithm for applying variational approximate inference to graphical models.

We have organized this chapter in the following manner. In Section 4.1, we demonstrate the framework of GEM-MP variational inference. We then derive GEM-MP's general update rule for Markov logic in Section 4.2. In section 4.5, we generalize GEM-MP's update rules to be applicable for MRFs. In Section 4.3, we discuss the distinctions between GEM-MP and standard LBP. In Section 4.6, we conduct a thorough experimental study. This is followed by a discussion in Section 4.7.

4.1 GEM-MP Framework

Both LBP and variational EM approaches share a similar objective which is to minimize a corresponding energy equation (Yedidia *et al.*, 2005), the Gibbs free energy and variational free energy, respectively. Thus, one can hybridize the two approaches for both learning and inference tasks. A core aspect of GEM-MP is that if we re-express the inference task of estimating marginals as an instance of the learning task of fitting approximate (or variational) model parameters, then variational EM can be adapted to guarantee convergence. Intuitively, we treat the extraction of marginals from a factor graph (which could be computed by LBP message passing inference) as a maximum marginal-likelihood parameter estimation task (that can be accomplished by variational EM). This implies transforming marginals into parameters in a marginal-likelihood space, where the local optima (of variational parameters) are the local minima (of marginals). In this space, we can guarantee that variational EM never overshoots the nearest local optimum (i.e., fixed point).

Now, before presenting GEM-MP in detail, let us consider a simple example factor graph $\mathcal{G}$ (Figure 4.1(*left*)), which is a fragment of the Cora example in Figure 2.2, 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 $\{\mathrm{SBib}(C_2, C_2), \mathrm{SBib}(C_2, C_1), \mathrm{SBib}(C_1, C_2)\}$ respectively.

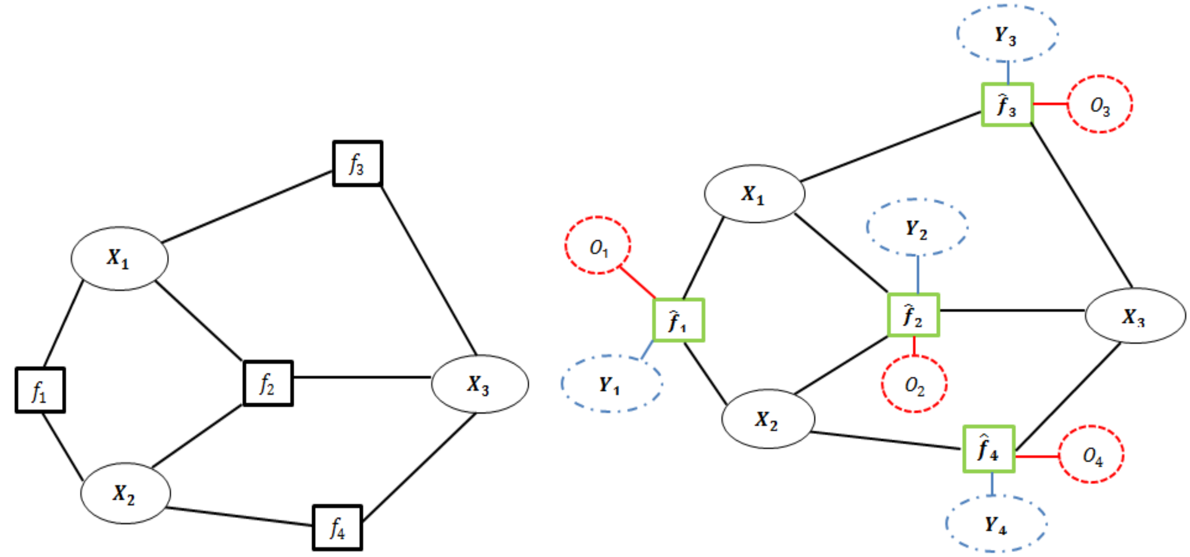

Figure 4.1 An example factor graph $\mathcal{G}$ (*left*) which is a fragment of the Cora example in Figure 2.2, that involves factors $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$ and three random variables $\{X_1, X_2, X_3\}$ representing query ground atoms $\{\text{SBib}(C_2, C_2), \text{SBib}(C_2, C_1), \text{SBib}(C_1, C_2)\}$. 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 $\hat{\mathcal{F}} = \left\{\hat{f}_1, \hat{f}_2, \hat{f}_3, \hat{f}_4\right\}$.

In our GEM-MP framework the first thing we do is 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 by transforming it into an *extended factor graph.* This extended factor graph $\hat{\mathcal{G}}$ (depicted in Figure 4.1(*right*)) extends the original factor graph $\mathcal{G}$ 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 valid 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:

$$\mathbb{1}_{(Y_i,f_i)} = \begin{cases} 1 & \text{If the state of } Y_i \text{ is identical to local entry of } f_i. \\ 0 & \text{Otherwise} \end{cases} \tag{4.1}$$

- 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. (4.1). 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:

$$\hat{f}_i(X_1 = x_1, \ldots, X_n = x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} = f_i(X_1 = x_1, \ldots, X_n = x_n) \tag{4.2}$$

But if the indicator constraint in Eq. (4.1) 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:

$$\hat{f}_i(X_1 = x_1, \ldots, X_n = x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = 0 \tag{4.3}$$

- When the activation node O_i is not equal to one, then it deactivates the indicator constraint in Eq. (4.1). Here, the extended factor 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 this assignment implies that the deactivation of the indicator constraint results in no impact on the distribution from the inclusion of the factor f_i.

Table 4.1 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 $\mathcal{G}$.*[1]

$$\sum_{Y_i} \hat{f}_i(X_1, \ldots, X_n, Y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = f_i(X_1, \ldots, X_n),\ \forall \hat{f}_i \in \hat{\mathcal{F}} \tag{4.4}$$

Proof. see Appendix A □

Proposition 2. *Any arbitrary factor graph $\mathcal{G}$ is equivalent, i.e., defining 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 □

To that end, we can recast the process of marginals estimation of variables as a maximum marginal-likelihood parameters estimation task, which is motivated by the extended factor graph model $\mathcal{M}$ in the following manner:

- 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 $\bar{O}_i = 1,\ \forall O_i \in \mathcal{O}$ (meaning that the auxiliary activation nodes are activated).

[1] Note that we will use the bar notation over the variable O_i (i.e., $\bar{O}_i$) to indicate that the variable O_i is observed at one of its values (e.g., 1).

Table 4.1 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

- 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.

- Now, we can set up our goal to estimate the marginals of variables in $\mathcal{X} = \{X_j\}_{j=1}^{n}$ that maximize the log marginal-likelihood (i.e., model evidence), $\log P(\mathcal{O}|\mathcal{M})$, which can be simply represented as follows:

$$\log P(\mathcal{O}|\mathcal{M}) = \log \sum_{\mathcal{H}} P(\mathcal{O}, \mathcal{H}|\mathcal{M}) = \log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M}) \tag{4.5}$$

However, to maximize Eq. (4.5), we will face the challenge of marginalizing over hidden mega-node variables $\mathcal{Y}$, which is intractable because it requires expressing all valid local entries of the factors (i.e., ground clauses) involved in the model. Instead, we add an auxiliary distribution $q(\mathcal{X}, \mathcal{Y})$ over the set of hidden variables to the log marginal-likelihood as follows:

$$\log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M}) = \log \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X}, \mathcal{Y}) \frac{P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})}{q(\mathcal{X}, \mathcal{Y})} \tag{4.6}$$

Now, profiting from the fact that "log"[2] is a concave function, we can apply Jensen's inequality to get a lower bound of the log marginal-likelihood:

$$\log \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X}, \mathcal{Y}) \frac{P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})}{q(\mathcal{X}, \mathcal{Y})} \geq \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X}, \mathcal{Y}) \log \frac{P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})}{q(\mathcal{X}, \mathcal{Y})} \tag{4.7}$$

Note that maximizing the lower bound, in Eq. (4.7), with respect to the free auxiliary distribution, $q(\mathcal{X}, \mathcal{Y}) = P(\mathcal{X}, \mathcal{Y}|\mathcal{O}, \mathcal{M})$, turns the inequality into an equality, which complicates the problem since evaluating exactly the true posterior distribution $P(\mathcal{X}, \mathcal{Y}|\mathcal{O}, \mathcal{M})$ is intractable. Instead we constrain the auxiliary distribution to be a factorized (separable) approximation:

$$q(\mathcal{X}, \mathcal{Y}) = q(\mathcal{X}; \mathcal{B}_{\mathcal{X}})\; q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}}) \tag{4.8}$$

Where $q(\mathcal{X}; \mathcal{B}_{\mathcal{X}})$ is an approximation of the true posterior distribution $P(\mathcal{X}|\mathcal{O}, \mathcal{M})$ over hidden variables, $\mathcal{X}$. This distribution is characterized by a set of variational parameters, $\mathcal{B}_{\mathcal{X}} = \left\{\beta_{X_j}\right\}_{j=1}^{n}$, representing the marginal probabilities of variables $X_j \in \mathcal{X}$, which were supposed to be computed by the standard message-passing inference (e.g., LBP). The distribution $q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})$ is an approximation to the true posterior distribution $P(\mathcal{Y}|\mathcal{O}, \mathcal{M})$ over hidden

[2] "Log" plays an important role in maintaining convergences via Jensen's inequality, as will be explained further on.

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 $Y_i \in \mathcal{Y}$. These variational parameters, $\alpha_{Y_i}(f_i)$, can be generally defined as:

$$\alpha_{Y_i}(f_i) = \begin{cases} v_s & \text{if the state of } Y_i \text{ satisfies } f_i, \\ v_u & \text{otherwise.} \end{cases} \tag{4.9}$$

Where v_s (and v_u) are the values obtained from f_i when a particular state of Y_i satisfies (and dissatisfies) the factor f_i respectively. Note that v_s and v_u can be adapted using both the variational distributions of argument variables and weights associated with factor f_i (as it will be explained in Subsections 4.2.1 and 4.2.2 for hard and soft factors respectively).

Now from Eqs. (4.6), (4.7) and (4.8), we have that:

$$\log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M}) \geq \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X};\mathcal{B}_{\mathcal{X}})\; q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) \log \frac{P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})}{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})\; q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} \tag{4.10a}$$

$$= \sum_{\mathcal{X}} q(\mathcal{X};\mathcal{B}_{\mathcal{X}}) \times \left[\sum_{\mathcal{Y}} q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) \left(\log \frac{P(\mathcal{O},\mathcal{Y},|\mathcal{X},\mathcal{M})}{q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} + \log \frac{P(\mathcal{X}|\mathcal{M})}{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})} \right) \right] \tag{4.10b}$$

$$= \mathcal{F}_{\mathcal{M}}(q(\mathcal{X};\mathcal{B}_{\mathcal{X}}), q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})) \tag{4.10c}$$

The lower bound $\mathcal{F}_{\mathcal{M}}$, in Eq. (4.10c), is the negative of a quantity that represents the variational free energy functional of the free distributions $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ and $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$. We can write the lower bound $\mathcal{F}_{\mathcal{M}}$ as follows:

$$\mathcal{F}_{\mathcal{M}} = E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\Big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\Big] + H\Big(q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})\Big) \tag{4.11}$$

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 (see Neal and Hinton, 1999, for more details).

Now the goal 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

$$\text{- } \mathcal{T}_{\mathcal{Y}}^{*} \propto \underset{\mathcal{T}_{\mathcal{Y}}}{\operatorname{argmax}}\ E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\Big] + H\Big(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}}) q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})\Big) \qquad (4.12a)$$

$$\text{- } \mathcal{B}_{\mathcal{X}}^{*} \propto \underset{\mathcal{B}_{\mathcal{X}}}{\operatorname{argmax}}\ E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\Big] + H\Big(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}}) q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})\Big) \qquad (4.12b)$$

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

$$H\Big(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}}) q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})\Big) = H\Big(q(\mathcal{X}, \mathcal{Y})\Big) = H\Big(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}})\Big) + H\Big(q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})\Big) \qquad (4.13)$$

Thus when maximizing $\mathcal{F}_{\mathcal{M}}$, with respect to the variational parameters $\mathcal{T}_{\mathcal{Y}}$ and $B_{\mathcal{X}}$, since in Eqs. (4.12a) and (4.12b), $q(\mathcal{X}; \mathcal{B}_{\mathcal{X}})$ does not depend on the entropy $H(q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}}))$, and $q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})$ does not depend on $H(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}}))$. We thus have

$$\text{- } \mathcal{T}_{\mathcal{Y}}^{*} \propto \underset{\mathcal{T}_{\mathcal{Y}}}{\operatorname{argmax}}\ E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\Big] + H\Big(q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})\Big) \qquad (4.14a)$$

$$\text{- } \mathcal{B}_{\mathcal{X}}^{*} \propto \underset{\mathcal{B}_{\mathcal{X}}}{\operatorname{argmax}}\ E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\Big] + H\Big(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}})\Big). \qquad (4.14b)$$

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:

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

$$\overbrace{\mathcal{T}_{\mathcal{Y}}^{(t+1)}}^{\text{Max. w.r.t } q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}} = \underset{\mathcal{T}_{\mathcal{Y}}}{\operatorname{argmax}}\ \overbrace{E_{q^{(t)}_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q^{(t)}_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\Big]}^{\text{E-step}} + H\Big(q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})\Big) \qquad (4.15)$$

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

$$\overbrace{\mathcal{B}_{\mathcal{X}}^{(t+1)}}^{\text{Max. w.r.t } q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}} = \underset{\mathcal{B}_{\mathcal{X}}}{\operatorname{argmax}}\ \overbrace{E_{q^{(t)}_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O}, \mathcal{X}, \mathcal{Y}|\mathcal{M})\Big]}^{\text{E-step}} + H\Big(q(\mathcal{X}; \mathcal{B}_{\mathcal{X}})\Big) \qquad (4.16)$$

Where here the arguments to Eqs. (4.15) and (4.16) 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.

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:

$$q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) = \prod_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \tag{4.17}$$

where $q(Y_i;\alpha_{Y_i})$ is our complete approximation to the true posterior 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:

$$q(\mathcal{X};\mathcal{B}_{\mathcal{X}}) = \prod_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}), \tag{4.18}$$

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

From Eqs. (4.17) and (4.18), we can write the expected log marginal-likelihood in Eqs. (4.15) and (4.16), as follows:

$$\begin{aligned} &E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\Big] \\ &\qquad = \sum_{\mathcal{Y}} \prod_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \Bigg[\sum_{\mathcal{X}} \prod_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}) \log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\Bigg] \end{aligned} \tag{4.19}$$

We now proceed to optimize the lower bound, through the use of Eqs. (4.15) and (4.16), 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. (4.19), (4.18) and (4.17) in Eq. (4.15). Hence, we have a maximization of the lower bound on the log marginal-likelihood as

$$\begin{aligned} \mathcal{T}_{\mathcal{Y}}^{(\mathrm{t+1})} = \underset{\mathcal{T}_{\mathcal{Y}}}{\operatorname{argmax}} \sum_{\mathcal{Y}} \prod_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \Bigg[\sum_{\mathcal{X}} \prod_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}) \log P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})\Bigg] \\ + \sum_{Y_i \in \mathcal{Y}} H(q(Y_i;\alpha_{Y_i})) \end{aligned} \tag{4.20}$$

One can then separate out the terms related to the updates of the variational parameters for each mega-node Y_i in Eqs (4.20). 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

$$\alpha_{Y_i}^{(\mathrm{t+1})} = \underset{\alpha_{Y_i}}{\operatorname{argmax}} \ \sum_{Y_i} q(Y_i;\alpha_{Y_i}) \Big[\sum_{\mathcal{X}_{\hat{f}_i}} \prod_{X_j \in \mathcal{X}_{\hat{f}_i}} q(X_j;\beta_{X_j}) \log \hat{f}_i(O_i, \mathcal{X}_{\hat{f}_i}, Y_i|\mathcal{M}) \Big] \\ +H(q(Y_i;\alpha_{Y_i})) \tag{4.21}$$

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

$$q(Y_i;\alpha^*_{Y_i}) = \frac{1}{\mathcal{Z}_{Y_i}} \exp\left(\overbrace{E_{\{q^{(t)}(X_j;\beta_{X_j}) \,|\, X_j \in \mathcal{X}_{\hat{f}_i}\}} \Big[\log \hat{f}_i(O_i, \mathcal{X}_{\hat{f}_i}, Y_i|\mathcal{M}) \Big]}^{E_{q(\mathcal{X})}\text{-step}} \right) \tag{4.22}$$

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

$$E_{\{q^{(t)}(X_j;\beta_{X_j}) \,|\, X_j \in \mathcal{X}_{\hat{f}_i}\}} \Big[\log \hat{f}_i(O_i, \mathcal{X}_{\hat{f}_i}, Y_i|\mathcal{M}) \Big] \\ = \sum_{\mathcal{X}_{\hat{f}_i}} \prod_{X_j \in \mathcal{X}_{\hat{f}_i}} q(X_j;\beta_{X_j}) \log \hat{f}_i(O_i, \mathcal{X}_{\hat{f}_i}, Y_i|\mathcal{M}) \tag{4.23}$$

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. (4.23) and (4.22) in Eq. (4.21), we also have that

$$\alpha_{Y_i}^{(\mathrm{t+1})} = \underset{\alpha_{Y_i}}{\operatorname{argmax}} \ \sum_{Y_i} q(Y_i;\alpha_{Y_i}) \ \log \ q(Y_i;\alpha^*_{Y_i}) + H(q(Y_i;\alpha_{Y_i}) - \log \mathcal{Z}_{Y_i} \tag{4.24a}$$

$$= \underset{\alpha_{Y_i}}{\operatorname{argmax}} \ - KL\Big[q(Y_i;\alpha_{Y_i}) \;||\; q(Y_i;\alpha^*_{Y_i}) \Big] + const. \tag{4.24b}$$

$$= \underset{\alpha_{Y_i}}{\operatorname{argmax}} \ - KL\Big[q(Y_i;\alpha_{Y_i}) \;||\; q(Y_i;\alpha^*_{Y_i}) \Big], \tag{4.24c}$$

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. (4.24b), is simply the logarithm of the normalization factor representing the variables' $\{q(X_j;\beta_{X_j})\}$ distributions, that are independent of $q(Y_i;\alpha_{Y_i})$. Note that, from Eq. (4.24c), 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

$$q(X_j;\beta^*_{X_j}) = \frac{1}{\mathcal{Z}_{X_j}} \exp\left(\overbrace{E_{\{q^{(t+1)}(Y_i;\alpha_{Y_i}),\, q^{(t)}(X_k;\beta_{X_k}) \mid \hat{f}_i \in \hat{\mathcal{F}}_{X_j}\}}\left[\log \hat{F}(O, X_j, \mathcal{Y}|\mathcal{M})\right]}^{E_{q(\mathcal{Y})}\text{-step}} \right) \tag{4.25}$$

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' $\{q(Y_i;\alpha_{Y_i})\}$ distributions in the Markov boundary of each X_j, and the $\{q(X_k;\beta_{X_k})\}$ from the old iteration for the other variables $X_k \neq X_j$ that are arguments to factors in which X_j appears.

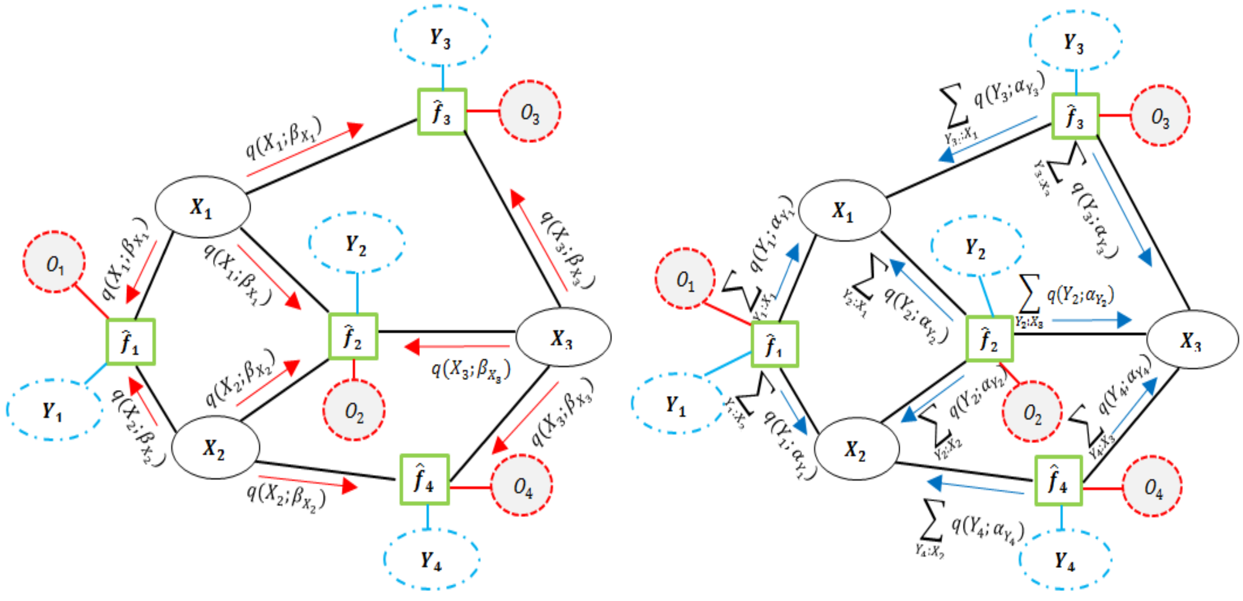

Figure 4.2 Illustrating message-passing process of GEM-MP. (*left*) $E_{q(\mathcal{X})}$-step messages from variables-to-factors; (*right*) $E_{q(\mathcal{Y})}$-step messages from factors-to-variables.

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[3] of

[3] This is equivalent to converging to a local minimum of the negative free energy functional $-\mathcal{F}_{\mathcal{M}}$, which

$\mathcal{F}_{\mathcal{M}}(q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}), q(\mathcal{X};\mathcal{B}_{\mathcal{X}}))$. These messages are different from simply running the standard LBP algorithm, but their structures 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. (4.15), (4.22), (4.16), and (4.25), 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 4.2 (*left*)). The aim of sending these messages is to perform the GEM-MP's $M_{q(\mathcal{Y})}$-step in Eq. (4.15). 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 expected log marginal-likelihood of Eq. (4.15). 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. (4.15). 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 4.2).

- **$E_{q(\mathcal{Y})}$-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 4.2 (*right*)). Sending these messages corresponds to the GEM-MP's $M_{q(\mathcal{X})}$-step in Eq. (4.16). 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 expected log marginal-likelihood in Eq. (4.16). 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. (4.16). 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 4.2).

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 4.2 we will derive a general update-rule for GEM-MP based on vari-

is a stable fixed point with respect to an inference task on the original factor graph.

ational 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 $\mathcal{Y}$). 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 $q(\mathcal{X}_j)$s approximations for variables $X_j \in \mathcal{X}$ could be computed in parallel, and updates to the $q(\mathcal{Y}_i)$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 $\mathcal{B}_{\mathcal{X}}$), 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. In fact, maximizing the lower bound $\mathcal{F}_{\mathcal{M}}(q(\mathcal{X};\mathcal{B}_{\mathcal{X}}), q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}))$ is equivalent to minimizing the Kullback-Leibler (KL) divergence between $q(x;\mathcal{B}_{\mathcal{X}})$ and $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ and the true posterior distribution, $P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})$, over hidden variables:

$$\log \sum_{\mathcal{X},\mathcal{Y}} P(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M}) - \mathcal{F}_{\mathcal{M}} = \sum_{\mathcal{X},\mathcal{Y}} q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) \, \log \frac{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}{P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})} \tag{4.26a}$$

$$= KL\Big[q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}}) \,||\, P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \tag{4.26b}$$

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 4.3):

$$KL\Big[q^{(t)}_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} \, q^{(t)}_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} \,||\, P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \geq KL\Big[q^{(t)}_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})} \, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} \,||\, P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\Big] \tag{4.27}$$

and similarly:

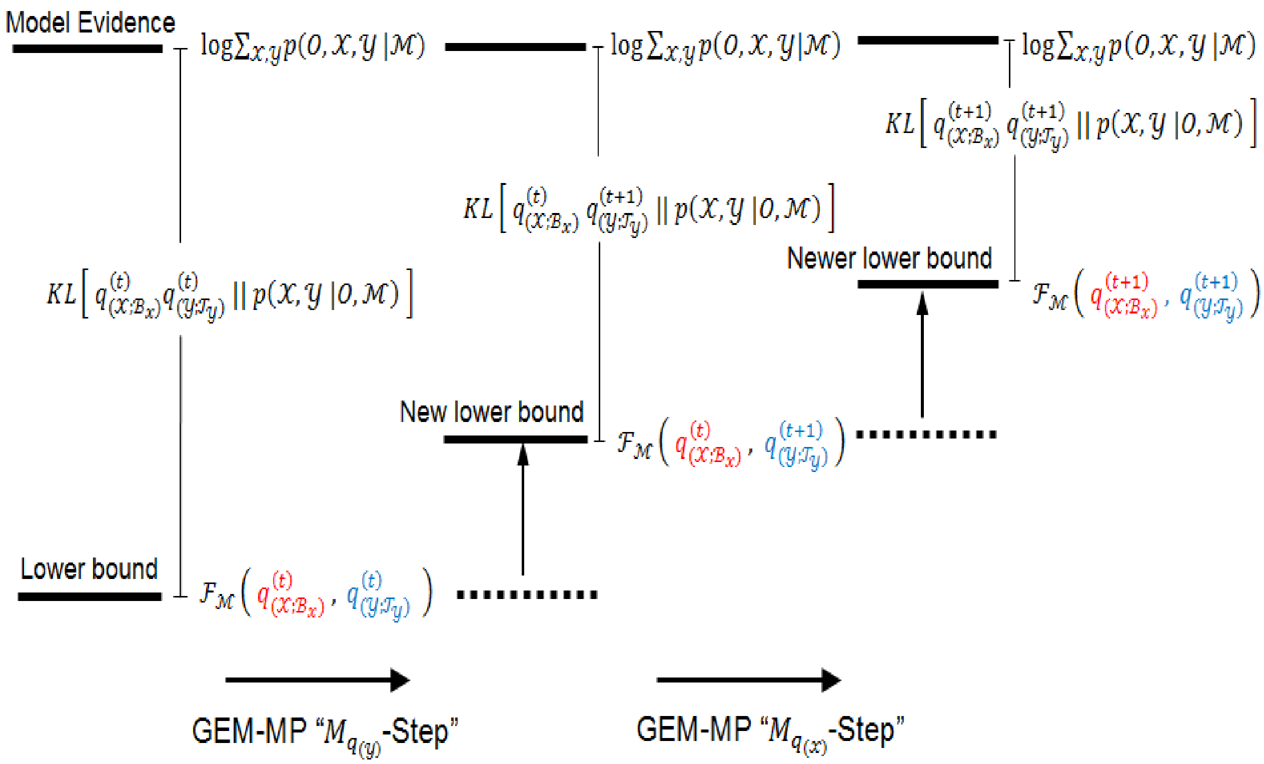
Model Evidence
$\log \sum_{\mathcal{X},\mathcal{Y}} p(\mathcal{O},\mathcal{X},\mathcal{Y}\,|\mathcal{M})$
$\log \sum_{\mathcal{X},\mathcal{Y}} p(\mathcal{O},\mathcal{X},\mathcal{Y}|\mathcal{M})$
$\log \sum_{\mathcal{X},\mathcal{Y}} p(\mathcal{O},\mathcal{X},\mathcal{Y}\,|\mathcal{M})$
$KL\left[\, q^{(t+1)}_{(\mathcal{X};\mathcal{B}_x)}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_y)} \,||\, p(\mathcal{X},\mathcal{Y}\,|\mathcal{O},\mathcal{M}) \right]$
$KL\left[\, q^{(t)}_{(\mathcal{X};\mathcal{B}_x)}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_y)} \,||\, p(\mathcal{X},\mathcal{Y}\,|\mathcal{O},\mathcal{M}) \right]$
Newer lower bound
$KL\left[\, q^{(t)}_{(\mathcal{X};\mathcal{B}_x)} q^{(t)}_{(\mathcal{Y};\mathcal{T}_y)} \,||\, p(\mathcal{X},\mathcal{Y}\,|\mathcal{O},\mathcal{M}) \right]$
$\mathcal{F}_{\mathcal{M}}\left(q^{(t+1)}_{(\mathcal{X};\mathcal{B}_x)},\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_y)} \right)$
New lower bound
$\mathcal{F}_{\mathcal{M}}\left(q^{(t)}_{(\mathcal{X};\mathcal{B}_x)},\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_y)} \right)$
Lower bound
$\mathcal{F}_{\mathcal{M}}\left(q^{(t)}_{(\mathcal{X};\mathcal{B}_x)},\, q^{(t)}_{(\mathcal{Y};\mathcal{T}_y)} \right)$
GEM-MP “$M_{q(\mathcal{Y})}$-Step”
GEM-MP “$M_{q(\mathcal{X})}$-Step”

Figure 4.3 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. (4.15). Then, in its “$M_{q(\mathcal{X})}$-step”, the variational distribution over hidden $\mathcal{X}$ variables is maximized according to Eq. (4.16).

$$KL\left[q^{(t)}_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} \mid\mid P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\right] \geq KL\left[q^{(t+1)}_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}\, q^{(t+1)}_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})} \mid\mid P(\mathcal{X},\mathcal{Y}|\mathcal{O},\mathcal{M})\right] \tag{4.28}$$

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.

Now since GEM-MP applies variational mean-field approximation for $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$ and $q(\mathcal{X};\mathcal{B}_{\mathcal{X}})$ distributions (refer to Eqs. (4.18) and (4.17)) over both mega-nodes and variables nodes respectively, then it inherits the guaranteed convergence of mean field method to a local minimum in the likelihood space. □

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", we can select between local or global approximation to distribution $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$. However, we restricted ourselves to local approximations.[4] 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 4.2) and Ising models (as will be explained in Section 4.5)

4.2 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

[4] 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. (4.17)).

the underlying factor graph often contains hard and soft clauses, then the GEM-MP will distinguish two update-rules (*Hard-update-rule* and *Soft-update-rule*) for tackling hard and soft clauses, respectively.

4.2.1 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 habitual $M_{q(\mathcal{Y})}$-step. So, we presume that we have already the constructed posterior distribution $q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})$. Also, we suppose here that all clauses are only hard clauses.

1. $M_{q(\mathcal{X})}$**-step:** Recalling Section 4.1, 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 $\beta_{X_j} \in \mathcal{B}_{\mathcal{X}}$ must be a proper probability distribution. Thus, we have an optimization problem of the form:[5]

$$\begin{aligned} \max_{\mathcal{B}_{\mathcal{X}}} \quad & E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\Big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\Big] \\ \text{s.t.} \quad & \left(\beta^{+}_{X_j} + \beta^{-}_{X_j}\right) = 1, \quad \forall \beta_{X_j} \in \mathcal{B}_{\mathcal{X}} \end{aligned} \tag{4.29}$$

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

$$\Lambda(\mathcal{B}_{\mathcal{X}}) = E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\Big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\Big] - T(\mathcal{B}_{\mathcal{X}}) \tag{4.30}$$

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:

$$T(\mathcal{B}_{\mathcal{X}}) = \sum_{X_j \in \mathcal{X}} \lambda_{X_j}\left(1 - \beta^{+}_{X_j} - \beta^{-}_{X_j}\right) \tag{4.31}$$

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:

$$\begin{aligned} & E_{q_{(\mathcal{X};\mathcal{B}_{\mathcal{X}})}q_{(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}}\Big[\log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M})\Big] \\ & \qquad = \sum_{Y_i \in \mathcal{Y}} q(Y_i;\alpha_{Y_i}) \sum_{X_j \in \mathcal{X}} q(X_j;\beta_{X_j}) \log P(\mathcal{O},\mathcal{Y}|\mathcal{X},\mathcal{M}) \end{aligned} \tag{4.32}$$

[5] 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 posterior distribution, $\log P(\mathcal{O}, \mathcal{Y}|\mathcal{X}, \mathcal{M})$, into individual distributions corresponding to hard ground clauses, and we have:[6]

$$E_{q(\mathcal{X};\mathcal{B}_{\mathcal{X}})q(\mathcal{Y};\mathcal{T}_{\mathcal{Y}})}\Big[\log P(\mathcal{O}, \mathcal{Y}|\mathcal{X}, \mathcal{M})\Big] \propto \sum_{X_j, Y_i} q(X_j; \beta_{X_j}) q(Y_i; \alpha_{Y_i}) \times \log\Bigg[\prod_{f_i^h \in \mathcal{F}^h} P(Y_i|\mathcal{X}, \mathcal{M})\Bigg] \tag{4.33}$$

Where $P(Y_i|\mathcal{X}, \mathcal{M})$ is the posterior 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:

$$\log\Bigg[\prod_{f_i^h \in \mathcal{F}^h} P(Y_i|\mathcal{X}, \mathcal{M})\Bigg] \approx \log\Bigg[\prod_{f_i^h \in \mathcal{F}^h} \prod_{X_j \in \mathcal{X}_{f_i^h}} \beta_{X_j}(Y_i(X_j))\Bigg] \tag{4.34}$$

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. (4.34), 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.

[6] 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 deeming only the valid local entries of the hard ground clauses.

Now, take the value of Eq. (4.34), and substituting it for Eq. (4.33). Next, we convert logarithms of products into sums of logarithms and exchanging summations, and handle each hard ground clause $f_i^h \in \mathcal{F}^h$ separately in a sum.

Subsequently, we then take the partial derivative of the Lagrangian function, in Eq. (4.30), with respect to an individual ground atom positive marginal $\beta^+_{X_j}$ and equate to zero:[7]

$$\frac{\partial}{\partial \beta^+_{X_j}}\left[\Lambda(\mathcal{B}_{\mathcal{X}})\right] = 0 \Rightarrow \beta^+_{X_j} = \frac{1}{\lambda_{X_j}} \underbrace{\left(\sum_{f_i^h \in \mathcal{F}^h_{X_j}} \overbrace{\sum_{Y_i:\forall y_k(X_j)=\text{``}+\text{''}} q(Y_i;\alpha_{Y_i})}^{(E_{q(\mathcal{Y})}\text{-step message})\ \ \mu_{f_i \to X_j}} \right)}_{Weight^+_{X_j}} \tag{4.35}$$

Where $\sum_{Y_i:\forall y_k(X_j)=\text{``}+\text{''}} q(Y_i;\alpha_{Y_i})$ is the $E_{q(\mathcal{Y})}$-step message that X_j will receive from each hard ground clause ($f_i^h \in \mathcal{F}^h_{X_j}$) conveying what it believes about X_j's positive marginal. Each $E_{q(\mathcal{Y})}$-step message is computed by adding a term for only those valid local entries ($Y_i : \forall y_k(X_j) = \text{``}+\text{''}$) 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 $\beta^-_{X_j}$:

$$\frac{\partial}{\partial \beta^-_{X_j}}\left[\Lambda(\mathcal{B}_{\mathcal{X}})\right] = 0 \Rightarrow \beta^-_{X_j} = \frac{1}{\lambda_{X_j}} \underbrace{\left(\sum_{f_i^h \in \mathcal{F}^h_{X_j}} \overbrace{\sum_{Y_i:\forall y_k(X_j)=\text{``}-\text{''}} q(Y_i;\alpha_{Y_i})}^{(E_{q(\mathcal{Y})}\text{-step message})\ \ \mu_{f_i(Y_i) \to X_j}} \right)}_{Weight^-_{X_j}} \tag{4.36}$$

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

$$\begin{gathered} \beta^-_{X_j} + \beta^+_{X_j} = 1 \overset{From\ Eqs.\ (4.35)\ and\ \ (4.36)}{\Rightarrow} \\ \left[\left(Weight^+_{X_j} + Weight^-_{X_j}\right)/\lambda_{X_j} = 1\right] \Rightarrow \lambda_{X_j} = \left(Weight^+_{X_j} + Weight^-_{X_j}\right) \end{gathered} \tag{4.37}$$

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. $\beta_{X_j} = \left[\beta^-_{X_j}, \beta^+_{X_j}\right]$).

[7] This implies taking the derivative of both expected log-likelihoods, which we obtain after substituting the value of Eq. (4.34) for Eq. (4.33), and the penalty term in Eq. (4.31).

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. (4.35) and (4.36).

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, $\sum_{Y_i:\forall y_k(X_j)=``-"}$ involves enumerating all the valid local entries for each $Y_i$ which is intractable. Instead, we will approximate the distribution $\sum_{Y_i:\forall y_k(X_j)=``-"} q(Y_i;\alpha_{Y_i})$ for each hard ground clause $f_i^h \in \mathcal{F}^h$ by using a probability $1-\xi(X_j, f_i^h)$, which we call the *probabilistic generalized arc consistency* (pGAC). At this point, let us pause to elaborate more on pGAC in the next subsection.

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 $X_j =$"$-$" is not generalized arc consistent with respect to f_i when those other ground atoms appearing in f_i are all individually inconsistent with this assignment since $X_j =$"$-$" can belong to an invalid local entry of f_i. This means that there is a probability that $X_j = d$ is unsatisfiable 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, and we assume Independence and approximate it as:

$$\xi(X_j, f_i) = \left(\prod_{X_k \in \mathcal{X}_{f_i^+} \setminus \{X_j\}} \left(1 - \beta_{X_k}^+\right) \cdot \prod_{X_k \in \mathcal{X}_{f_i^-} \setminus \{X_j\}} \left(\beta_{X_k}^+\right) \right) \tag{4.38}$$

As indicated in Eq. (4.38), $\xi(X_j, f_i)$ 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 13 (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:*

$$0 < 1 - \xi(X_j, f_i) \leq 1 \tag{4.39}$$

The definition of the traditional GAC in Section 2.4.1 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 almost 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 2.4.1). 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. (4.38), 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 it will be shown in Propositional 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 $X_j = +$ samples the valid local entries of f_i that involve $X_j = +$, in a closed form. Thus we have that:

$$\Big[1 - \xi(X_j, f_i)\Big]\Big|_{X_j=\text{``}+\text{''}} \propto \sum_{Y_i: \forall y_k(X_j)=\text{``}+\text{''}} q(Y_i; \alpha_{Y_i}) \tag{4.40}$$

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

$$\Big[1 - \xi(X_j, f_i)\Big]\Big|_{X_j=\text{``}-\text{''}} \propto \sum_{Y_i: \forall y_k(X_j)=\text{``}-\text{''}} q(Y_i; \alpha_{Y_i}) \tag{4.41}$$

Based on Eqs. (4.40) and (4.41), we can use pGAC for computing the two components of $E_{q(\mathcal{Y})}$-step message, in Eqs. (4.35)and (4.36), 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, $X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}$. Hence, simultaneously passing the $E_{q(\mathcal{X})}$-step messages, which conveying the marginals, from ground atoms $\mathcal{X}_{f_i}$ to f_i could be one of the best choices. 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, and 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 of 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.

Using pGAC in the derivation of 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. (4.35) and (4.36) as follows:

$$Weight^{+}_{X_j} = \sum_{f_i^h \in \mathcal{F}^h_{X_j}} \sum_{Y_i:\forall y_k(X_j)=\text{``+''}} q(Y_i; \alpha_{Y_i}) \tag{4.42a}$$

$$= \left[\sum_{f_i^h \in \mathcal{F}^h_{X_j+}} \sum_{Y_i:\forall y_k(X_j)=\text{``+''}} q(Y_i; \alpha_{Y_i}) \right] + \left[ \sum_{f_i^h \in \mathcal{F}^h_{X_j-}} \sum_{Y_i:\forall y_k(X_j)=\text{``+''}} q(Y_i; \alpha_{Y_i}) \right] \tag{4.42b}$$

$$\approx \sum_{f_i^h \in \mathcal{F}^h_{X_j+}} [1] + \sum_{f_i^h \in \mathcal{F}^h_{X_j-}} (1 - \xi(X_j, f_i^h)) \tag{4.42c}$$

$$= \left| \mathcal{F}^h_{X_j} \right| - \sum_{f_i^h \in \mathcal{F}^h_{X_j-}} \xi(X_j, f_i^h) \tag{4.42d}$$

Where, in Eq. (4.42b), 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. (4.42c), in the first positive summation, we replaced the inner summation with the constant value of 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 4.2.1).

The end result, as in Eq. (4.42d), 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 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:

$$Weight^{-}_{X_j} = \left| \mathcal{F}^h_{X_j} \right| - \sum_{f_i^h \in \mathcal{F}^h_{X_j+}} \xi(X_j, f_i^h) \tag{4.43}$$

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

4.2.2 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 4.1(*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:

$$\alpha_{Y_i}\left(f_i^h\right) = \begin{cases} 1 & \text{if the state of } Y_i \text{ satisfies } f_i^h \ , \\ 0 & \text{Otherwise.} \end{cases}$$
$$\alpha_{Y_i}\left(f_i^s\right) = \begin{cases} \exp(w_{f_i^s}) & \text{if the state of } Y_i \text{ satisfies } f_i^s, \\ 1 & \text{Otherwise.} \end{cases} \tag{4.44}$$

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 hard updating rule, in Eq. (4.42d). Note that the idea of using the exponential form in $\alpha_{Y_i}\left(f_i^s\right)$ is to imitate the soft clauses for assigning weights to the local entries. In MLNs, each soft ground clause f_i^s assigns the valid local entries a value equal to $\exp(w_{f_i^s})$. Otherwise it assigns a value 1 (i.e., exp(0)) to the invalid local entries. This implies taking the exponential transformation as follows:

$$\overset{Softening}{\Rightarrow} \beta^{+}_{X_j} = \frac{1}{\lambda_{X_j}} \left(\sum_{Y_i:\forall y_k(X_j)=\text{“+”}} \underbrace{\exp\left[\sum_{f_i^h\in\mathcal{F}^h_{X_j}} q(Y_i;\alpha_{Y_i}(f_i^h))\right]}_{\prod_{f_i^s\in\mathcal{F}^s_{X_j}} \exp\left(q(Y_i;\alpha_{Y_i}(f_i^h))\right)} \right) \tag{4.45}$$

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

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

where

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

.

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 $\sum_{Y_i:\forall y_k(X_j)=\text{``}+\text{''}}$ in Eq. (4.45)) 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), which ending up with:

$$\overset{Embedding}{\Rightarrow} \beta^+_{X_j} = \frac{1}{\lambda_{X_j}} \underbrace{\left(\prod_{f_i^s \in \mathcal{F}^s_{X_j}} q\left(Y_i;\alpha_{Y_i}(f_i^s)\right)\right)}_{Weight^+_{X_j}} \tag{4.46}$$

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

$$\begin{aligned}
Weight^{+}_{X_j} &= \prod_{f_i^s \in \mathcal{F}^s_{X_j}} q\left(Y_i; \alpha_{Y_i}(f_i^s)\right) && \text{(4.47a)}\\
&= \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j+}} q\left(Y_i; \alpha_{Y_i}(f_i^s)\right)\right] \times \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j-}} q\left(Y_i; \alpha_{Y_i}(f_i^s)\right)\right] && \text{(4.47b)}\\
&\approx \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j+}} \exp(w_{f_i^s})[1]\right]\\
&\quad \times \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j-}} \Big[(1-\xi(X_j, f_i^s))\exp(w_{f_i^s}) + \xi(X_j, f_i^s)\cdot 1\Big]\right] && \text{(4.47c)}\\
&= \left[\exp(\sum_{f_i^s \in \mathcal{F}^s_{X_j}} w_{f_i^s})\right]\\
&\quad - \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j+}} \exp(w_{f_i^s}) \times \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j-}} \xi(X_j, f_i^s)(\exp(w_{f_i^s})-1)\right]\right] && \text{(4.47d)}
\end{aligned}$$

Note that comparing Eq. (4.47c) to its corresponding Eq. (4.42c) for the update rules of the hard factors, we have an additional term "$\xi(X_j, f_i^s)\cdot 1$" in the second summation. This is because computing the second part of Eq. (4.47b) implies computing two terms as appeared in the second part of Eq. (4.47c): the first is $(1-\xi(X_j, f_i^s))$ representing the probability that X_j being positive satisfies the factor f_i^s that include X_j as negative ground atom, and therefore it is multiplied by $\exp(w_{f_i^s})$ since at the satisfaction of soft ground clause f_i^s we obtain $\exp(w_{f_i^s})$. The second term is $\xi(X_j, f_i^s)$ representing the probability that X_j being positive dissatisfies the factor f_i^s, and therefore it is multiplied by 1 since at the dissatisfaction of f_i^s we obtain 1. This "$\xi(X_j, f_i^s)\cdot 1$" term disappeared from the update rules of the hard factors in Eq. (4.42c) because $\xi(X_j, f_i^s)$ 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:

$$\begin{aligned}
Weight^{-}_{X_j} &= \left[\exp(\sum_{f_i^s \in \mathcal{F}^s_{X_j}} w_{f_i^s})\right]\\
&\quad - \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j-}} \exp(w_{f_i^s}) \times \left[\prod_{f_i^s \in \mathcal{F}^s_{X_j+}} \xi(X_j, f_i^s)(\exp(w_{f_i^s})-1)\right]\right] && \text{(4.48)}
\end{aligned}$$

Note that the weight parts (in Eqs. (4.47d) and (4.48)) used for the soft-update-rule, are soft versions of previously derived weight parts (in Eqs. (4.42d) and (4.43)) 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. (4.42d), (4.43), (4.47d) and (4.48) and substitute these for the $Weight^{+}_{X_j}$ and $Weight^{-}_{X_j}$ in Eqs. (4.35) and (4.36) to obtain our ultimate set of GEM-MP's rules in order to update the marginals of query ground atoms. This is in Table 4.2. 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 GEM-MP framework requires re-deriving 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 4.2 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 necessitate 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. (4.35) and (4.36) vs. LBP in Eqs. (2.10) and (2.11) 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. (4.10c), 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. (4.7) 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) geometrical average of the incoming messages in a setting where there is no such bound.

The second important difference between both 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 multiplying 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. Conversely, 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, the GEM-MP's outgoing $E_{q(\mathcal{Y})}$-step message is an approximate posterior 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 posterior 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 4.2.1). 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. 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 3 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 $X_j \in \mathcal{X}$ do
 2:     $\beta_{X_j} \leftarrow \mathcal{U}\,[0,1]$;
 3: end for
    // discriminate query atoms.
 4: $\mathcal{X}_h \leftarrow X_j \in \mathcal{X}$; // involved in hard ground clauses $\mathcal{F}^h \in \mathcal{F}$.
 5: $\mathcal{X}_s \leftarrow X_k \in \mathcal{X}$; // involved in soft ground clauses $\mathcal{F}^s \in \mathcal{F}$.
    // inferring marginals
 6: repeat
 7:     for each $X_j \in \mathcal{X}_h$ do
 8:         $\beta_{X_j} \leftarrow$ Hard-Update-Rule; // as in Table 4.2
 9:     end for
10:     for each $X_k \in \mathcal{X}_s$ do
11:         $\beta_{X_j} \leftarrow$ Soft-Update-Rule; // as in Table 4.2
12:     end for
13: until convergence or termination of $\mathcal{I}_{\max}$
14: Return $\mathcal{B}_{\mathcal{X}}$;
```

Algorithm 3 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 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 hard-update-rule (lines 7-9). Then it updates the marginals for query atoms of the second subset by applying 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[8]) or until the termination of 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 Soft-update-rule, and propagating the information about soft ground clauses to query atoms in $\mathcal{X}_h$ when it is used by 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 in the onset and then recall their values.

4.5 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 (as defined in Subsection 2.2.1) with arbitrary topology which are a specific subset of the canonical (pairwise) Markov random fields (MRFs). Although pairwise MRFs are commonly used as a benchmark for inference because they have a simple and compact representation, they often pose a challenge for inference. Now if we consider the factor graph representation of an Ising Model, each uni-variate potential $\phi_i(X_i)$ can be represented as a unit clause (factor node) involving only one variable X_i with associated weight θ_i in which it equals e^{θ_i} when it is

[8] Note that ϵ is commonly assigned a very small value (e.g., 10^{-8}) to ensure that the algorithm converges properly.

satisfied and 1 otherwise. Similarly, each $\phi_{ij}(X_i, X_j)$ can be formulated as a conjunction of two clauses[9] $\Big[(\neg X_i \vee X_j) \wedge (X_i \vee \neg X_j)\Big]$, 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: $\Big[(\neg X_i \vee X_j) \wedge (X_i \vee \neg X_j), \theta_{ij} = \eta \cdot C\Big]$, $\forall X_i, X_j \in \mathcal{E}$

Now, without difficulty, we can directly apply the soft-update-rule from Table 4.2 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 uni-variate 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 to e^{∞} and all its invalid local entries equal to 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 4.2 when computing the marginals.

4.6 Experimental 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?

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

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 to a uniform distribution*

To answer these 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[10] 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 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[11] that are built-in to the Alchemy system[12](Kok *et al.*, 2007):

- **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) and improved the one proposed by Gogate *et al.* (2012).

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

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

[12] Alchemy 0.2 software is one of the powerful tools that is based on probabilistic theorem proving inference engine for performing inference on Markov logic models. It is publicly available at: `http://alchemy.cs.washington.edu/`

MC-SAT rapidly reasons in the limit of determinism and L-Im is the recent lifted importance sampling inference 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. Additionally, 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 not feasible for the underlying datasets.

4.6.1 Datasets

Cora. This dataset consists of 1295 citations of 132 different computer science papers[13]. Recently, the dataset was cleaned and split by Singla and Domingos (2006a) into five subsets for cross-validation.

- **MLN:** We used the MLN model which is similar to the established one of Singla and Domingos (2006a). 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. Also it 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).
- **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[14]. It contains four subsets, each of which contains the information about 450 proteins.

[13] Primarily, it was labeled by Andrew McCallum: `https://www.cs.umass.edu/~Cmccallum/data/cora-refs.tar.gz`

[14] Originally, it was prepared by Munich Information Center for Protein Sequence.

- **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[15]. 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)).

4.6.2 Metrics

Since computing the exact posterior marginals is not feasible for the underlying domains, we evaluated the accuracy using two metrics:

- Average conditional log marginal-likelihood (CLL). The CLL, which approximates the KL-divergence between the actual and computed marginals returned by an inference

[15] Available at: `http://alchemy.cs.washington.edu/data/uw-cse/`

algorithm for 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 CLL of a query atom is computed by averaging the log-marginal probabilities of the true values over all its groundings.

- Balanced F_1 score. 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).

4.6.3 Methodology and Results

All the experiments were run on a cluster of nodes with multiprocessors running at 2.4 GHz Intel CPUs and 4 GB of RAM with 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 the experimental evaluations through five experiments.

Experiment I

The first experiment was dedicated to answer 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, for producing the marginals of all groundings of query atoms being true. Such marginal probabilities were used to compute F_1 and average CLL metrics.

Although, a traditional way to assess the inference algorithms would be to run them until convergence and compare their running times and their accuracy, diagnosing the convergence in this way is problematic 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 the same running time that is 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 calculated the F_1 based on the results we obtained at the end of the allotted time on only one held-out sub-dataset, which has the grounding truth: Cora (contains ≈ 35659 ground truth: 11234 for *SameBib*, 12640 for *SameTitle*, 5339 for *SameAuthor* and 6441 for *SameVenue*), Yeast (contains ≈ 46225: 21097 for *Interaction* and 25128 for *Function*), and UW-CSE (contains ≈ 76856 for *AdvisedBy*)

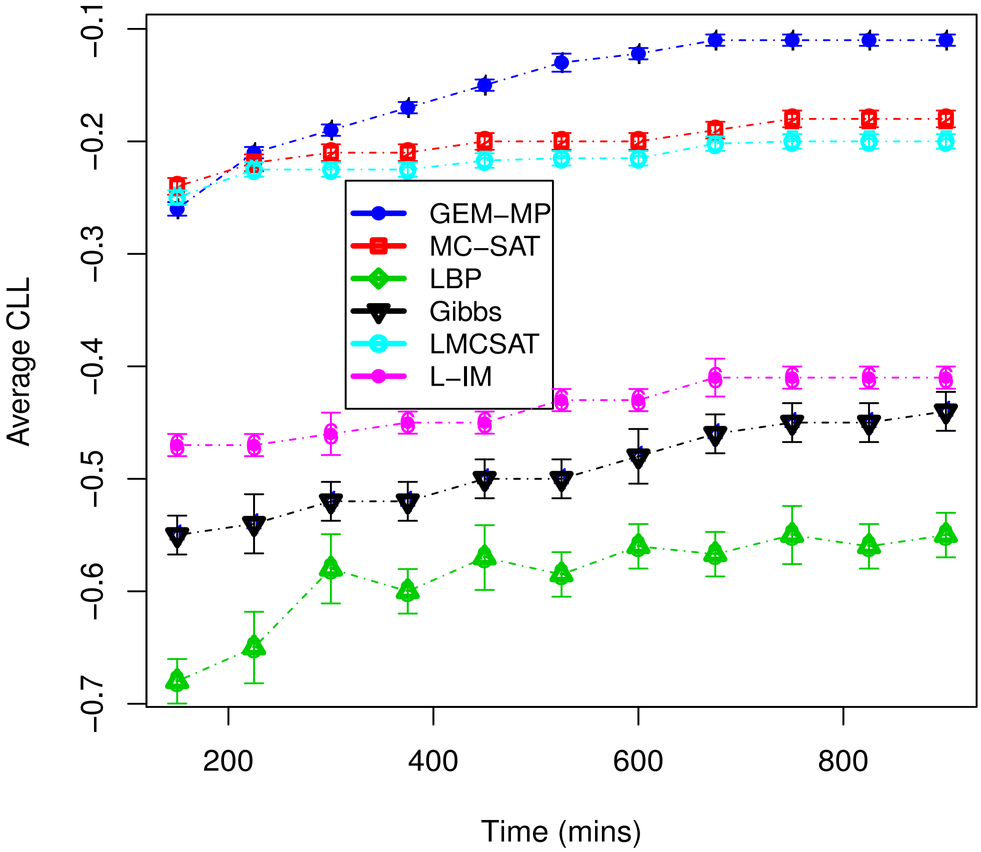

Figure 4.4 Average CLL as a function of inference time for GEM-MP, MC-SAT, LBP, Gibbs, LMCSAT, and L-Im algorithms on Cora.

Figures 4.4, 4.5, and 4.6 show 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 (i.e.,

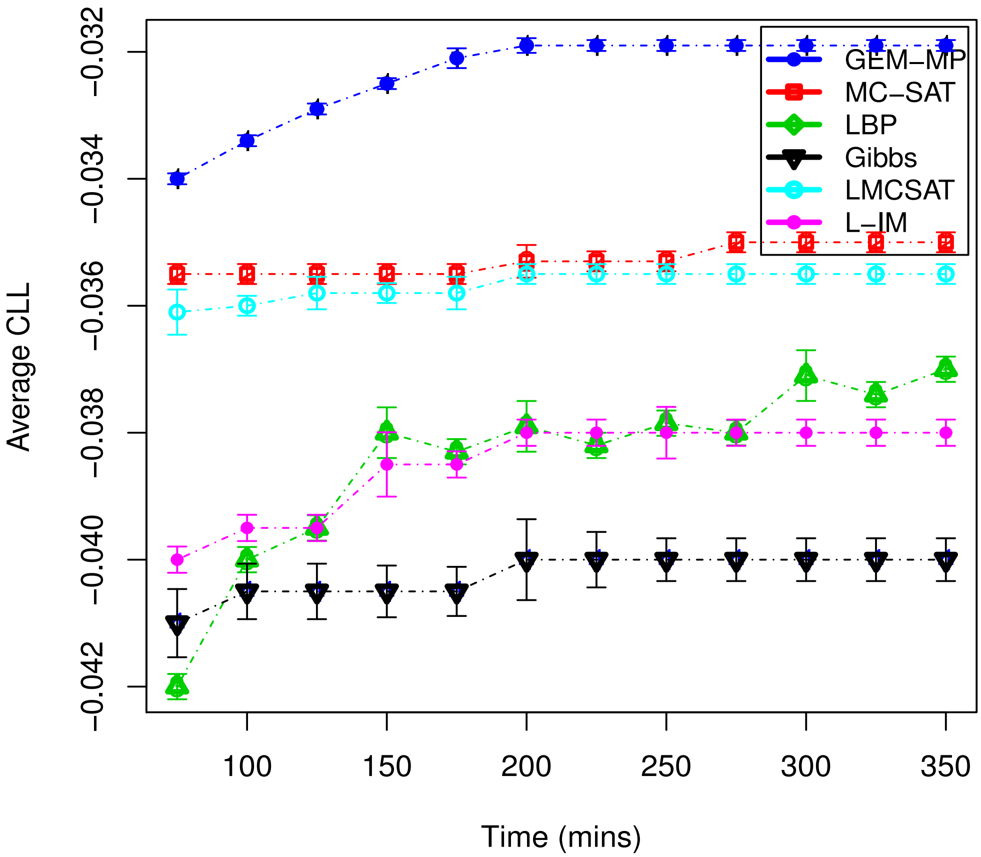

Figure 4.5 Average CLL as a function of inference time for GEM-MP, MC-SAT, LBP, Gibbs, LMCSAT, and L-Im algorithms on Yeast.

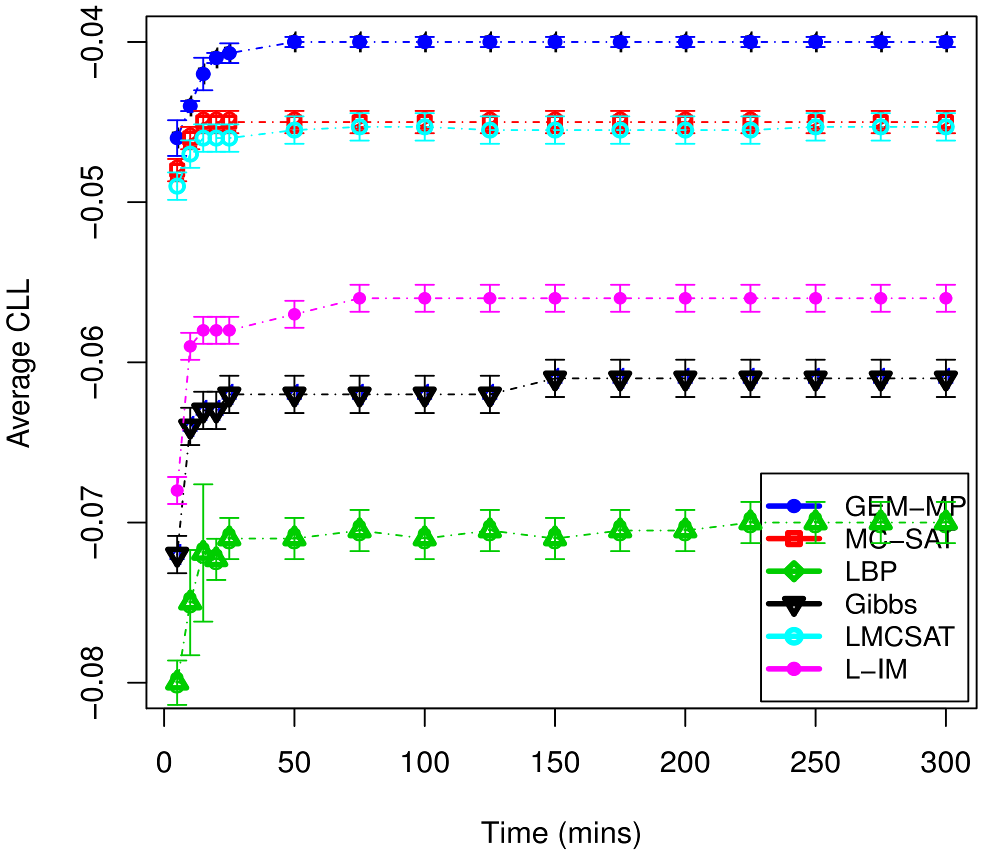

Figure 4.6 Average CLL as a function of inference time for GEM-MP, MC-SAT, LBP, Gibbs, LMCSAT, and L-Im algorithms on UW-CSE.

per marginals node). 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 compared algorithms, achieving the best average CLL on Yeast and UW-CSE datasets (*this answers Q1*).[16] For the Cora dataset, it took about 225 minutes to dominate all other inference algorithms.[17] MC-SAT came close behind GEM-MP on both Cora and UW-CSE, but considerably further 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 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 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 more than LBP's accuracy with large margins on Cora and UW-CSE, and slightly less accurate than LBP on the Yeast dataset. The accuracy of MC-SAT and its lazy algorithm (LMCSAT) were very close on all the used datasets.

Table 4.3 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 4.3 reports the average F_1 scores for the inference algorithms. The results complement those of Figures 4.4, 4.5 and 4.6, underscoring the promise of our proposed GEM-MP algorithm for obtaining the highest quality among the alternatives for predicting marginals,

[16] Note that the average standard deviation error bars do not overlap between the GEM-MP's curve and MC-SAT's curve at any point, which means that the difference between the two curves is statistically significant (with $p = 0.05$).

[17] Note that, for the Cora dataset, the construction of the grounded network required for inference takes $\approx$ 185 minutes on average.

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 algorithms on all datasets, achieving 39%, 37% and 33% greater accuracy than LBP, Gibbs and L-IM 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*). Both Gibbs and LBP rivaled each other on the tested datasets, and both were overshadowed by MC-SAT. LMCSAT was very competitive to its propositional MC-SAT with approximately 2.2% loss in accuracy.

Experiment II

Here we concentrated on Q3. To obtain robust answers, we examined the performance of GEM-MP, MC-SAT and LBP algorithms at varying amounts of determinism. Thus, we re-ran experiment I, but at gradual amounts of determinism. We marked each amount of determinism as a *zone*, where determinism-zones are $\in [0, 50]$. Note that the 0-zone stands for zero percentage of determinism (i.e., all clauses in the model are considered soft clauses), and 50-zone means 50 percentage of determinism (i.e., we considered 50% of the clauses in the model as hard clauses and 50% as soft clauses). Each zone of determinism in the range would represent a level of comfort[18] or discomfort for both GEM-MP and LBP algorithms.

Figures 4.7, 4.8, and 4.9 report the average CLL as a function of time for GEM-MP and LBP inference algorithms at different determinism zones. 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 traversing from one zone of determinism to the next, where a greater percentage of determinism is present, we observe an increase in the accuracy of GEM-MP and a decrease in the accuracy of LBP. Also, on all datasets, at each zone of determinism, GEM-MP prevailed over the corresponding LBP in terms of accuracy of results (*answering Q3*). In addition, the greater amount of determinism in the zone, the greater the convergence for GEM-MP, and the greater the non-convergence for LBP (*answering Q2*). Remarkably, the 0-zone, which has no amount of determinism, represents the most uncomfortable[19]level for GEM-MP. Conversely, it was the most comfortable level for LBP. However, even in this zone, GEM-MP surpassed LBP on all datasets. For MC-SAT, increasing the determinism in the model has a small impact on its accuracy since traversing from one zone to a higher one with a larger percentage of determinism, a decrease in its

[18] Note that we used the term “Comfort” to reflect the quality of the convergence and the accuracy of the marginals obtained from the algorithm.

[19] Note that, at 0-zone, the 0-amount of determinism (i.e., no hard clauses) eliminates the usage of the Hard-update-rule from GEM-MP, which could place GEM-MP outside of its comfortable zone.

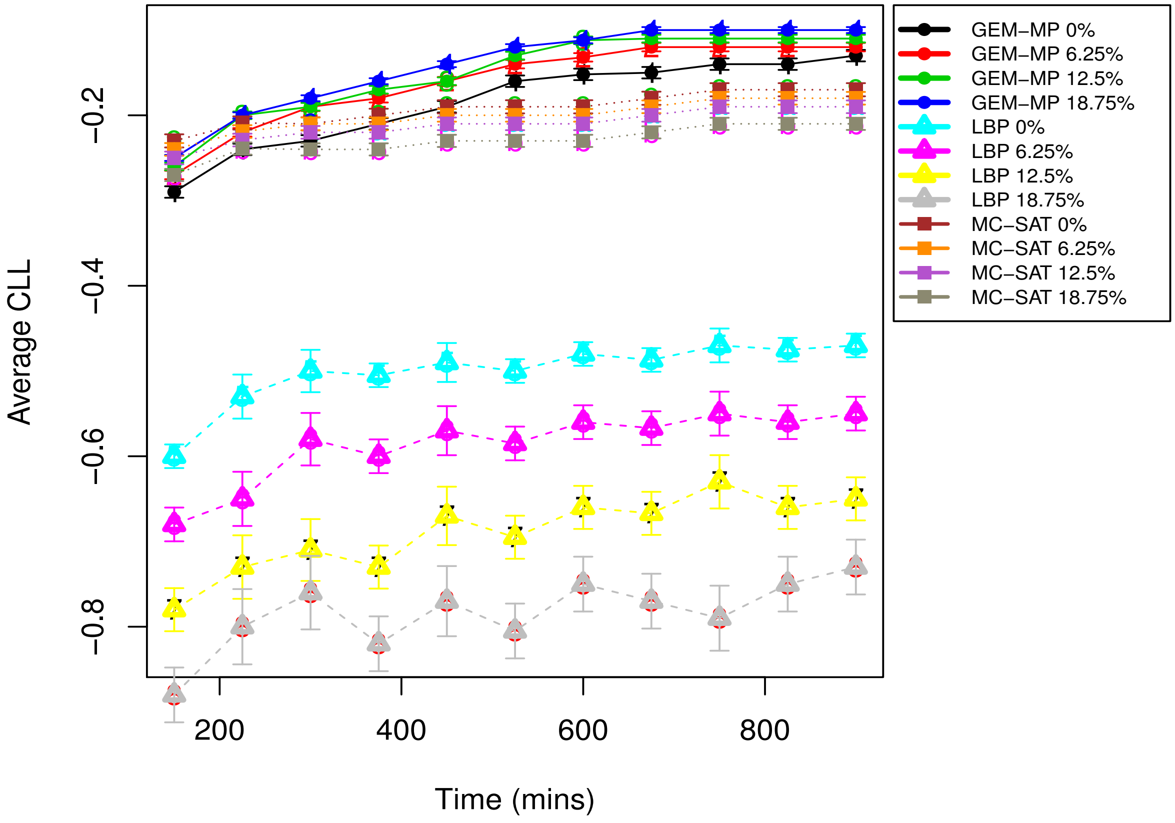

Figure 4.7 The impact of gradual zones of determinism on the accuracy of GEM-MP, MC-SAT and LBP algorithms for Cora dataset.

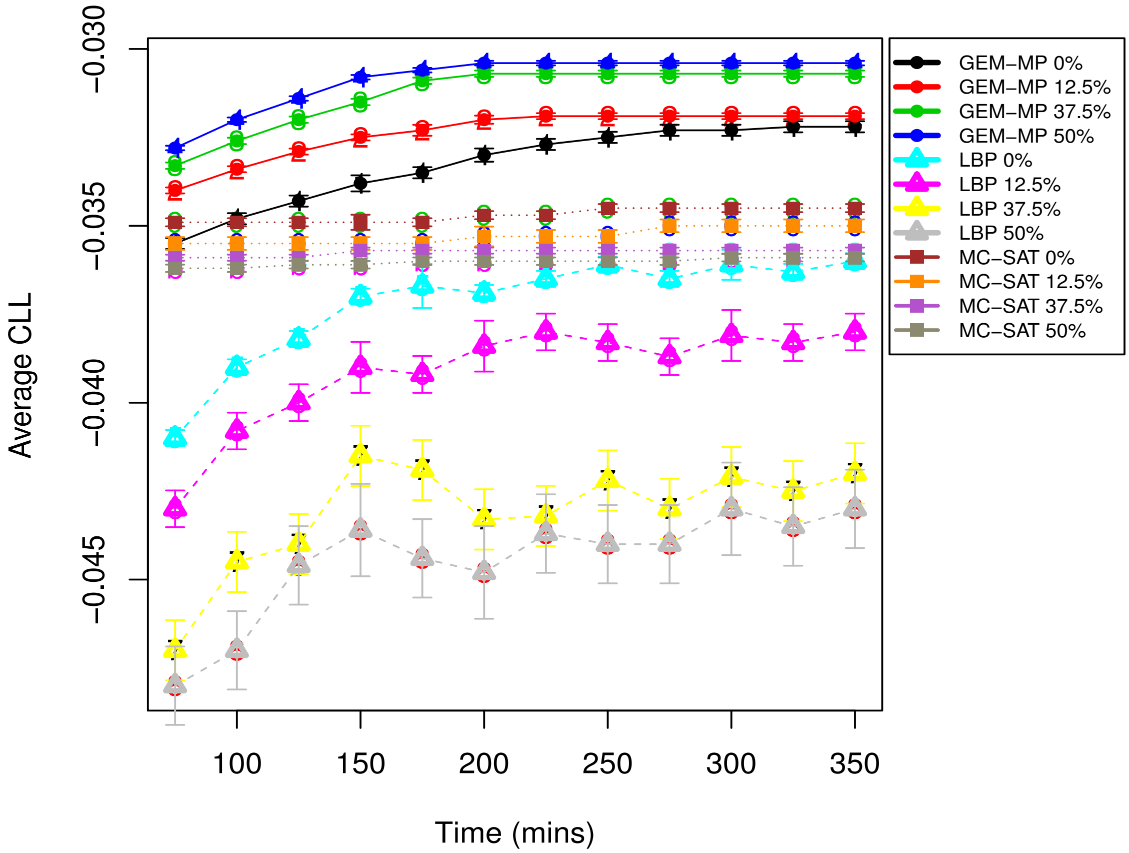

Figure 4.8 The impact of gradual zones of determinism on the accuracy of GEM-MP, MC-SAT and LBP algorithms for Yeast dataset.

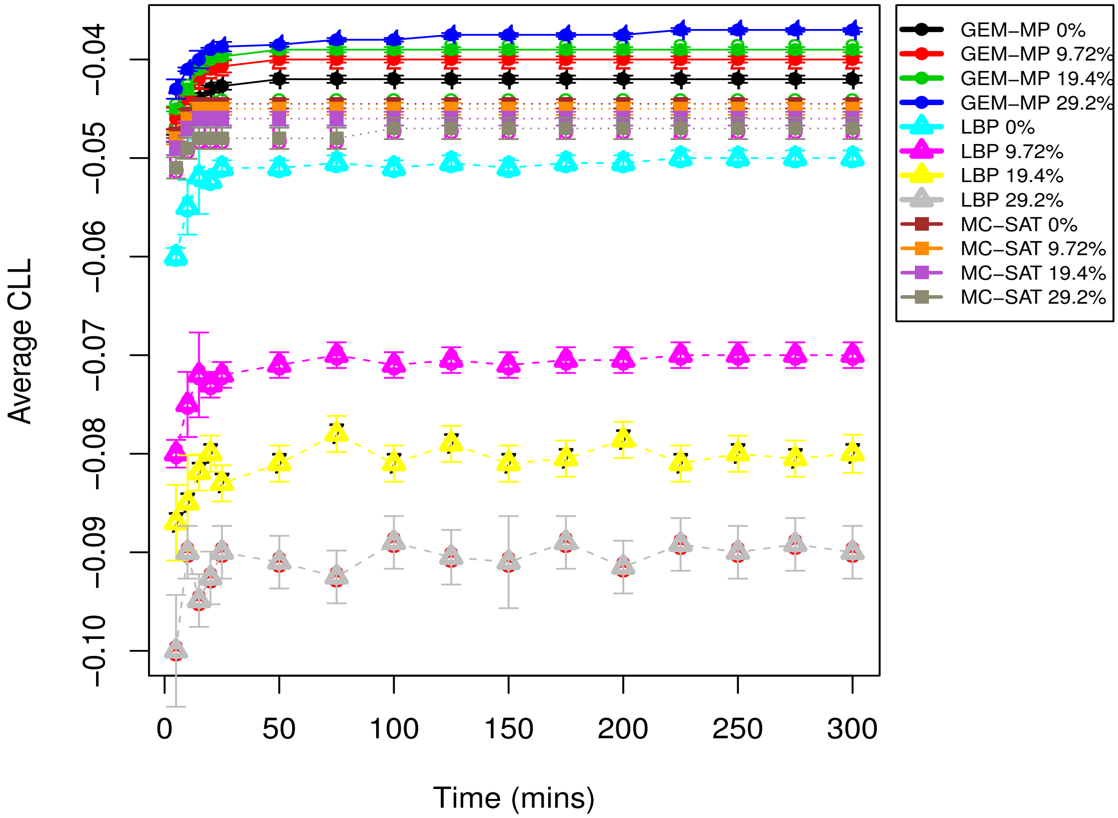

Figure 4.9 The impact of gradual zones of determinism on the accuracy of GEM-MP, MC-SAT and LBP algorithms for UW-CSE dataset.

accuracy was observed on all the tested datasets.

Experiment III

This experiment examines Q4. We are interested here in judging the scalability of various inference algorithms. To guarantee a fair comparison, we rerun experiment I, but we increased the number of objects in the domain from 100 to 200 in increments of 25, adhering to the methodology previously used by Poon *et al.* (2008); Shavlik and Natarajan (2009). Then we reported the average running time after converging (if the algorithm was able to converge) or allowing a maximum of 5000 and 10000 iterations respectively for the entire inference process.

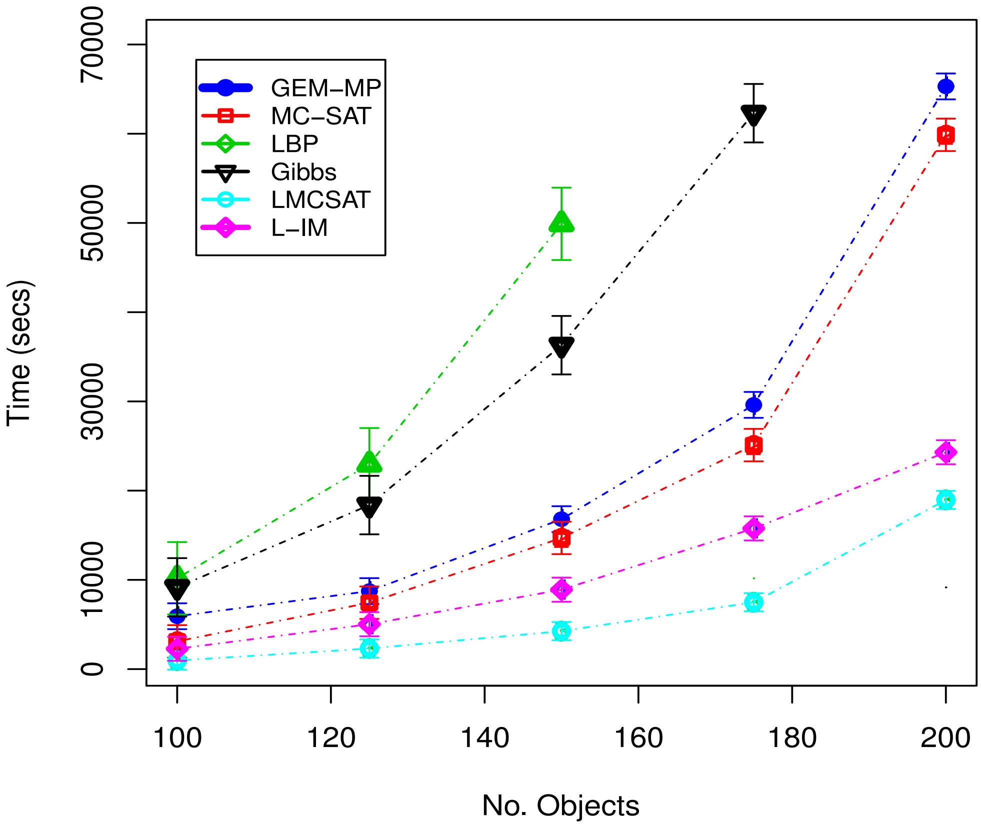

Figure 4.10 Inference time vs. number of objects in Cora.

Figures 4.10, 4.11 and 4.12 report the average inference time as a function of the varied number of objects in the domain. Overall, the results show that both LMCSAT (a lazy-based

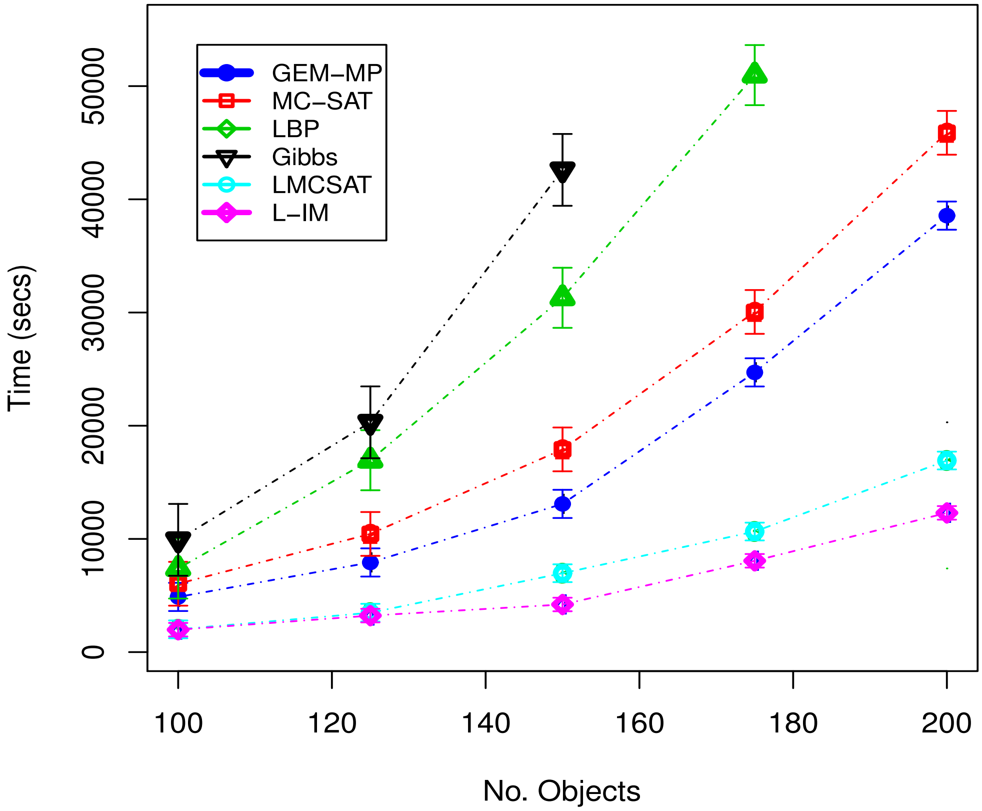

Figure 4.11 Inference time vs. number of objects in Yeast.

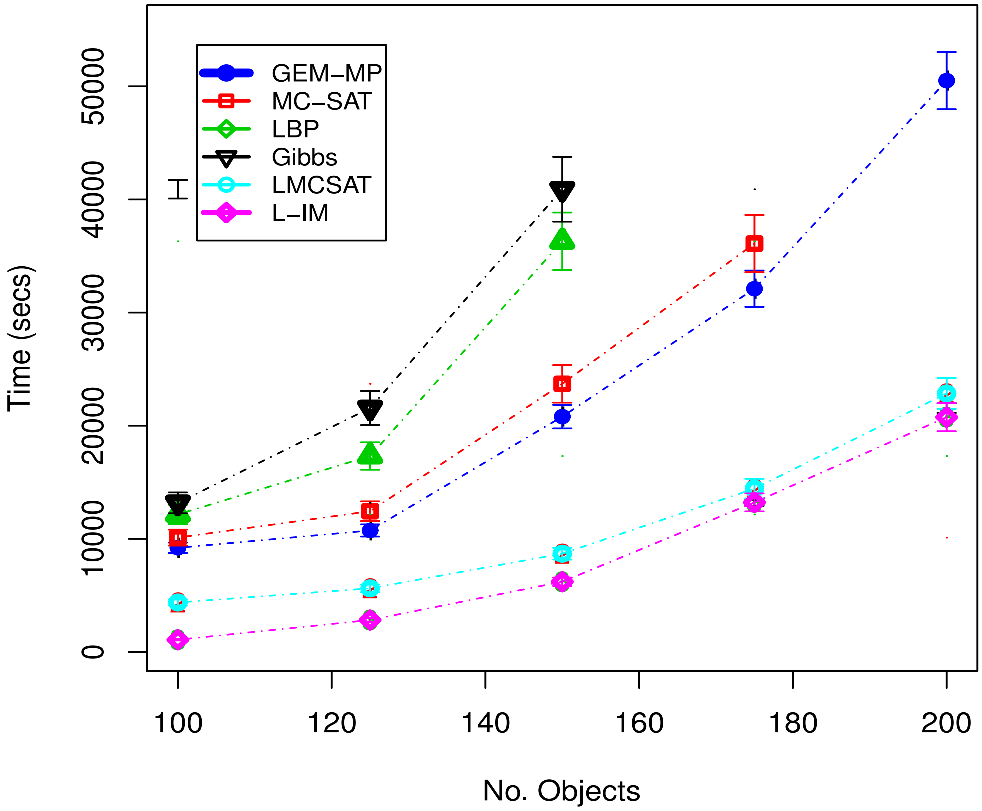

Figure 4.12 Inference time vs. number of objects in UW-CSE.

algorithm) and L-IM (lifted-based algorithm) rivaled each other, and both dominate all other compared propositional algorithms. 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 Yeast and UW-CSE datasets, whereas MC-SAT ran out of memory with 200 objects in UW-CSE dataset.

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 in 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 for evaluating message-passing algorithms, as it provides 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 in two determinism zones: Zone1 is $[0\% - 20\%]$ and Zone2 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 with time allowed to run until convergence up to 500 iterations (Note that, in this experiment, the inference algorithms are assigned only 500 iterations since the tested (Ising grid) datasets are translated into graphical models (defined over ≈ 400 variables) which are much smaller than those graph models representing Cora, Yeast and UW-CSE.). 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.

Figure 4.13 displays the cumulative percentage of convergence as a function of the number of iterations for each algorithm at the determinism Zone1 $[0\% - 20\%]$ and Zone2 $[20\% - 40\%]$. 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. In the first determinism zone, it finishes at 97% convergence rates versus 82% for L2-Convex, 68% for CCCP, and 59% for residual BP. In the second determinism zone, where the amount of determinism increases, it clearly achieves at least 17.5%, 34.8 %, and 48.4 % better convergence than L2-Convex, CCCP, and residual BP respectively.

Figure 4.14 displays the average KL-divergence (KL) between the approximate and exact node marginals for each algorithm as a function of the number of iterations for each algorithm in the two determinism zones considered. The results complement those of Figure 4.13, here again these results underscore 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, CCCP, and residual BP methods respectively. Also, it finishes at KL-divergence of 0.23 and 0.19 in the two determinism zones respectively. This ensures that the quality of marginals obtained by GEM-MP in the second determinism zone are more accurate than the ones obtained in the first determinism zone, 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.

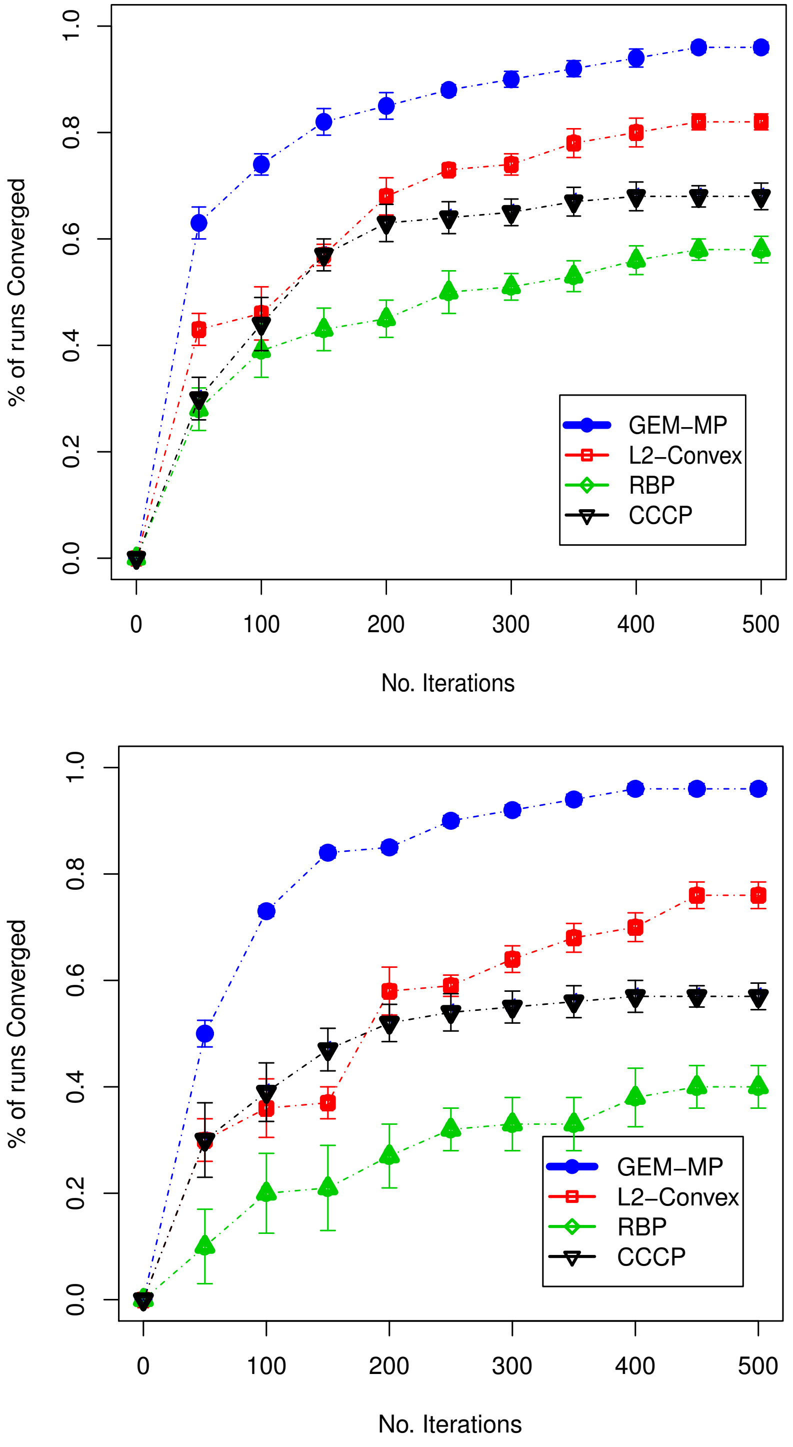

Figure 4.13 The results of 20 × 20 grids of Ising model: The cumulative percentage of convergence (Convergence %) vs. number of iterations at determinism Zone1 [0% − 20%] (*Top*) and at determinism Zone2 [20% − 40%] (*Bottom*).

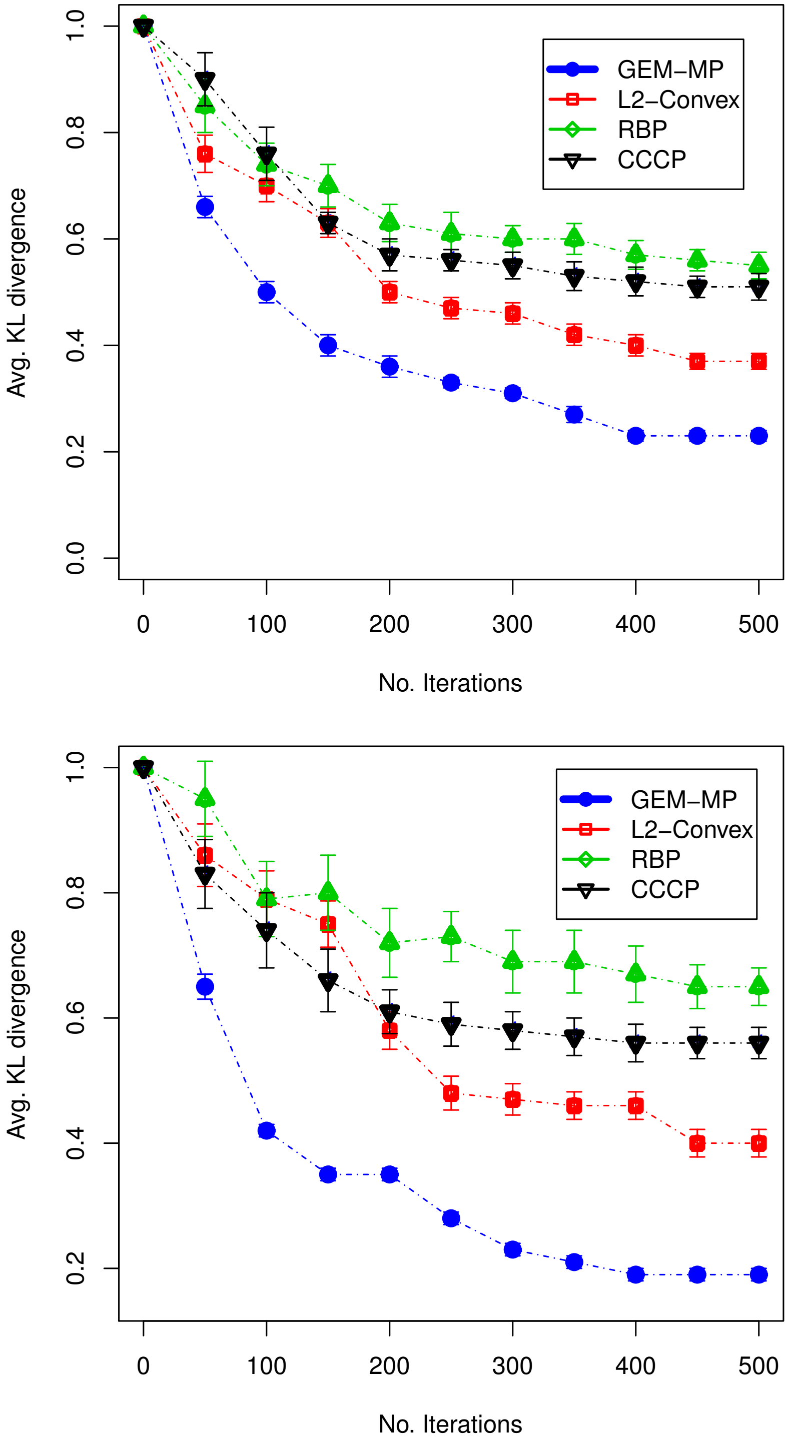

Figure 4.14 The results of 20 × 20 grids of Ising model: The average KL-divergence (KL) metric vs. number of iterations at determinism Zone1 $[0\% - 20\%]$ (*Top*) and at determinism Zone2 $[20\% - 40\%]$ (*Bottom*).

Experiment V

This experiment attempts to answer Q6. The goal is to compare the quality of solutions returned by GEM-MP at different initializations settings of marginals: GEM-MP with random initialization (GEM-MP-random) and GEM-MP with uniform initialization (GEM-MP-uniform). That is to say, 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.

Figure 4.15 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 zones 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.

4.7 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 local minimum. Second, it improves 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 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) algorithms. 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

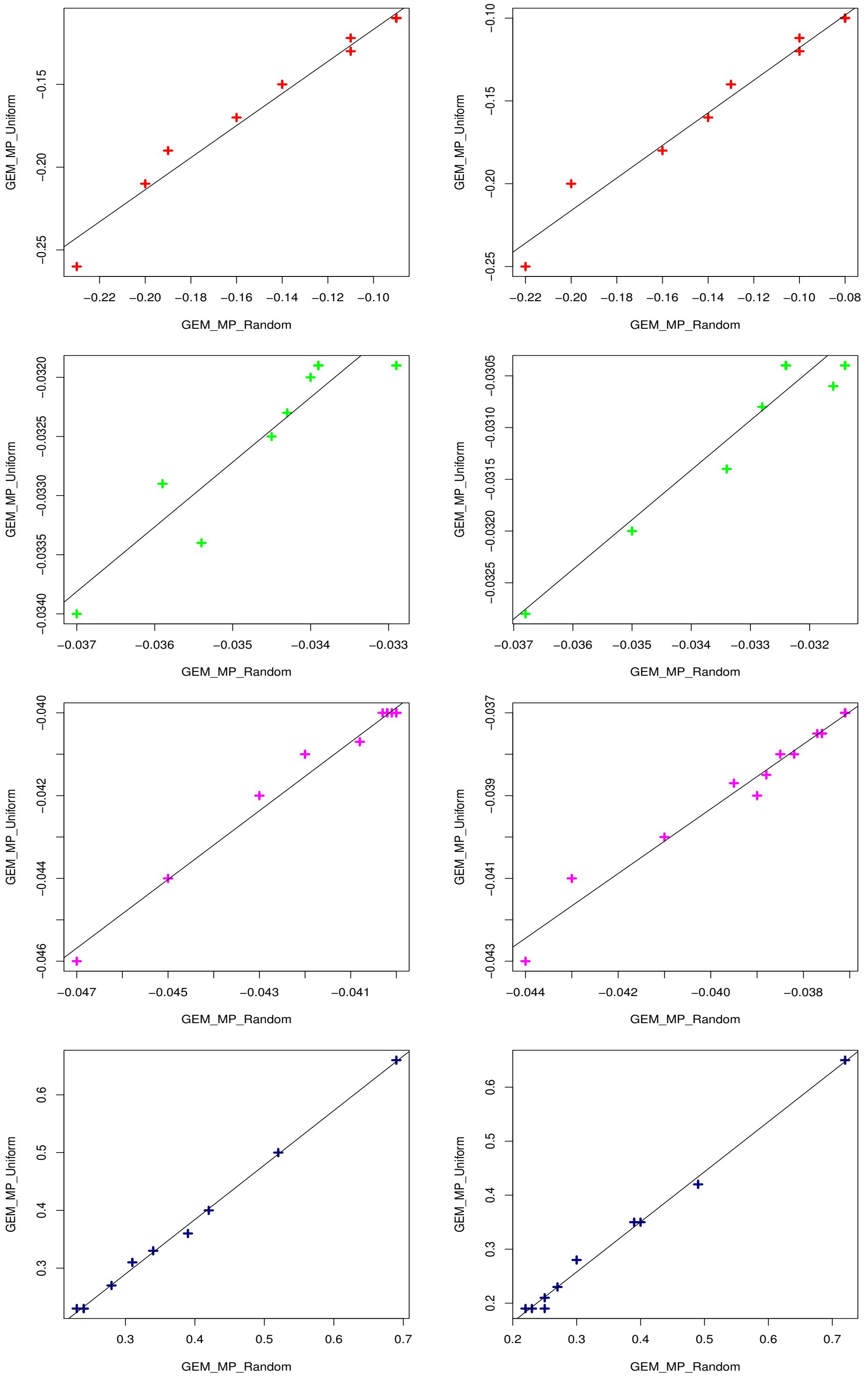

Figure 4.15 From Top to middle: The average CLL of GEM-MP-random (x-axis) vs. the average CLL of GEM-MP-Uniform (y-axis) for Cora (*red*), Yeast (*green*) and UW-CSE (*magenta*) at two determinism zones, respectively. Bottom: the average KL-divergence of GEM-MP-random vs. the average KL-divergence of GEM-MP-Uniform for 20×20 grids of Ising model at determinism zone1 $[0\% - 20\%]$ (*left-blue*) and at determinism zone2 $[20\% - 40\%]$ (*blue*) during iterations.

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 the 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 ages. 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, apparently, the result of determinism 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-Covex, 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. CCCP algorithm uses a convex-concave Bethe energy for achieving 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 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. However, generally speaking, it appears that GEM-MP is able to reach an accurate result given any initialization, but maybe with minor sacrificing in the allowed computational cost.

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 none of them 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 on 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.

CHAPTER 5 EXPLOITING DETERMINISM TO SCALE INFERENCE

Although determinism frequently poses a significant challenge to probabilistic inference (as discussed in Chapter 1), it remains an intrinsic part of the structure of SRL models. One approach (Allen and Darwiche, 2003) based on recursive conditioning has been proposed to exploit determinism in order to speed up exact inference by using standard logical techniques such as unit propagation. But the approach does not scale to large real-world problems. More recently, an efficient algorithm (Papai *et al.*, 2011) based on ideas from constraint programming has been introduced to use determinism in order to prune objects in the domain of atoms by enforcing generalized arc consistency on the set of hard constraints in the theory. For each hard constraint, the algorithm iteratively performs two main operations (join and project) and therefore its complexity (both space and time) is sensitive to the complexity of those operations that are in some cases exponential in the number of atoms.

In this chapter, our main objective is to exploit determinism to improve inference for large, real-world problems. That is, we propose Preference Relaxation (PR), a two-stage strategy that uses the determinism (i.e. hard constraints) present in the underlying model to improve the scalability of relational inference. Strictly speaking, PR takes advantage of the fact that hard constraints are sufficient to extract zero marginal probabilities for the considerable number of query ground atoms that violate them, and this without ever having to waste computational time (and memory usage) with preferences.

In what follows we begin by demonstrating our general PR approach in Section 5.1. Then we explain how PR applies to both MC-SAT and Belief Propagation algorithms in Section 5.2. Finally we present experimental evaluations in Section 5.4, followed by our discussion in Section 5.5.

5.1 Scaling Up Relational Inference via PR

In graphical models that feature determinism, it is wasteful to allocate memory to all constraints in advance when performing inference. The PR framework first allocates memory to the hard constraints along with their corresponding variables. In domains where most variables are more likely to violate hard constraints, this can save memory and also speed up the computation since we do not allocate memory and perform computations for soft constraints that do not affect the marginal probabilities of some query variables.

Definition 14. *Let $\mathcal{F}$ be a set of constraints. Each constraint $f_i(X_1, ...X_l) \in \mathcal{F}$, where the*

*X_j's are either variables or functions, has an inference state. The default state of constraints is **awake**, meaning that the constraint will be considered when constructing the underlying grounded network for answering the queries (i.e. for inference). Otherwise the constraint is **relaxed**, meaning that it will be ignored. A variable is **awake** if it appears directly as an argument of an awake constraint or in a function that is an argument of an awake constraint. Otherwise it is considered to be **relaxed**.*

5.1.1 The PR Framework

Assume that $\mathcal{A}$ is an inference algorithm that we want to transform via PR to obtain inference algorithm $\hat{\mathcal{A}}$. We make the following assumptions about $\mathcal{A}$ so that our framework can be used: (1) the theory of $\mathcal{A}$ involves carrying out a propositionalization step, then calling a propositional inference algorithm; (2) the propositional inference algorithm used by $\mathcal{A}$ updates one variable at a time.

We now describe how to derive $\hat{\mathcal{A}}$ from $\mathcal{A}$. Algorithm 4 takes as input an underlying model of constraints $\mathcal{M}$ (e.g. a Markov logic model that involves clauses with associated weights), query variables $\mathcal{X}$ whose marginals we want to compute, a pruning threshold γ, and an evidence database $\mathcal{DB}$ that involves variables we can condition. Its output are the marginal probabilities $\mathcal{B}$ of query variables $\mathcal{X}$. In the first stage we relax all soft constraints and allocate memory for the hard constraints that are not satisfied when all evidence variables are set to their fixed values. This consequently awakens the non-evidence variables that are involved in these hard constraints. The awake variables and awake hard constraints become the initial set of variables and constraints that are used to perform inference, computing marginal probabilities of awake query variables. Since many local-search-based inference algorithms (e.g., WalkSAT) as well as MCMC-based algorithms resort to randomization to choose the next variable to update, when relaxing the preferences we apply a smart randomization technique (Poon *et al.*, 2008) over awake variables only. After convergence, $\hat{\mathcal{A}}$ filters the set of awake query variables: those whose marginals are at most γ are removed from $\mathcal{X}$ and are added to evidence database $\mathcal{DB}$ as false evidence, and those whose marginals are at least $1-\gamma$ are added as true evidence.

In the second stage $\hat{\mathcal{A}}$ awakens all preferences that were previously relaxed. It then constructs the grounded network based on both the enlarged evidence database $\mathcal{DB}^*$ and the reduced set of query variables $\mathcal{X}^{**}$. That is to say it allocates memory for awake constraints (both hard and soft) that are not satisfied in the new evidence database $\mathcal{DB}^*$ to answer query variables $\mathcal{X}^{**}$. Then it applies the appropriate probabilistic inference to compute the marginal probabilities of awake query variables $\mathcal{X}^{**}$. It is worth noting that although the query variables are awakened at some point during inference, ultimately all of their marginal probabilities are guaranteed to be computed, but some are computed in the first stage (i.e., filtered query variables) and the rest will be computed in the second stage.

Algorithm 4 PR-based Inference algorithm $\hat{\mathcal{A}}$

```
Input: Evidence database ($\mathcal{DB}$), set of query variables ($\mathcal{X}$), underlying model ($\mathcal{M}$), pruning
       Threshold ($\gamma$).
Output: Marginals of query variables ($\mathcal{B}$).
// Preference Relaxation (PR) step
$\mathcal{F}^h \leftarrow$ awake hard constraints that are not satisfied by $\mathcal{DB}$;
$\mathcal{X}_h \leftarrow$ awake query variables in $\mathcal{X}$ that appear in $\mathcal{F}^h$;
$\mathcal{M}^h \leftarrow$ ConstructNetwork($\mathcal{X}_h, \mathcal{F}^h, \mathcal{DB}$);
$\mathcal{B}_{\mathcal{X}_h} \leftarrow$ Infer($\mathcal{X}_h, \mathcal{M}^h$);
// Filtering query variables $\mathcal{X}_h$
for each $X_j \in \mathcal{X}_h$ do
    if $\beta^+_{X_j} \leq \gamma$ then
        $\mathcal{DB} \leftarrow \mathcal{DB} \cup \{\neg X_j\}$ ;
        $\mathcal{B}_{\hat{\mathcal{X}}} \leftarrow \mathcal{B}_{\hat{\mathcal{X}}} \cup \left\{\beta^+_{X_j}\right\}$ ; $\mathcal{X} \leftarrow \mathcal{X} \setminus \{X_j\}$ ;
    else if $\beta^+_{X_j} \geq 1 - \gamma$ then
        $\mathcal{DB} \leftarrow \mathcal{DB} \cup \{X_j\}$ ;
        $\mathcal{B}_{\hat{\mathcal{X}}} \leftarrow \mathcal{B}_{\hat{\mathcal{X}}} \cup \left\{\beta^+_{X_j}\right\}$ ; $\mathcal{X} \leftarrow \mathcal{X} \setminus \{X_j\}$ ;
    end if
end for
$\mathcal{DB}^* \leftarrow \mathcal{DB}$; // Enlarge evidence database
$\mathcal{X}^* \leftarrow \mathcal{X}$; //Shrink query set
// Awake all relaxed preferences
$\mathcal{F}^* \leftarrow$ awake hard and soft constraints that are not satisfied by $\mathcal{DB}^*$;
$\mathcal{X}^{**} \leftarrow$ awake query variables in $\mathcal{X}^*$ that appear in $\mathcal{F}^*$;
//Construct the reduced ground network
$\mathcal{M}^* \leftarrow$ ConstructNetwork($\mathcal{X}^{**}, \mathcal{F}^*, \mathcal{DB}^*$);
$\mathcal{B}_{\mathcal{X}^{**}} \leftarrow$ Infer($\mathcal{X}^{**}, \mathcal{M}^*$);
$\mathcal{B} \leftarrow \mathcal{B}_{\mathcal{X}^{**}} \cup \mathcal{B}_{\hat{\mathcal{X}}}$;
Return $\mathcal{B}$;
```

The advantage of our PR strategy is three-fold. First we avoid unnecessary computations with soft constraints to obtain marginal probabilities of awake query variables that violate awake hard constraints. Second we reduce the set of query variables $\mathcal{X}$ by filtering out some awake query variables that violate hard constraints. Third we enlarge the evidence database by adding those filtered query variables, which reduces the effective size of the grounded network that will be constructed for inference in the second stage since the evidence variables are fixed to their truth values.

5.2 PR-based Relational Inference Algorithms

In relational domains the variables are ground atoms which are defined over Boolean domains $\mathcal{D} = \{+, -\}$. A ground atom is an evidence if its marginal $\beta^+ = 0$ (false evidence) or $\beta^- = 0$ (true evidence). The constraints are ground clauses. The ground clause is *relaxed* if it is set to a non-default state and otherwise it is *awake*. A ground atom is awake if it is included by (at least) one awake ground clause, otherwise it is relaxed.

PR-based inference algorithm $\hat{\mathcal{A}}$ is general and can be combined with many relational inference algorithms such as MCMC sampling (e.g. Gibbs sampling, simulated tempering, etc.), MC-SAT, Walk-SAT, and Belief propagation (BP). Here we illustrate PR-based relational inference for Belief propagation and MC-SAT, and use the latter in our experiments.

5.2.1 PR-BP

Given a factor graph $\mathcal{G}$, PR-BP performs three steps. The situation is depicted in Fig. 5.1(b).

- **Step (i) - Relaxing the Factor Graph:** We relax soft factors and construct the factor graph for factor nodes and variable nodes that represent the awake ground hard clauses and their awake ground atoms, respectively. Assuming there is no evidence, all awake variable nodes are queries. We simulate BP inference by alternating the passing of awake messages.[1] The message from an awake variable X_j to an awake factor f_i is:

$$\mu^{\text{awake}}_{X_j \to f_i} = \prod_{f_k \in \mathcal{F}_{X_j} \setminus \{f_i\}} \mu^{\text{awake}}_{f_k \to X_j} \tag{5.1}$$

  and the message from an awake factor f_i to awake variable X_j is:

$$\mu^{\text{awake}}_{f_i \to X_j} = \sum_{\mathcal{X}_{f_i} \setminus \{X_j\}} \left(f_i^{\text{awake}} \prod_{X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}} \mu^{\text{awake}}_{X_k \to f_i} \right) \tag{5.2}$$

- **Step (ii) - Enlarging the Evidence Database**: We keep track of which variables and factors send and receive *violating awake messages*: an awake message is violating if it is less than γ or greater that $1 - \gamma$. The weight of an edge is the number of times the variable node receives an identical violating message from the same factor node. If

[1] The message is awake if it is passed between the awake variable node and the awake factor node, or vice versa.

we obtain a specified weight of such a message for a variable, then this means that it violates the hard factor. Thus we filter the variable and relax its messages.

For instance at Fig. 5.1(b)-ii, if variable C violates a factor we mark it as evidence (i.e. black circle) by adding it to $\mathcal{DB}$ and relax its passing of messages with this factor (i.e. dashed line). In addition if a factor is satisfied by marking one of its argument variable nodes as evidence then we relax this factor (i.e. black square) and relax its messages as well.

- **Step(iii) - Constructing the reduced Factor Graph:** We awake soft factors and construct the reduced grounded factor graph based on the last information we obtained from step (2). We then run the standard BP algorithm on it.

5.2.2 PR-MC-SAT

PR-MC-SAT initializes by relaxing all soft clauses and awakens only those hard clauses that are not satisfied by the given evidence database $\mathcal{DB}$. It then calls Walk-SAT to find a solution to all awake hard clauses (which is a satisfying assignment to their awake ground atoms). At each iteration it generates membership M by sampling from the currently satisfied awake hard clauses. Note that since the candidate clauses in M are all hard, we do not select them based on their weights (all uniformly infinite) but whenever their auxiliary variable is greater than 1. We then run unit propagation in order to simplify the selected clauses in M. PR-MC-SAT next calls SampleSAT (Wei *et al.*, 2004) to obtain the next state over awake atoms by sampling near-uniformly from their solutions in M, which is initialized using smart randomization (Poon *et al.*, 2008) over the awake atoms. Once the *num_samples* threshold is reached we identify awake atoms whose marginals are less than γ or greater than $1 - \gamma$. These atoms are removed from the query set and added to the evidence database. We then awake the soft clauses and construct a grounded Markov network based on the shrunken query list and enlarged evidence database, on which we perform standard MC-SAT inference.

5.3 Combining PR with Lazy Inference

One key advantage of PR is that it can be combined with other state-of-the-art approaches which improve the scalability of inference, such as Lazy and Lifted. Algorithm 5 shows how to combine PR with Lazy MC-SAT (Poon *et al.*, 2008): it only differs from PR with propositional MC-SAT at Lines 2, 5, 6, and 8. Lazy-PR starts by calling Lazy-SAT to find the solution to awake hard constraints and then runs unit propagation among *active-awake* hard clauses in M and their atoms. It then calls Lazy-SampleSAT to obtain the next state

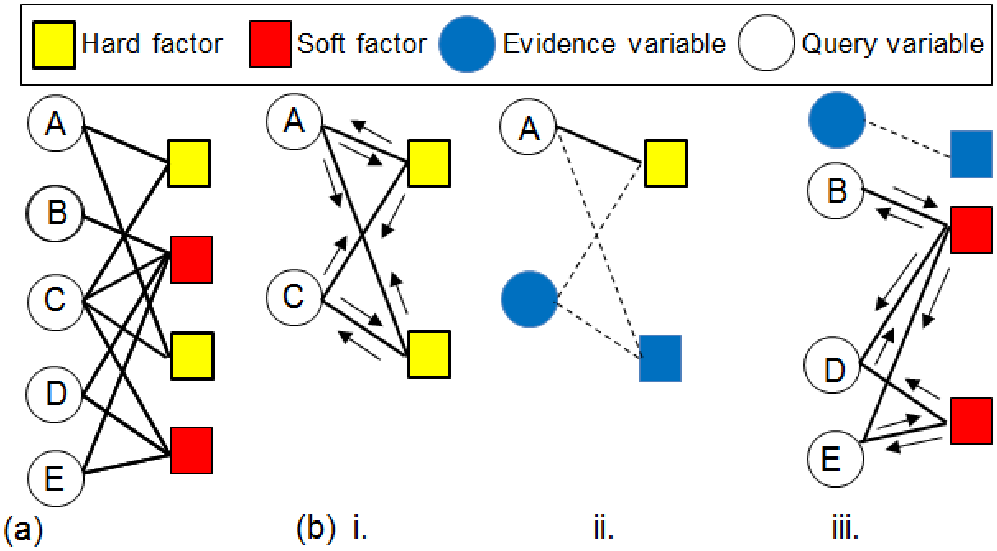

Figure 5.1 a) Propositional-BP. b) From left to right, the steps of PR-BP inference algorithm.

by sampling near-uniformly from the solutions in M. After reaching the *num_samples* threshold, it filters *active-awake* query atoms and enlarges the evidence database.

Algorithm 5 Combining PR with Lazy MC-SAT.

```
Relax soft clauses and maintain only active
hard clauses that are awakened;
 // call Lazy-SAT
$\mathcal{X}_h^{(0)} \leftarrow$ Satisfy (active-awake hard clauses);
for $i \leftarrow 1$ to $num\_samples$ do
    $M \leftarrow \emptyset$;
    for all $c_k \in$ active-awake hard clauses satisfied by $\mathcal{X}_h^{(i-1)}$ do
        add $c_k$ to $M$;
    end for
    Sample $\mathcal{X}_h^{(i-1)} \sim \mathcal{U}_{\mathrm{SAT(M)}}$;  //call Lazy SampleSAT
end for
$[\mathcal{DB}^*, \mathcal{X}^*] \leftarrow$ Filtering the active-awake query atoms $\mathcal{B}_{\mathcal{X}_h}$;
Construct the grounded network lazily on $\mathcal{DB}^*$ and $\mathcal{X}^*$;
call MC-SAT;
```

5.4 Experimental Evaluations

The goal of our experimental evaluation is to investigate the following key questions.

- (**Q1**) Is PR powerful enough to reduce significantly the size of grounded networks?
- (**Q2**) Is PR-based inference competitive with prominent state-of-the-art methods such as Lazy Inference?
- (**Q3**) How is the scalability of PR-based inference influenced by the amount of determinism present in the underlying model and the pruning threshold that is used in filtering the query variables?

We experimented on a protein interaction task in a molecular biology domain, a link prediction task in a social networking domain, and an entity resolution task in a citation matching domain. In these problem domains it is possible to transform certain soft constraints into hard constraints. This provides us with the ability to test PR at differing levels of hard constraints.

5.4.1 Metrics

To assess PR, we compared the scalability and accuracy (with and without PR) for both propositional MC-SAT (Poon and Domingos, 2006) and its lazy inference version (Poon *et al.*, 2008), as implemented in the Alchemy system (Kok *et al.*, 2007). We use the following metrics:

- Space consumption
- Running time of network construction and inference.
- Average conditional log-likelihood (CLL).

5.4.2 Methodology

We conducted the experimental evaluations in the following manner. In the training phase, we learned the weights using a preconditioned scaled conjugate gradient (PSCG) algorithm (Lowd and Domingos, 2007) by performing a four-way cross-validation for the protein interaction task, and a five-way cross-validation for both the link prediction and entity resolution tasks. In the testing phase, we carried out inference on the held-out dataset to produce the marginals of all groundings of query atoms being true by using the following six underlying inference algorithms:

- **MC-SAT**

- **Lazy-MC-SAT**
- **PR-MC-SAT 0.001**
- **PR-MC-SAT 0.01**
- **PR-Lazy-MC-SAT 0.001**
- **PR-Lazy-MC-SAT 0.01**

We ran each algorithm until it either converged or its number of iterations exceeded 10 000. To obtain robust answers to the proposed questions, we did the following:

- vary the number of objects in the domains, following methodology previously used (Poon *et al.*, 2008);
- use two pruning thresholds 0.001 and 0.01 for PR;
- vary the amounts of determinism in the models (i.e., starting with hard constraints that initially exist in MLN, we gradually increase the number of hard constraints and re-run the experiment).

All of the experiments were run on a cluster of nodes with 3.0 GHz Intel CPUs, 3 GB of RAM, RED HAT Linux 5.5, and we implemented PR as an extension to the Alchemy software (Kok *et al.*, 2007).

5.4.3 Datasets

Protein Interaction

Here we use the MLN model of Davis and Domingos (2009) for a Yeast protein interaction problem. This dataset has been already described in detail in Chapter 4. The goal of inference is to predict the interaction relation (*Interaction, and Function*). Here, we ran the experiment at two amounts of determinism: 25% (i.e., ratio: 2 constraints out of 8 constraints, initially present as hard in MLN) and 37.5% (ratio: 3 constraints out of 8 constraints were considered hard), and we varied the number of objects in the domain from 0 to 1000 by increments of 50.

Link Prediction

For the link prediction task, we used the MLN model available on the Alchemy website (excluding the 22 unit clauses) of the UW-CSE dataset from (Richardson and Domingos, 2006). This dataset has been previously described in detail in Chapter 4. The inference task is to predict advisory relationships (*AdvisedBy*), and all other atoms are evidence (corresponding to all the information scenario (Richardson and Domingos, 2006)). Here we ran the experiment at two amounts of determinism: $\approx 9.7\%$ (ratio: 7 out of 72 constraints were hard) and $\approx 38.9\%$ (ratio: 28 out of 72 constraints were considered hard). In addition, we varied the number of objects in the domain from 0 to 400 by increments of 50.

Entity Resolution

For the entity resolution experiment, we used the MLN model that is similar to the established one of Singla (Singla and Domingos, 2006a) on the Cora dataset from Poon *et al.* (2008), which has been described in detail in Chapter 4. The goal of inference is to predict which pairs of citations refer to the same citation, author, title and venue (i.e, *SameBib, SameTitle, SameAuthor and SameVenue*). The other atoms are considered evidence atoms. We ran the experiment at two amounts of determinism: $\approx 12.5\%$ (ratio: 4 out of 32 constraints, actually already appearing in MLN as hard) and $\approx 25\%$ (ratio: 8 out of 32 constraints were considered hard). Additionally, we varied the number of objects in the domain from 0 to 500 by increments of 50.

5.4.4 Results

Figures 5.2–5.9 display the space and time for total inference (where total inference = construction time + inference time) as a function of the number of objects in the domain for the six underlying inference algorithms at different amounts of determinism. The results show that PR-MC-SAT at thresholds 0.001 and 0.01 finishes at least four orders of magnitude faster than the propositional MC-SAT on both Yeast and UW-CSE datasets, and three orders of magnitude on the Cora dataset. It was also very competitive with Lazy Inference on Yeast and on UW-CSE with 38.9% determinism. In addition, PR-lazy-MC-SAT at thresholds 0.001, 0.01 exceeds both the propositional MC-SAT and the lazy-MC-SAT on all underlying tested datasets. Clearly, PR-lazy-MC-SAT 0.01 was able to handle all full datasets, whereas lazy-MC-SAT ran out of memory with 1000 objects in the Yeast dataset.

Table 5.1 summarizes the average Construction (with and without PR) and inference times (mins.), memory (MB) and accuracy (CLL) metrics of Propositional grounding and PR-based

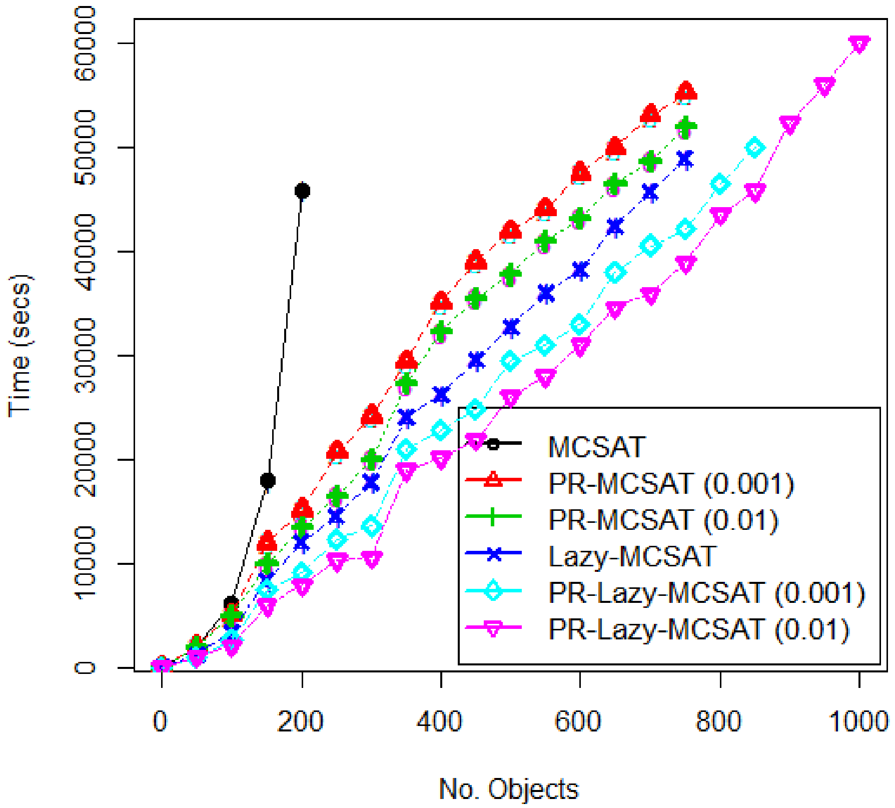

Figure 5.2 Inference time (secs) vs. number of objects in Yeast at 25% amount of determinism.

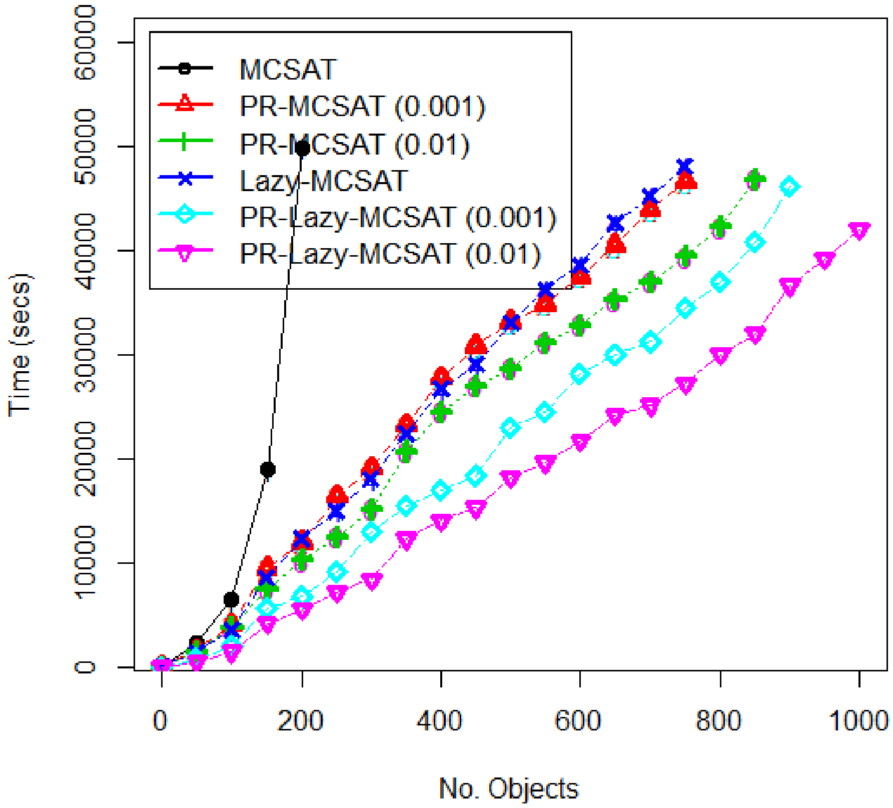

Figure 5.3 Inference time (secs) vs. number of objects in Yeast at 37.5% amount of determinism.

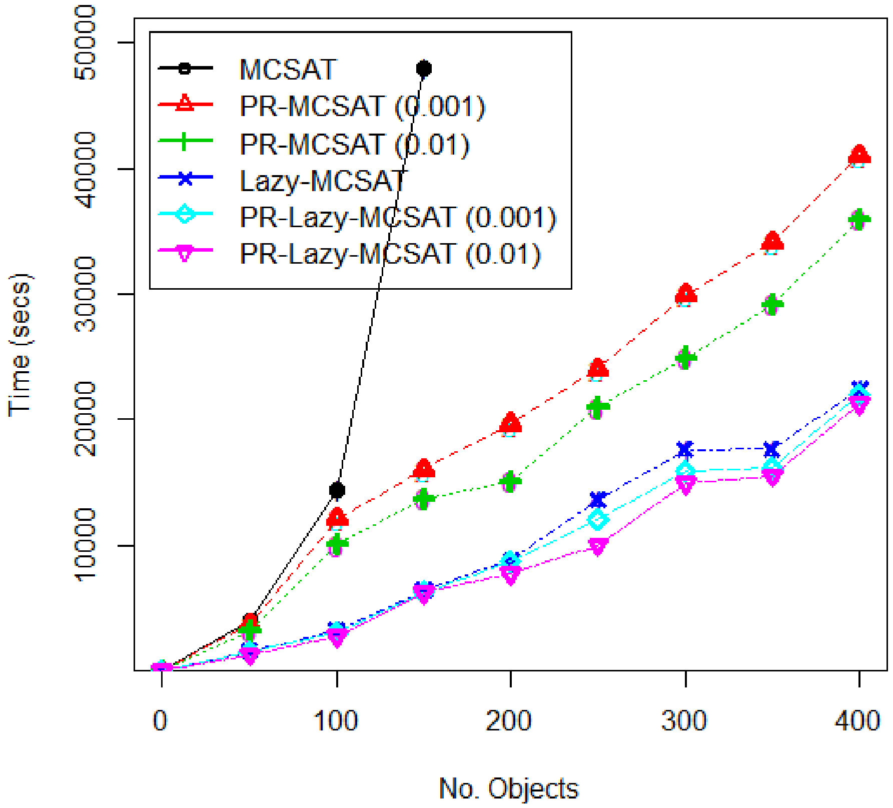

Figure 5.4 Inference time (secs) vs. number of objects in UW-CSE at 9.7% amount of determinism.

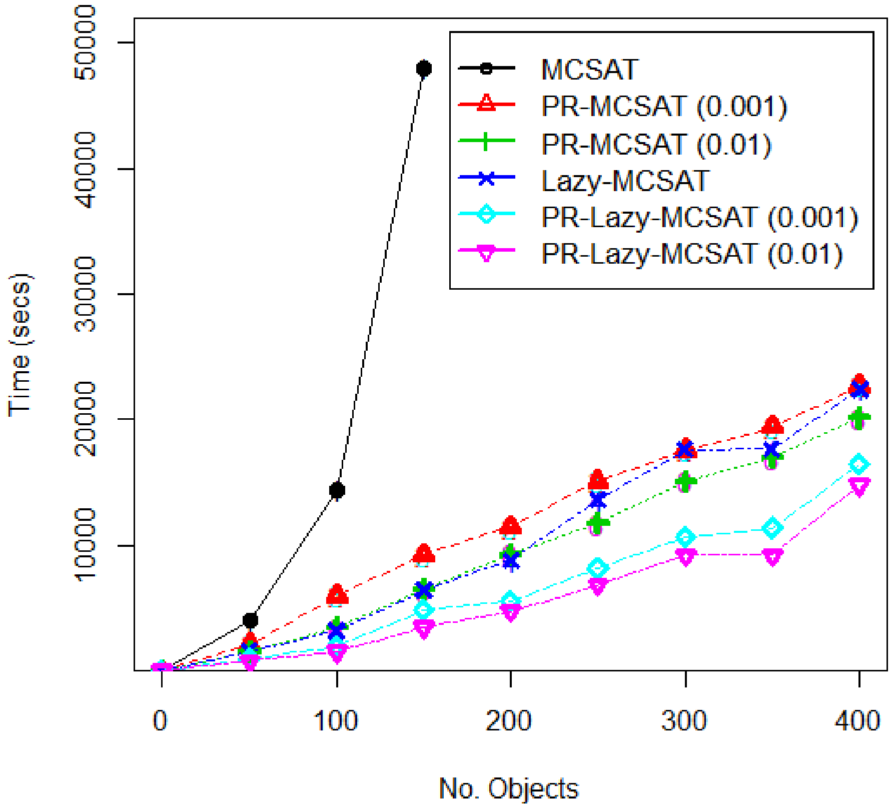

Figure 5.5 Inference time (secs) vs. number of objects in UW-CSE at 38.9% amount of determinism.

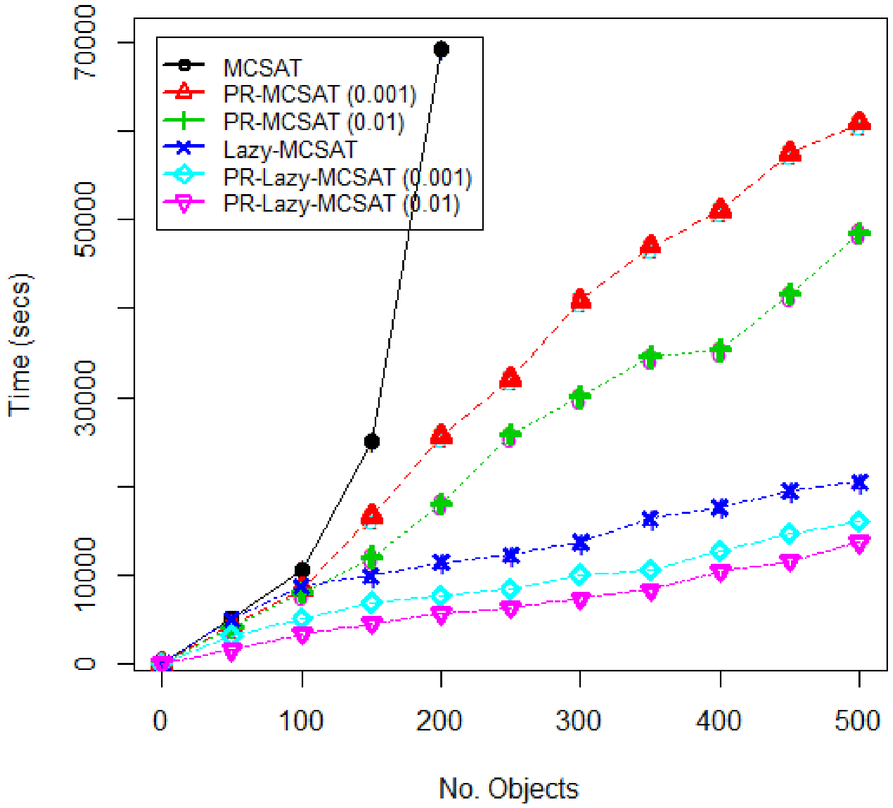

Figure 5.6 Inference time (secs) vs. number of objects in Cora at 12.5% amount of determinism.

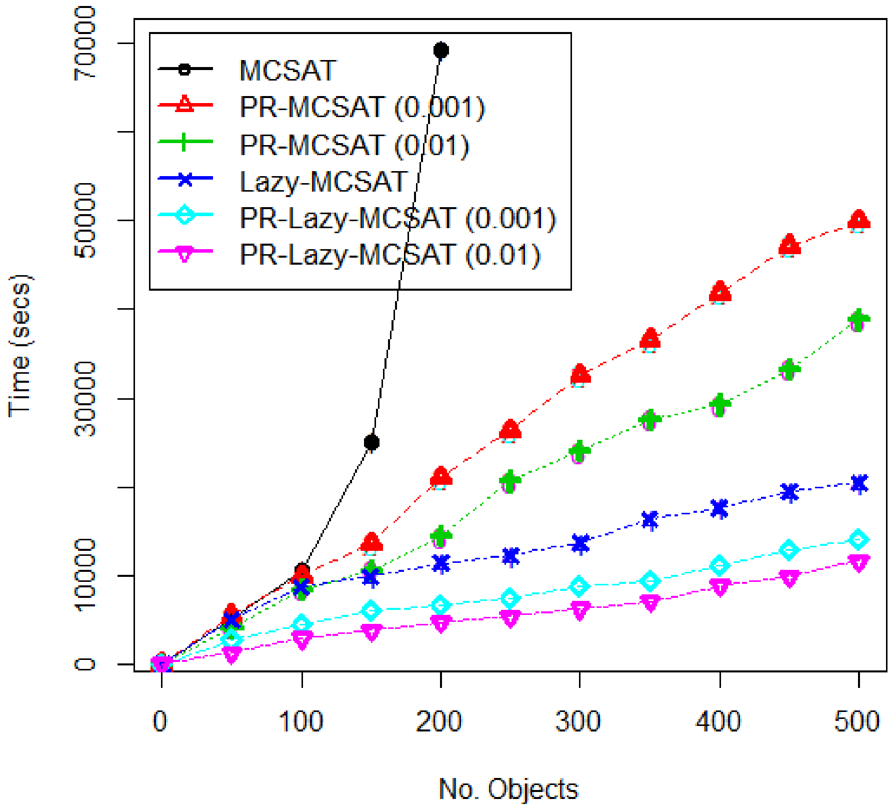

Figure 5.7 Inference time (secs) vs. number of objects in Cora at 25% amount of determinism.

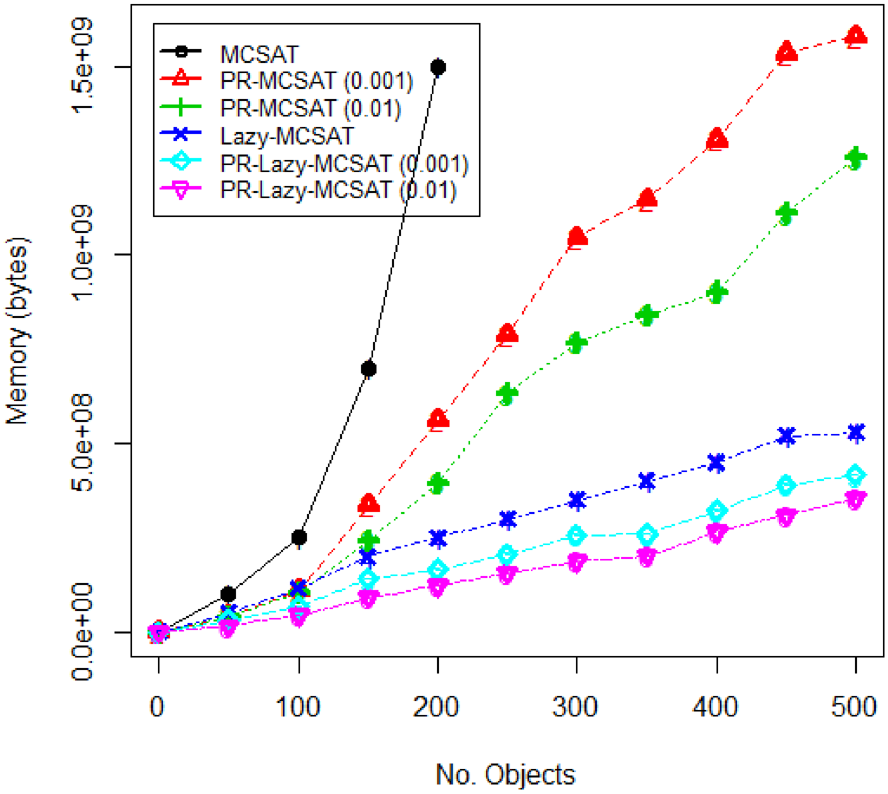

Figure 5.8 Inference memory space (bytes) vs. number of objects in Cora at 25% amount of determinism.

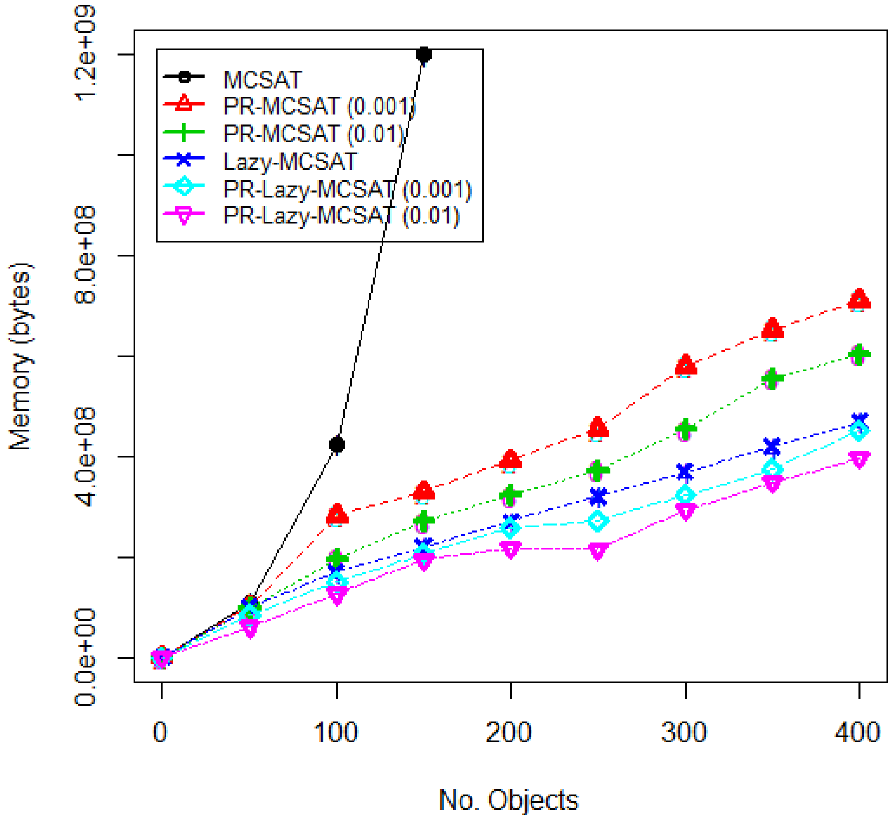

Figure 5.9 Inference memory space (bytes) vs. number of objects in UW-CSE at 25% amount of determinism.

MC-SAT inference algorithms on the underlying datasets. The results complement those of Figure 5.8, ensuring the promise of PR-based algorithms to improve MC-SAT's inference time and memory space. It also shows that RP-based algorithms at low thresholds (0.001) maintain approximately the same accuracy on the three underlying datasets, although they have a minor loss in accuracy to estimate marginals with a large pruning threshold (0.01), particularly in the Cora dataset.

5.5 Discussion

Overall the results clearly show that PR-based algorithms substantially improve the scalability of propositional MC-SAT inference. This is due to, first, avoiding irrelevant computations as well as memory usage involving soft ground clauses for the large number of query ground

Table 5.1 Average Construction (with and without PR) and Inference times (mins.), memory (MB) and accuracy (CLL) metrics of Propositional grounding and PR-based MC-SAT inference algorithms on the Yeast data set over 200 objects, the UW-CSE data set over 150 objects, and the Cora data set over 200 objects.

Datasets		Yeast		UW-CSE		Cora	
		Determinism %		Determinism %		Determinism %	
Algo.	Metric	25%	37.5%	9.7%	38.9%	12.5%	25%
Proposit.	*Cons.*	63.82	63.83	7.92	7.92	80.85	81.01
	Infer.	716.26	797.44	783.75	873.38	1074.15	1390.33
	Mem.	389	429	1144.41	1445.93	1430.51	1653.87
	CLL	−0.0353	−0.0357	−0.0433	−0.0457	−0.210	−0.245
PR 0.001	*PR-Cons.*	99.93	117.58	13.27	14.97	162.5	170.5
	Infer.	150.94	90.15	169.05	140.03	270.13	249.62
	Mem.	125.85	93.29	314.68	213.60	535.88	439.43
	CLL	−0.0354	−0.0362	−0.0443	−0.0467	−0.219	−0.261
PR 0.01	*PR-Cons.*	98.33	110.82	12.98	13.09	195.7	201.02
	Infer.	126.32	70.38	114.06	93.60	164.3	131.51
	Mem.	112.40	80.17	258.39	160.93	376.97	301.58
	CLL	−0.0365	−0.0370	−0.0459	−0.0470	−0.235	−0.280

atoms that violate hard ground clauses. Second, the enlarging of the evidence database that provides great implications for reducing the effective size of the grounded network.

PR-based algorithms were also very competitive with respect to Lazy Inference whenever a substantial amount of determinism is present in the model. This can be attributed to the fact that determinism offers a trade-off for the capacity of PR-based algorithms in terms of saving/wasting computational time and memory: on one hand the PR step costs both memory and time for inference on a large number of hard clauses, but on the other hand it shrinks the query set and enlarges the evidence database. Moreover, PR-based algorithms dominate both propositional MC-SAT and lazy-MC-SAT on all tested data sets when they were combined with Lazy Inference. This is because the result of such combination is the exploitation of both sparseness in Lazy and determinism in PR to scale up the inference.

CHAPTER 6 IMPROVING MAP INFERENCE USING CLUSTER BACKBONES

In this chapter, our main objective is to bring SP's success and SRL closer together for the benefit of MAP inference. That is, to remedy the aforementioned limitations of local search algorithms, which are due to search space clustering as explained in chapter 1, we introduce Weighted Survey Propagation-Inspired Decimation (WSP-Dec), a family of message-passing algorithms for applying MAP inference to SRL models and Markov logic in particular.

We have organized this chapter in the following manner. In Section 6.1, we demonstrate the family of WSP-χ algorithms. Then, in Section 6.2, we explain how WSP-χ can be applied to improve MAP inference in Markov logic. In Section 6.4, we conduct a thorough experimental study, which is followed by a discussion in Section 6.5.

6.1 WSP-χ Framework

6.1.1 Factor Graph Re-parameterization

To set up the framework of WSP-χ, the first thing we do is to generalize the natural interpretation of core assignments (see Definition 11) to be applicable to MAP inference in SRL models. That is, we introduce a new notion of max-core (denoted as "$\mathcal{W}$-core"):

Definition 15. *A max-core ($\mathcal{W}$-core) is a valid complete assignment $X \in \{0, 1, *\}^n$ such that:*

- *The total weight of its satisfying ground clauses equals $\mathcal{W}$*
- *It contains no unconstrained ground atoms equaling* 0 *or* 1
- *It satisfies all the hard ground clauses (if there are both hard and soft ground clauses involved in the model).*

Intuitively, the simple interpretation of a max-core is that it is a combinatorial object providing a representative generalization of MAP solutions within a cluster. Additionally, the core defined in Definition 11 is a particular case of the $\mathcal{W}$-core wherein $\mathcal{W}$ is the sum of all the ground clauses.

Now the second thing to be done in WSP-χ's framework is to modify the factor graph such that it defines a joint probability over complete assignments that are max-cores. Specifically,

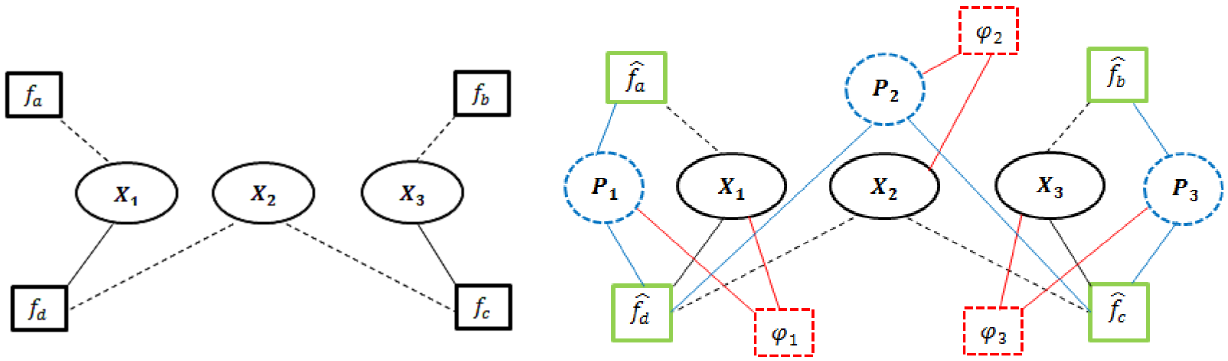

Figure 6.1 (*Left*) An example factor graph $\mathcal{G}$ that involves grounding clauses $\mathcal{F} = \{f_a, f_b, f_c, f_d\}$, and three ground atoms $\{X_1, X_2, X_3\}$, where dashed and solid lines represent "-" and "+" appearance of the atoms, respectively. (*Right*) The extended factor graph $\hat{\mathcal{G}}$, after adding auxiliary mega-node variables $\mathcal{P} = \{P_1, P_2, P_3\}$ and auxiliary factor nodes $\Phi = \{\varphi_1, \varphi_2, \varphi_3\}$, which yields a set of extended factors $\hat{\mathcal{F}} = \left\{\hat{f}_a, \hat{f}_b, \hat{f}_c, \hat{f}_d\right\}$.

we need to re-parameterize the factor graph in such a way that carrying out a BP on the new parameterization is equivalent to running a weighted variant of SP on the original factor graph for MAP inference. In Figure 6.1(*left*), we consider a simple example of a factor graph $\mathcal{G}$ that involves four ground clauses $\mathcal{F} = \{f_a, f_b, f_c, f_d\}$, and three ground atoms $\{X_1, X_2, X_3\}$. We re-parameterize this factor graph by transforming it into an *extended factor graph* $\hat{\mathcal{G}}$, shown in Figure 6.1(*right*), as follows:

- We extend the domain of each ground atom X_j to take values in $\{0, 1, *\}$.

- We add an *auxiliary mega-node* P_j (dashed circle) corresponding to each ground atom node X_j. Each of these mega-nodes P_j captures the parent set $P_j(X)$ of X_j. Although there could be several ways to define $P_j(X)$, we use a definition similar to the one used by Maneva et al. (2007)

$$P_j(X) = \{f \in \mathcal{F} | \text{CON}_{j,f}(X_f) = 1\} \tag{6.1}$$

  That is, $P_j(X)$ is the set of ground clauses for which X_j is the unique satisfying variable in X (i.e., the set of ground clauses constraining X_j to its value). $\mathcal{P} = \{P_j\}_{j=1}^{n}$ is the set of auxiliary mega-nodes in $\mathcal{G}$, with $n = 3$ in the example factor graph of Figure 6.1(*right*).

- Now, since we expand the arguments of each factor f_i by including auxiliary mega-node variables that correspond to their ground atoms $\mathcal{X}_{f_i}$, we then have an *extended factor*

$\hat{f}_i$. $\hat{\mathcal{F}} = \{\hat{f}_i\}_{i=1}^{m}$ is the set of extended factors in $\mathcal{G}$, with $m = 4$ in the example factor graph. Note that this extension implies that the complete assignments $\{X\}$ in the original factor graph $\mathcal{G}$ extended to corresponding configurations $\{\rho_X\}$ in the extended factor graph. Each configuration ρ_X takes the form $\rho_X = \{X_j, P_j(X)\}_{X_j \in \mathcal{X}}$ such that the projection $\pi_{\rho_X}(\mathcal{X})$ of ρ_X onto variables $\mathcal{X}$ produces a complete assignment X to $\mathcal{G}$, representing a candidate for a max-core.

Each $\hat{f}_i$ defines the following function for each configuration ρ_X

$$\hat{f}_i(\rho_X(\mathcal{X}_{\hat{f}_i})) = \xi(\pi_{\rho_X}(\mathcal{X}_{f_i})) \times \prod_{X_k \in \mathcal{X}_{f_i}} \delta([f_i \in P_k(X)], \mathrm{CON}_{k,f_i}(\mathcal{X}_{f_i})) \tag{6.2}$$

where ξ is a reward function that assigns different values to the projection $\pi_{\rho_X}(\mathcal{X}_{f_i})$ of X onto the original factor f_i as follows:

$$\xi(\pi_{\rho_X}(\mathcal{X}_{f_i})) = \begin{cases} e^{\hat{w}_i \cdot y} & \text{If } \pi_{\rho_X}(\mathcal{X}_{f_i}) \text{ satisfies } f_i, \\ \hat{\nu}_i & \text{If } \pi_{\rho_X}(\mathcal{X}_{f_i}) \text{ violates } f_i, \\ 0 & \text{If } \pi_{\rho_X}(\mathcal{X}_{f_i}) \text{ is invalid for } f_i. \end{cases} \tag{6.3}$$

where $\hat{w}_i$ is the weight associated with original factor f_i, y is a cooling parameter that plays a similar role to the y in SP-y algorithm (Battaglia *et al.*, 2004), and $\hat{\nu}_i$ is a violation value that equals 1, if f_j is a soft ground clause, and 0, if it is a hard ground clause. Therefore the role of ξ function is to provide a reward $e^{\hat{w}_i \cdot y}$ into the joint probability of a valid max-core X if it satisfies f_i, and to penalize it by 0 if X violates f_j, which is a hard ground clause. The second term in Eq. 6.2 involves a multiplication of the Kronecker delta function δ:

$$\delta(\overbrace{[f_i \in P_k(X)]}^{\kappa_1}, \overbrace{\mathrm{CON}_{k,f_i}(\mathcal{X}_{f_i})}^{\kappa_2}) = \begin{cases} 1 & \kappa_1 = \kappa_2 \\ 0 & \kappa_1 \neq \kappa_2 \end{cases} \tag{6.4}$$

which represents a constraint that enforces the consistency between the parent set $P_k(X)$ and the set of ground clauses constraining the ground atom X_k to its value.

- In addition, we attach an *auxiliary factor node* φ_j (dashed square) that connects each ground atom node X_j with its corresponding auxiliary mega-node P_j. Each φ_j defines the following function for any configuration ρ_X:

$$\varphi_j(\rho_X(X_j, P_j)) = \begin{cases} \gamma & \text{If } P_j(X) = \emptyset, \text{ and } X_j \neq *. \\ \chi & \text{If } P_j(X) = \emptyset, \text{ and } X_j = *, \\ 1 & \text{for other valid } (X_j, P_j), \end{cases} \tag{6.5}$$

where χ and γ are smoothing parameters restricted with a condition $\gamma + \chi = 1$. In the first case, the function φ_j assigns a value γ to a complete assignment X that is an invalid max-core (since the projection of ρ_X onto $\mathcal{X}$ has an unconstrained variable X_j set to 1 or 0, it represents an invalid max-core). In the second case, if the complete assignment X represents a valid max-core then it is assigned a positive value χ. In the third case the validity of (X_j, P_j) means that if $X_j = 1$ (resp. $X_j = 0$), then $P_j(X) \subseteq \mathcal{F}_{X_j+}$ (resp. $P_j(X) \subseteq \mathcal{F}_{X_j-}$). $\Phi = \{\varphi_j\}_{j=1}^n$ is the set of auxiliary factors in $\mathcal{G}$, with $n = 3$ in the example factor graph of Figure 6.1(*right*).

- For example, Table 6.1 depicts the expansion of complete assignments $\{1, 0, 1\}$ and $\{1, 1, 1\}$ in $\mathcal{G}$ to their corresponding ρ_X configurations in $\hat{\mathcal{G}}$. As an illustration, consider the complete assignment $\{1, 0, 1\}$ in $\mathcal{G}$, this assignment has $X_1 = 1$ constrained by f_d, $X_2 = 0$ constrained by $\{f_c, f_d\}$, and $X_3 = 1$ constrained by f_c. The assignment $\{1, 1, 1\}$ implies the same constrained ground clauses for X_1 and X_3, but not for X_2 which can not be constrained to 1 by any ground clause. In the extended factor graph $\hat{\mathcal{G}}$, we note that flipping the X_2 value from 0 to 1 will not violate or satisfy any additional ground clauses. Thus, we have additionally the complete assignment $\{1, *, 1\}$. Now, each one of three assignments $\{1, 0, 1\}$, $\{1, 1, 1\}$ and $\{1, *, 1\}$ can be a max-core with a certain probability. However, both of the complete assignments $X = \{1, 0, 1\}$ and $X = \{1, 1, 1\}$ have identical parent sets $P_1 = \{\hat{f}_d\}$, $P_2 = \emptyset$, and $P_3 = \{\hat{f}_c\}$. Note that these two assignments yield an invalid max-core since they violate the second condition in the concept of max-core (see Definitions 6 and 15). Thus, their corresponding ρ_X configurations have joint probability multiplied by γ. On the contrary, $X = \{1, *, 1\}$ is a valid max-core and its corresponding ρ_X configurations have joint probability multiplied by χ.

Note that decreasing the value of γ, which is equivalent to increasing χ's value, implies increasing the probability that both $\{1, 0, 1\}$ and $\{1, 1, 1\}$ assignments can not be a max-core. It also increases the probability that $\{1, *, 1\}$ can be a max-core. (It will be shown in subsection 6.1.3 that the best parameterization to $\hat{\mathcal{G}}$ is to choose $\chi = 1$ and $\gamma = 0$.)

Now it should be noted that since the values of a complete assignment X of $\mathcal{X}$ determine

Table 6.1 (*Left*) The joint probabilities of complete assignments $\{1,0,1\}$ and $\{1,1,1\}$ in the original factor graph. (*Right*) The (solution cluster-based) joint probabilities of their corresponding configurations ρ_X in the extended factor graph, where $\hat{w}_c$ (resp. $\hat{w}_d$) are the weights associated with the factors f_c (resp. f_d) that are satisfied by the underlying complete assignments.

X_1	X_2	X_3	$p(X_1, X_2, X_3)$
1	0	1	$e^{(\hat{w}_c+\hat{w}_d)\cdot y}$
1	1	1	$e^{(\hat{w}_c+\hat{w}_d)\cdot y}$
.	.	.	.

X_1	X_2	X_3	P_1	P_2	P_3	$p(X_1, X_2, X_3, P_1, P_2, P_3)$
1	0	1	f_d	$\emptyset$	f_c	$\gamma \times e^{(\hat{w}_c+\hat{w}_d)\cdot y}$
1	1	1	f_d	$\emptyset$	f_c	$\gamma \times e^{(\hat{w}_c+\hat{w}_d)\cdot y}$
1	$*$	1	f_d	$\emptyset$	f_c	$\chi \times e^{(\hat{w}_c+\hat{w}_d)\cdot y}$
.	.	.	.	.	.	.

uniquely the values of $\{P_j(X)\}_{X_j \in \mathcal{X}}$ in ρ_X, then the distribution over ρ_X is a distribution over X. Hence the extended factor graph defines the joint probability over complete assignment X as follows:

$$p(X) = p(\pi_{\rho_X}(\mathcal{X})) = \prod_{\hat{f}_i \in \hat{\mathcal{F}}} \hat{f}_i(\rho_X(\mathcal{X}_{\hat{f}_i})) \prod_{X_j \in \mathcal{X}} \varphi_j(\pi_{\rho_X}(X_j, P_j)) \propto \gamma^{n_0} \chi^{n_*} \prod_{f_i \in S(X)} e^{\hat{w}_i . y} \tag{6.6}$$

where n_0 is the number of unconstrained ground atoms in X having the value 1 or 0, and n_* is the number of unconstrained ground atoms in X having joker value $*$. $S(X)$ represents the set of all ground clauses satisfied by X.

6.1.2 WSP-χ Message-Passing

The message passing process of WSP-χ proceeds by iteratively sending two types of messages on the extended factor graph $\hat{\mathcal{G}}$ until convergence. These messages are slightly different from simply running the standard BP algorithm, but their structures are categorized based on the value of the set of auxiliary mega-nodes $\mathcal{P}$ in $\hat{\mathcal{G}}$ as follows:

- Factor-to-variable message ($\eta_{\hat{f}_i \to X_j}$): each extended factor $\hat{f}_i$ sends to its ground atoms $X_j \in \mathcal{X}_{\hat{f}_i}$ a message which is a vector of four components:

$$\eta_{\hat{f}_i \to X_j} = \begin{cases} \eta^s_{\hat{f}_i \to X_j} & \text{if } X_j = s_{i,j}, P_j \subseteq \mathcal{F}^s_{\hat{f}_i}(j) \cup \{\hat{f}_i\} \\ \eta^u_{\hat{f}_i \to X_j} & \text{if } X_j = u_{i,j}, P_j \subseteq \mathcal{F}^u_{\hat{f}_i}(j) \\ \eta^*_{\hat{f}_i \to X_j} & \text{if } (X_j = *, P_j = \emptyset) \vee (X_j = s_{i,j}, P_j \subseteq \mathcal{F}^s_{\hat{f}_i}(j)) \\ 0 & \text{Otherwise.} \end{cases} \tag{6.7}$$

where $\eta^s_{\hat{f}_i \to X_j}$ represents the probability of the warning that X_j is constrained to be uniquely satisfying for $\hat{f}_i$, $\eta^u_{\hat{f}_i \to X_j}$ is the probability that X_j can violate $\hat{f}_i$, and $\eta^*_{\hat{f}_i \to X_j}$ represents the probability that $\hat{f}_i$ does not care about the value of X_j (i.e., X_j is either unconstrained by $\hat{f}_i$ or at least one other variable X_k in $\mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ satisfies $\hat{f}_i$ and X_j equals $*$).

- Variable-to-Factor message ($\mu_{X_j \to \hat{f}_i}$). Here each ground atom X_j sends to its extended factors $\hat{f}_i \in \mathcal{F}_{X_j}$ a message that consists of three components:

$$\mu_{X_j \to \hat{f}_i} = \begin{cases} \mu^s_{X_j \to \hat{f}_i} & \text{if } X_j \text{ is constrained to satisfy } \hat{f}_i, \\ \mu^u_{X_j \to \hat{f}_i} & \text{if } X_j \text{ is constrained to violate } \hat{f}_i, \\ \mu^*_{X_j \to \hat{f}_i} & \text{if } X_j \text{ is unconstrained to } \hat{f}_i \text{ or equals } * \end{cases} \tag{6.8}$$

where the components represent the probabilities that X_j warned by other extended factors $\mathcal{F}_{\hat{f}_i}(j)$ to satisfy, violate, and be unconstrained to $\hat{f}_i$, respectively.

6.1.3 A Family of Extended Factor Graphs

In a promising attempt at understanding the success of SP, it was suggested that the solutions of random formulas typically do not possess non-trivial cores (Maneva *et al.*, 2007) (the core assignment is non-trivial if it has at least one frozen variable). This implies that the variants of SP(ρ) are most effective for values of ρ close to and not necessarily equal to 1. That is, pure SP, denoted as SP(1), is not always the most effective method that one usually wants to use, and that other versions of BP could be preferable. This is because the near-core assignments which are the ones of maximum weight in this case, may correspond to quasi-solutions of the cavity equations (Montanari *et al.*, 2004). However, this explanation has been shortly dismissed by experiments that ensure that non-trivial cores simply do exist for large formulas (Kroc *et al.*, 2007). This means that the pure SP is surprisingly the most accurate at computing marginals over these cores despite the existence of many cycles in the formulas (Kroc *et al.*, 2007). Recently, it has been shown that the cores can represent singleton clusters (with very few variables taking the $*$ value). These cores are called *degenerate covers* (Chieu *et al.*, 2007). In addition, it has been proven that, in many structured weighted Max-SAT problems, cores are often degenerate — see Lemma (2) in (Chieu *et al.*, 2007).

All of the aforementioned observations motivate the studying of a full family of WSP-χ extended factor graphs at various values of the smoothing parameter (i.e., $0 \leq \chi \leq 1$). This can be helpful to investigate if the MAP solutions possess non-trivial max-cores or not. From the satisfiability perspective, we believe that this can be beneficial for more understanding about the combinatorial properties of the solution space of structured SAT problems in relational domains. It can also be used to study the satisfiability threshold of these problems, and classifying them according to connectivity of the solution space.

Based on that, if we now adjust the value of the smoothing parameter[1], $\chi \in [0,1]$, in the auxiliary factor nodes (see Eq. 6.5), we consequently have a family of extended factor graphs that can be categorized into three cases:

- **Case 1. set $\chi \neq 0$ and $\gamma \neq 0$:** we have a subset of extended factor graphs parameterized by $0 < \chi < 1$. That is, for each value of $\chi \in (0,1)$, we have an extended factor graph that defines a positive joint distribution over max-cores as defined in Eq. (6.6). Hence, running WSP-χ's message-passing (as explained in subsection 6.1.2) recovers a family of Weighted SP algorithms corresponding to the values of $\chi \in (0,1)$.

- **Case 2. set $\chi = 0$ and $\gamma = 1$:** In this case the max-cores with $n_* = 0$ are the only ones having a positive joint distribution as given in Eq. (6.6). This means that the ground atoms are not allowed to take a joker value $*$, but take only values 1 or 0. This implies that the extended factor graph $\hat{\mathcal{G}}$ is equivalent to the original factor graph $\mathcal{G}$ in such a way that renders running WSP-χ's message-passing on $\hat{\mathcal{G}}$ equivalent to loopy max-product BP on $\mathcal{G}$.

- **Case 3. set $\chi = 1$ and $\gamma = 0$:** Here each auxiliary factor node (as defined in Eq. 6.5) takes the form

$$\varphi_j(\rho_X(X_j, P_j)) = \begin{cases} 0 & \text{If } P_j(X) = \emptyset, \text{ and } X_j \neq *, \\ 1 & \text{If } P_j(X) = \emptyset, \text{ and } X_j = *, \\ 1 & \text{for other valid } (X_j, P_j) \end{cases} \tag{6.9}$$

  where in the first case the function φ_j assigns a value 0 only if X is an invalid max-core (from Definition 15, when the complete assignment has unconstrained ground atom set to 1 or 0 then it can not be a max-core). Otherwise it assigns a positive value 1 when X is a valid max-core. Therefore, the extended factor graph in this case has a joint distribution over max-core X, as defined in the following theorem 2.

Theorem 2. *The underlying joint distribution defined by the extended factor graph, $\hat{\mathcal{G}}$ with $\chi = 1$, is positive only over valid max-cores.*

Proof. see Appendix A □

[1] Note that since $\gamma + \chi = 1$ then $\gamma = 1 - \chi$, so we can specify the setting of the auxiliary factor node using only one parameter χ.

Consequently, performing WSP-χ's message-passing on $\hat{\mathcal{G}}$ with $\chi = 1$ produces a pure version of Weighted SP algorithm (WSP-1).

Theorem 3. *When $y \to \infty$, then WSP-*1 *estimates marginals corresponding to the stationary point of the Bethe free energy on a uniform distribution over max-cores.*

Proof. see Appendix A □

From the above formulations, we have a parameterized family of Weighted Survey Propagation algorithms (WSP-χ), ranging from a traditional max-product BP (WSP-0) to pure weighted SP (WSP-1). In Subsection 6.4.3, our experimental evaluation shows the success of pure WSP-1 on real-world problems for finding the most accurate MAP solutions. This is consistent with experimental results in (Kroc *et al.*, 2007; Chieu *et al.*, 2007), showing that non-trivial cores often do exist for large formulas of structured problems.

6.1.4 Derivation of WSP-χ's Update Equations

In this subsection we derive the update equations for WSP-χ's message passing. For simplicity, and without lose of generality, we consider the derivation of WSP-1 (a pure version of WSP-χ when setting $\chi = 1$ in $\hat{\mathcal{G}}$). Now with the messages expressed in Eqs. (6.7) and (6.8), we can update the WSP-1's message passing in the following way.

Variable-to-Factor Updates

Let us start the variable-to-factor updates by computing the update of the message $\mu^s_{X_j \to \hat{f}_i}$. This message represents the event that X_j is constrained by other extended factors to satisfy $\hat{f}_i$, and therefore, it is specified by the values of $X_j = s_{i,j}$ and its mega-node $P_j = Z^j \cup \{\hat{f}_i\}$. That is, it takes the form:

$$\mu^s_{X_j \to \hat{f}_i} = \left\{ \sum_{Z^j \subseteq \mathcal{F}^s_{\hat{f}_i}(j)} \eta_{\hat{f}_i \to X_j} \middle| X_j = s_{i,j}, P_j = Z^j \cup \{\hat{f}_i\} \right\} \tag{6.10}$$

where $P_j = Z^j \cup \{\hat{f}_i\}$ is a notation representing the following event for a ground atom X_j

$$\hat{f}_i \in P_j \text{ and } Z^j = P_j \setminus \{\hat{f}_i\} \subseteq \mathcal{F}^s_{\hat{f}_i}(j) \tag{6.11}$$

Now we can compute Eq. (6.10) as

$$\mu^s_{X_j \to \hat{f}_i} = \sum_{Z^j \subseteq \mathcal{F}^s_{\hat{f}_i}(j)} \prod_{\hat{f}_k \in Z^j} \eta^s_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j) \setminus Z^j} \eta^*_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} \tag{6.12a}$$

$$= \left[\prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \left(\eta^s_{\hat{f}_k \to X_j} + \eta^*_{\hat{f}_k \to X_j} \right) \right] \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} \tag{6.12b}$$

where from Eq. (6.12b), we update $\mu^s_{X_j \to \hat{f}_i}$ by multiplying two parts: the first one represents the probability that X_j satisfies all its constrained factors $\mathcal{F}^s_{\hat{f}_i}(j)$ and the second is the probability that X_j violates all its violating factors $\mathcal{F}^u_{\hat{f}_i}(j)$.

Similarly for the message $\mu^u_{X_j \to \hat{f}_i}$. This message is specified by values of $X_j = u_{i,j}$ and its mega-node $P_j \subseteq \mathcal{F}^u_{\hat{f}_i}(j)$. Thus, we have:

$$\mu^u_{X_j \to \hat{f}_i} = \left\{ \sum_{Z^j \subseteq \mathcal{F}^u_{\hat{f}_i}(j)} \eta_{\hat{f}_i \to X_j} \middle| X_j = u_{i,j}, P_j = Z^j \right\} \tag{6.13a}$$

$$= \sum_{Z^j \subseteq \mathcal{F}^u_{\hat{f}_i}(j), Z^j \neq \emptyset} \prod_{\hat{f}_k \in Z^j} \eta^s_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j) \setminus Z^j} \eta^*_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} \tag{6.13b}$$

$$- \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j}$$

$$= \left[\prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \left(\eta^s_{\hat{f}_k \to X_j} + \eta^*_{\hat{f}_k \to X_j} \right) - \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} \right] \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} \tag{6.13c}$$

where from Eq. (6.13c), we update $\mu^s_{X_j \to \hat{f}_i}$ by multiplying the probability that X_j violates all its constrained factors $\mathcal{F}^s_{\hat{f}_i}(j)$ with the probability that X_j satisfies all its violating factors $\mathcal{F}^u_{\hat{f}_i}(j)$.

Finally, computing the message $\mu^*_{X_j \to \hat{f}_i}$ specifies the values of $X_j = s_{i,j}$ with $P_j = \mathcal{F}^s_{\hat{f}_i}(j)$, and

$X_j = *$ with $P_j = \emptyset$, as follows:

$$\begin{aligned}
\mu^*_{X_j \to \hat{f}_i} &= \left\{ \sum_{Z^j \subseteq \mathcal{F}^s_{\hat{f}_i}(j)} \eta_{\hat{f}_i \to X_j} \middle| X_j = s_{i,j}, P_j = Z^j \right\} + \left. \eta_{\hat{f}_i \to X_j} \middle| X_j = *, P_j = \emptyset \right\} && \text{(6.14a)} \\
&= \sum_{Z^j \subseteq \mathcal{F}^s_{\hat{f}_i}(j), Z^j \neq \emptyset} \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^s_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} \\
&\quad - \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} + \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} && \text{(6.14b)} \\
&= \left[\prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j)} \left(\eta^s_{\hat{f}_k \to X_j} + \eta^*_{\hat{f}_k \to X_j} \right) - \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} \right] \prod_{\hat{f}_k \in \mathcal{F}^u_{\hat{f}_i}(j)} \eta^u_{\hat{f}_k \to X_j} \\
&\quad + \prod_{\hat{f}_k \in \mathcal{F}^s_{\hat{f}_i}(j) \cup \mathcal{F}^u_{\hat{f}_i}(j)} \eta^*_{\hat{f}_k \to X_j} && \text{(6.14c)}
\end{aligned}$$

where from Eq. (6.14c), we update $\mu^*_{X_j \to \hat{f}_i}$ by considering the probability that X_j is unconstrained from either its satisfying factors $\mathcal{F}^s_{\hat{f}_i}(j)$ or its violating factors $\mathcal{F}^u_{\hat{f}_i}(j)$.

Factor-to-Variables Updates

Let us start here with the message $\eta^s_{\hat{f}_i \to X_j}$. This message implies that $X_j = s_{i,j}$ and $\hat{f}_i \in P_j$, and that the only possible assignment for the other ground atoms $X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ is $u_{i,k}$ and their mega-nodes are $P_k \subseteq \mathcal{F}^u_{\hat{f}_i}(k)$. Accordingly, using the definition of $\mu^u_{X_k \to \hat{f}_i}$ in Eq. (6.13a), we obtain the following:

$$\begin{aligned}
\eta^s_{\hat{f}_i \to X_j} &= \left\{ \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left[\sum_{P_k \subseteq \mathcal{F}^u_{\hat{f}_i}(k)} \mu_{X_k \to \hat{f}_i} \right] \times e^{\hat{w}_i \cdot y} \middle| X_k = u_{i,k}, P_k \subseteq \mathcal{F}^u_{\hat{f}_i}(k) \right\} && \text{(6.15a)} \\
&= \left[\prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^u_{X_k \to \hat{f}_i} \right] \times \overbrace{e^{\hat{w}_i \cdot y}}^{\text{reward}} && \text{(6.15b)}
\end{aligned}$$

That is, from Eq. (6.15b), we update $\eta^s_{\hat{f}_i \to X_j}$ by considering the product of the messages of all other ground atoms of $\hat{f}_i$ except X_j that are violating it. Then we reward the result by multiplying it with the factor term $e^{\hat{w}_i \cdot y}$ (refer to Eq. 6.3 for the definition of the reward term).

Now moving to the message $\eta^u_{\hat{f}_i \to X_j}$. This message represents the probability that X_j can violate $\hat{f}_i$. That is to say, we have $X_j = u_{i,j}$ and $P_j \subseteq \mathcal{F}^u_{\hat{f}_i}(j)$. This probability implies a combination of three possibilities (having weights labeled as W_1, W_2 and W_3) for the other ground atoms $X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ in a potential complete assignment:

1. There is one ground atom in $\mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ satisfying $\hat{f}_i$, and all the other ground atoms are violating it

$$\begin{aligned} \mathrm{W}_1 = &\left\{ \sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \sum_{Z^k \subseteq \mathcal{F}^s_{\hat{f}_i}(k)} \mu_{X_k \to \hat{f}_i} \middle| X_k = s_{i,k}, P_k = Z^k \cup \{\hat{f}_i\} \right\} \\ &\times \left\{ \prod_{X_i \in \mathcal{X}_{\hat{f}_i} \setminus \{X_k, X_j\}} \left[\sum_{Z^i \subseteq \mathcal{F}^u_{\hat{f}_i}(i)} \mu_{X_i \to \hat{f}_i} \right] \middle| X_i = u_{i,i}, P_i = Z^i \right\} && \text{(6.16a)} \\ &= \sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^s_{X_k \to \hat{f}_i} \times \prod_{X_i \in \mathcal{X}_{\hat{f}_i} \setminus \{X_k, X_j\}} \mu^u_{X_i \to \hat{f}_i} && \text{(6.16b)} \end{aligned}$$

2. There are two or more ground atoms in $\mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ satisfying $\hat{f}_i$ or equal joker $*$, and all other ground atoms are violating it

$$\begin{aligned} \mathrm{W}_2 = &\sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left[\left\{ \sum_{Z^k \subseteq \mathcal{F}^s_{\hat{f}_i}(k)} \mu_{X_k \to \hat{f}_i} \middle| X_k = s_{i,k}, P_k = Z^k \right\} + \left\{ \mu_{X_k \to \hat{f}_i} \middle| X_k = *, P_k = \emptyset \right\} \right] \\ &\times \left\{ \prod_{X_i \in \mathcal{X}_{\hat{f}_i} \setminus \{X_k, X_j\}} \left[\sum_{Z^i \subseteq \mathcal{F}^u_{\hat{f}_i}(i)} \mu_{X_i \to \hat{f}_i} \right] \middle| X_i = u_{i,i}, P_i = Z^i \right\} && \text{(6.17a)} \\ &= \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left[\mu^u_{X_k \to \hat{f}_i} + \mu^*_{X_k \to \hat{f}_i} \right] - \sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^*_{X_k \to \hat{f}_i} \times \prod_{X_i \in \mathcal{X}_{\hat{f}_i} \setminus \{X_k, X_j\}} \mu^u_{X_i \to \hat{f}_i} \\ &- \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^u_{X_k \to \hat{f}_i} && \text{(6.17b)} \end{aligned}$$

   Note that the weight assigned to the event that each ground atom is either satisfying or $*$ is $\prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left[\mu^u_{X_k \to \hat{f}_i} + \mu^*_{X_k \to \hat{f}_i} \right]$, and the weight W_2 is given by subtracting from this quantity the weight assigned to the event that there are not at least two joker ground atoms $*$ or satisfying. This event is a combination of two disjoint events that either all other ground atoms in $X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ are violating (which weight $\prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^u_{X_k \to \hat{f}_i}$) or that only one ground atom is $*$ or satisfying (with weight $\sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^*_{X_k \to \hat{f}_i} \times \prod_{X_i \in \mathcal{X}_{\hat{f}_i} \setminus \{X_k, X_j\}} \mu^u_{X_i \to \hat{f}_i}$).

3. All other ground atoms in $\mathcal{X}_{\hat{f}_i} \setminus \{X_j\}$ are violating $\hat{f}_i$. So here, there is a penalty factor

$e^{-\hat{w}_i \cdot y}$ for updating the message:

$$\mathrm{W}_3 = \left\{ \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \Big[\sum_{Z^k \subseteq \mathcal{F}^u_{\hat{f}_i}(k)} \mu_{X_k \to \hat{f}_i} \Big] \times e^{-\hat{w}_i \cdot y} \middle| X_k = s_{i,k}, P_k = Z^k \right\} \tag{6.18a}$$

$$= \left[\prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^u_{X_k \to \hat{f}_i} \right] \times \overbrace{e^{-\hat{w}_i \cdot y}}^{\text{penalty}} \tag{6.18b}$$

Now, bringing together the weight forms of W$_1$, W$_2$, and W$_3$ from Eqs. (6.16b), (6.17b) and (6.18b) results in:

$$\begin{aligned} \eta^u_{\hat{f}_i \to X_j} = \Bigg[& \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left(\mu^u_{X_k \to \hat{f}_i} + \mu^*_{X_k \to \hat{f}_i} \right) + \prod_{X_i \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j, X_k\}} \mu^u_{X_i \to \hat{f}_i} \\ & \sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left(\mu^s_{X_k \to \hat{f}_i} - \mu^*_{X_k \to \hat{f}_i} \right) \Bigg] - \Bigg[\overbrace{(1 - e^{-\hat{w}_i \cdot y})}^{\text{penalty}} \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^u_{X_k \to \hat{f}_i} \Bigg] \end{aligned} \tag{6.19}$$

Hence, from Eq. (6.19), updating $\eta^u_{\hat{f}_i \to X_j}$ requires considering the difference between two parts. The first includes the probability that two (or more) ground atoms in $\hat{f}_i$ satisfies it plus the probability that there is exactly one of $\hat{f}_i$'s ground atoms (except X_j) satisfying $\hat{f}_i$ and all other ground atoms are violating it. The second part involves the probability that all other ground atoms are violating f_i. This latter probability is penalized with the factor $(1 - e^{-w_j \cdot y})$ since the same part is rewarded in Eq. (6.15b) when satisfying $\hat{f}_i$.

Finally, the message $\eta^*_{\hat{f}_i \to X_j}$ represents the probability that X_j can be unconstrained by $\hat{f}_i$. This probability is a combination of two possibilities: either X_j is satisfying $\hat{f}_i$ and all other ground atoms are unconstrained, or X_j is unconstrained (i.e., $X_j = *$ with $P_i = \emptyset$)

$$\eta^*_{\hat{f}_i \to X_j} = \sum_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left[\left\{ \sum_{Z^k \subseteq \mathcal{F}^s_{\hat{f}_i}(k)} \mu_{X_k \to \hat{f}_i} \middle| X_k = s_{i,k}, P_k = Z^k \right\} + \left\{ \mu_{X_k \to \hat{f}_i} \middle| X_k = *, P_k = \emptyset \right\} \right] \tag{6.20}$$

Note that the first part of Eqs. (6.17a) and (6.17b) is identical to Eq. (6.20). Thus, we substitute the computation of this part from Eqs. (6.17a) and (6.17b) into Eq. (6.20), and we have:

$$\eta^*_{\hat{f}_i \to X_j} = \left[\prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \left(\mu^u_{X_k \to \hat{f}_i} + \mu^*_{X_k \to \hat{f}_i} \right) \right] - \prod_{X_k \in \mathcal{X}_{\hat{f}_i} \setminus \{X_j\}} \mu^u_{X_k \to \hat{f}_i} \tag{6.21}$$

where, as in Eq. (6.21), we simply update $\eta^*_{\hat{f}_i \to X_j}$ by considering the messages on which the other ground atoms in $\hat{f}_i$ except X_j are either unconstrained or satisfying $\hat{f}_i$ minus the probability that they are violating $\hat{f}_i$.

Estimating the Marginals

Now let us explain the derivation of ground atoms' marginals over max-cores in $\hat{\mathcal{G}}$. Computing the unnormalized positive marginal of a ground atom X_j requires multiplying the satisfying income messages from the ground clauses in which X_j appears positively by the violating income messages from the ground clauses in which X_j appears negatively:

$$\tilde{\theta}_j^{+} = \prod_{\hat{f}_i \in \mathcal{F}^s(j)} \left\{ \eta_{\hat{f}_i \to X_j} \middle| X_j = s_{i,j}, P_j = \mathcal{F}^s(j) \right\} \times \prod_{\hat{f}_i \in \mathcal{F}^u(j)} \left\{ \eta_{\hat{f}_i \to X_j} \middle| X_j = u_{i,j}, P_j = \mathcal{F}^u(j) \right\} \tag{6.22a}$$

$$= \prod_{\hat{f}_i \in \mathcal{F}_{X_j+}} \left\{ \eta_{\hat{f}_i \to X_j} \middle| X_j = +, P_j = \mathcal{F}_{X_j+} \right\} \times \prod_{\hat{f}_i \in \mathcal{F}_{X_j-}} \left\{ \eta_{\hat{f}_i \to X_j} \middle| X_j = -, P_j = \mathcal{F}_{X_j-} \right\} \tag{6.22b}$$

$$= \prod_{\hat{f}_i \in \mathcal{F}_{X_j-}} \eta^u_{\hat{f}_i \to X_j} \times \left[\prod_{\hat{f}_i \in \mathcal{F}_{X_j+}} \left(\eta^s_{\hat{f}_i \to X_j} + \eta^*_{\hat{f}_i \to X_j} \right) - \prod_{\hat{f}_i \in \mathcal{F}_{X_j+}} \eta^*_{\hat{f}_i \to X_j} \right] \tag{6.22c}$$

Similarly, we can obtain the unnormalized negative marginal by multiplying the satisfying income messages from the factors in which X_i appears negatively by the violating income messages from the factors in which X_i appears positively:

$$\tilde{\theta}_j^{-} = \prod_{\hat{f}_i \in \mathcal{F}_{X_j+}} \eta^u_{\hat{f}_i \to X_j} \times \left[\prod_{\hat{f}_i \in \mathcal{F}_{X_j-}} \left(\eta^s_{\hat{f}_i \to X_j} + \eta^*_{\hat{f}_i \to X_j} \right) - \prod_{\hat{f}_i \in \mathcal{F}_{X_j-}} \eta^*_{\hat{f}_i \to X_j} \right] \tag{6.23}$$

Finally, we can estimate the unnormalized joker marginal by multiplying all the unconstrained incoming messages from all factors in which X_j appears:

$$\tilde{\theta}_j^{*} = \prod_{\hat{f}_i \in \mathcal{F}_{X_j}} \left\{ \eta_{\hat{f}_i \to X_j} \middle| X_j = *, P_j = \emptyset \right\}$$

$$= \prod_{\hat{f}_i \in \mathcal{F}_{X_j}} \eta^*_{\hat{f}_i \to X_j} \tag{6.24a}$$

Now by normalizing the quantities in Eqs. (6.22c), (6.23) and (6.24a), we obtain the marginal of X_j as follows:

$$\beta^{+}_{X_j} = \mathcal{Z}_j^{-1}\tilde{\theta}^{+}_j \tag{6.25a}$$

$$\beta^{-}_{X_j} = \mathcal{Z}_j^{-1}\tilde{\theta}^{-}_j \tag{6.25b}$$

$$\beta^{*}_{X_j} = \mathcal{Z}_j^{-1}\tilde{\theta}^{*}_j \tag{6.25c}$$

$$\tag{6.25d}$$

and

$$\mathcal{Z}_j = \tilde{\theta}^{+}_j + \tilde{\theta}^{-}_j + \tilde{\theta}^{*}_j \tag{6.26}$$

where $\mathcal{Z}_j$ is the normalizing constant, given the evidence E.

The simple interpretation of the positive marginal $\beta^{+}_{X_j}$, in Eqs. (6.22c) and (6.25a), is that it estimates the probability of randomly picking up a max-core (i.e., a cluster), and finding that X_j is frozen to "+". This max-core satisfies the ground clauses of total maximal weights in which X_j appears positively and violates the ground clauses of total minimal weights in which X_j appears negatively. In other words, $\beta^{+}_{X_j}$ approximates the fraction of the max-cores that contains potentially the optimal MAP solutions in which X_j ="+".

Hence, when $\beta^{+}_{X_j}$ is greater than $\beta^{-}_{X_j}$, it is an indication that the fraction of clusters containing X_j frozen to "+" have MAP solutions better optimized (i.e., having more total weights) than the fraction of clusters in which X_j is frozen to "−". This means that the probability of getting inside a cluster that involves optimal solutions increases when X_j is frozen to "+" compared to X_j frozen to "−". This intuitively implies that fixing X_j ="+" is more likely to be a part of the optimal MAP solution than X_j ="+".

6.2 Using WSP-χ for MAP Inference in Markov Logic

As demonstrated in the previous subsection, the marginals obtained from WSP-χ correspond to surveys over max-cores representing clusters of MAP solutions. That is to say, they provide information about the fraction of clusters in which ground atoms are frozen or unfrozen in their MAP solutions. Thus, a direct use of that information is to apply a marginalization-decimation algorithm (Kroc *et al.*, 2009) based on WSP-χ, recovering a family of WSP-χ inspired decimation (**WSP-Dec**) algorithms for solving MAP inference on SRL models.

WSP-χ Inspired Decimation

For convenience, and without loss of generality, we focus on MLN when explaining how WSP-Dec finds a MAP solution. As clarified in Algorithm 6, WSP-Dec is a two-stage strategy:

Algorithm 6 WSP-Dec for MAP inference in MLN.

```
Input: Set of Clauses and their Weights ($\mathcal{F}$,$\mathcal{W}$), set of query atoms $\mathcal{X}$, Evidence database
       $\mathcal{DB}$, Maximum number of iterations $\mathcal{I}_{\max}$, cooling parameter $\hat{y}$, Magnetization thresh-
       old $\hat{\mathcal{T}}$, smoothing parameter $\hat{\chi}$.
Output: MAP solution $\mathcal{X}^{MAP}$.
 1: Set the parameters $y = \hat{y}$, $\mathcal{T} = \hat{\mathcal{T}}$ and $\chi = \hat{\chi}$;
    // Using WSP-$\hat{\chi}$ as a pre-processing
 2: $\eta_{\hat{f}_i \leftarrow X_j} \in \mathcal{U}\,[0,1]\,,\ \forall X_j \in \mathcal{X},\ \forall f_i \in \mathcal{F}$;    // Messages initialization;
 3: repeat   // Updating the messages
 4:    Use $\eta_{\hat{f}_i \to X_j}$ to update $\mu_{X_j \to \hat{f}_i}$;    // Using Eqs.(6.12b),(6.13c), and (6.14c)
 5:    Use $\mu_{X_j \to \hat{f}_i}$ to update $\eta_{\hat{f}_i \to X_j}$;    // Using Eqs.(6.15b),(6.19), and (6.21)
 6: until (Convergence or termination of $\mathcal{I}_{\max}$)
 7: Return a fixed point of the messages $\hat{\eta}_{\hat{f}_i \to X_j}$;
 8: if (non-trivial $\hat{\eta}_{\hat{f}_i \to X_j} \neq 0$ are found) then
 9:    for each $X_j \in \mathcal{X}$ do
10:       Compute: $\beta_{X_j} = [\beta^{+}_{X_j}, \beta^{-}_{X_j}]$; //Using Eqs.(6.22c),(6.23),(6.24a), (6.25a), (6.25b), (6.25c)
11:    end for
12:    $\beta \leftarrow$ Select[ground atoms $X_j$s having ( $|\beta^{+}_{X_j} - \beta^{-}_{X_j}| > \mathcal{T}$)]  // obtain frozen ground atoms

13:    $\mathcal{X}^* \leftarrow \mathcal{X} \setminus \{\beta\}$;   // remove $\beta$ from queries
14:    $\mathcal{DB}^* \leftarrow \mathcal{DB} \cup \{\beta\}$;   // add $\beta$ to evidence database
15:    $\beta^* \leftarrow \beta^* \cup \beta$;    // store all $\beta$ as a portion of $\mathcal{X}^{MAP}$
16:    Simplify the model clauses $\mathcal{F}$ into $\mathcal{F}^*$;   // Fix frozen ground atoms ($\beta$)
17:    Go to (2) and Re-run WSP-$\hat{\chi}$ for ($\mathcal{F} := \mathcal{F}^*, \mathcal{X} := \mathcal{X}^*$);
18: else if (trivial $\hat{\eta}_{\hat{f}_i \to X_j} = 0$ are found) then
19:    Run MaxWalkSAT on the simplified grounded network constructed by ($\mathcal{X}^*, \mathcal{F}^*, \mathcal{DB}^*$);
20: end if
21: $\mathcal{X}^{MAP} \leftarrow$ Combination of returns from steps 15 and 19;
22: Return $\mathcal{X}^{MAP}$;
```

- In the first stage, the goal is to use WSP-χ for scaling the MLN's grounded network and obtaining a portion of the optimal MAP solution, as follows:
  - We assign the smoothing parameter χ the value $\hat{\chi}$ to specify the WSP-$\hat{\chi}$ algorithm from the family WSP-χ that will be used as pre-processing. We then adjust its

setting for both the cooling parameter $\hat{y}$ and magnetization[2] threshold $\hat{\mathcal{T}}$ (line 1).

- The specified WSP-$\hat{\chi}$ starts by initializing its messages uniformly at random, as in line 2. It then iteratively applies a set of decimation steps until reaching a trivial fixed point of the messages.
- At each decimation step, lines 8-11, it iteratively updates its messages, using Eqs.(6.12b),(6.13c), (6.14c), (6.15b), (6.19), and (6.21), until either exceeding the maximum number of iterations or converging to a non-trivial fixed point of the messages (lines 4-7). It then estimates the marginals of ground query atoms in $\mathcal{X}$ using Eqs.(6.22c),(6.23),(6.24a), (6.25a), (6.25b), and (6.25c).
- Subsequently, as in line 12, it uses the computed marginals to identify frozen ground atoms (i.e, cluster backbones): the fraction of ground atoms in $\mathcal{X}$ that have a magnetization $|\beta^{+}_{X_j} - \beta^{-}_{X_j}| \geq \mathcal{T}$.[3]
- Afterwards, it fixes the frozen ground atoms to their more likely truth values (i.e., magnetized values). In addition, as in line 14, it adds them to the evidence database $\mathcal{DB}$: those whose ($\beta^{+}_{X_j} < \beta^{-}_{X_j}$) are added to $\mathcal{DB}$ as false evidence, and those whose ($\beta^{+}_{X_j} > \beta^{-}_{X_j}$) are added as true evidence.
- At this point, it should be noted that the advantage of fixing the frozen ground atoms is two-fold: shrinking the set of query atoms (i.e., $\mathcal{X} \setminus \{\beta\}$), and enlarging the evidence database (i.e., $\mathcal{DB} \cup \{\beta\}$). This in turn results in reducing the set of ground clauses, and therefore simplifying the grounded network that instantiates the MAP inference problem (line 16).
- It successively repeats the decimation process over the simplified MAP problem to increase the set of the frozen ground atoms (line 17).
- Eventually, as in line 18, if it reaches a trivial fixed point of the messages: those that often produce demagnetized marginals (i.e., marginals that are not biased to either positive or negative values) and yield paramagnetic solutions (paramagnetic solution refers to a generalized complete assignment that is not biased to any value for all variables). Then either the complex parts of the grounded network have been decimated by fixing the frozen ground atoms and/or the remaining query ground atoms define a simple MAP inference that can be efficiently solved using any off-the-shelf local search algorithm (e.g., MaxWalkSAT).

[2] Given the marginal probability of a variable X_j, the magnetization of X_j is the difference between marginals of the variable being positive and it being negative.

[3] Note that the schema, that is used here to identify the frozen ground atoms, is identical to the one that is used previously for filtering the query variables in Algorithm 4.

- In the second stage, we run the MaxWalkSAT algorithm (line 19) to solve the remaining simplified MAP inference problem. The output returned from MaxWalkSAT combined with the total set of the frozen ground atoms obtained from the WSP will provide the overall MAP solution (line 21).

6.3 Combining WSP-χ with Lazy MAP Inference

One key advantage of WSP-χ algorithms is that they can be combined with other state-of-the-art approaches which greatly improve the scalability of MAP inference such as Lazy and Lifted. Algorithm 7 shows how to combine WSP-χ with a Lazy MAP inference, which yields **Lazy-WSP-Dec**. Lazy-WSP-Dec mainly differs from the WSP-Dec of Algorithm 6 in both the initial set of underlying query atoms and clauses, and the local search algorithm that will be used to solve the simplified MAP inference (line 19). That is, Lazy-WSP-Dec starts by grounding the network lazily, and maintaining only active ground clauses and their active ground atoms that are sufficient to answer the queries. It then calls the specified WSP-$\hat{\chi}$ algorithm, steps 1-17 of algorithm 6, to scale the lazy ground network, which was built using those active clauses and atoms, by fixing the frozen active atoms. After reaching a trivial fixed point, it runs Lazy-MaxWalkSat, as in line 6, on the simplified network instead of propositional MaxWalkSAT as in algorithm 6.

Algorithm 7 Combining WSP-Dec with lazy MAP inference.

```
Input: Set of Clauses and their Weights ($\mathcal{F}$,$\mathcal{W}$), set of query atoms $\mathcal{X}$, Evidence database $\mathcal{DB}$,
       Maximum number of iterations $\mathcal{I}_{\max}$, cooling parameter $\hat{y}$, Magnetization threshold $\mathcal{T}$.
Output: MAP solution $\mathcal{X}^{MAP}$.
1: $\mathcal{X} \leftarrow$ (atoms in clauses unsatisfied by $\mathcal{DB}$);    // Consider only active atoms
2: $\mathcal{F} \leftarrow$ (clauses activated by $\mathcal{X}$);    // Consider only active clauses
3: $\mathcal{W} \leftarrow$ Weights associated with $\mathcal{F}$;
   // Call WSP-$\chi$: steps 1-17 in algorithm 6.
4: $[\mathcal{X}^*, \mathcal{F}^*, \mathcal{DB}^*] \leftarrow$ WSP-$\hat{\chi}(\mathcal{X}, \mathcal{F}, \mathcal{W})$;   //Simplifying the lazy grounded network
5: if (trivial $\hat{\eta}_{\hat{f}_i \to X_j} = 0$ are found) then
6:     Run Lazy-MaxWalkSAT on the lazy grounded network constructed by $(\mathcal{X}^*, \mathcal{F}^*, \mathcal{DB}^*)$;
7: end if
```

6.4 Experimental Evaluation

The goal of our experimental evaluation is to investigate the following key questions.

- (**Q1**) Is the WSP-Dec algorithm competitive with the state-of-the-art inference algorithms for finding an optimal MAP solution?

- (**Q2**) Is WSP-χ powerful enough to reduce significantly the size of grounded networks compared to the prominent state-of-the-art scalable methods such as Lazy Inference?

- (**Q3**) How is the behavior of WSP-Dec influenced by the choice of the cooling parameter y and magnetization threshold $\mathcal{T}$?

- (**Q4**) How is the performance of WSP-Dec algorithm affected by tuning the value of the smoothing parameter χ in WSP-χ?

- (**Q5**) Does the combination of WSP-χ with Lazy inference, i.e., Lazy-WSP-Dec, improve the efficiency of WSP-χ based MAP inference?

We experimented on a protein interaction task, a hyperlink analysis task, and an entity resolution task in a citation matching domain. We used both the MLNs and datasets available from the Alchemy web page[4].

Protein Interaction. We used both the MLN model (Davis and Domingos, 2009) and *Yeast* dataset, which have been already described in detail in Chapter 4.

- **Query:** The goal of MAP inference here is to predict truth MAP solution of interaction relations (i.e., *Interaction, and Function*). All other atoms (e.g., location, protein-class, enzyme, etc.) are considered evidence atoms.

Hyperlink Analysis. We used the *WebKB* dataset that consists of labeled web pages from the computer science departments of four universities. It features 4165 web pages and $10,935$ web links, along with the words on the webpages, anchors of the links, and neighborhoods around each link. Each web page is marked with some subset of the categories: person, student, faculty, professor, department, research project, and course.

- **MLN:** We used the MLN model (Kok *et al.*, 2007) that involves only formulas linking words to page classes, and page classes to the classes of linked pages. The final knowledge base contains 3 atoms and 6 formulas.

- **Query:** The goal of MAP inference is to predict truth MAP solution of the web pages point to each other, given their topics.

[4] http://alchemy.cs.washington.edu/

Entity Resolution. We used both the MLN model (Singla and Domingos, 2006a) and *Cora* dataset (Davis and Domingos, 2009), which have been already described in detail in Chapter 4.

- **Query:** The goal of MAP inference is to predict truth MAP solution 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.

6.4.1 Methodology

To evaluate WSP-Dec, we compare its results with both the MaxWalkSAT (MWS) algorithm (Kautz *et al.*, 1997; Selman *et al.*, 1993) and its lazy version (Lazy-MWS) which are the state-of-the-art MAP inference algorithms in alchemy system (Kok *et al.*, 2007). In addition, to obtain robust answers to the proposed questions, we did the following:

- Varying the number of objects in the domains, following the methodology previously used for MLNs (Poon *et al.*, 2008)
- Varying the cooling parameter, following the methodology used for relaxed SP (Chieu and Lee, 2009)
- Varying the magnetization threshold $\mathcal{T} \in \{0.2, 0.5\}$. Thus we mainly considered two WSP-Dec algorithms, **WSP-Dec-0.2 and WSP-Dec-0.5**, on which we selected the ground atom as a frozen if its magnetization is greater than 0.2 and 0.5, respectively.

We then conducted the experimental evaluations in the following manner. In the training phase, we learned the weights using a preconditioned scaled conjugate gradient (PSCG) algorithm (Lowd and Domingos, 2007) by performing a four-way cross-validation for protein interaction task, and a five-way cross-validation for both the link prediction and entity resolution tasks. In the testing phase, we carried out a MAP inference on the held-out dataset using six underlying inference algorithms (WSP-Dec-0.2, WSP-Dec-0.5, MWS, Lazy-WSP-Dec-0.2, Lazy-WSP-Dec-0.5, Lazy-MWS) to produce the MAP solution. .

All of the experiments were run on a cluster of nodes with 3.0 GHz Intel CPUs, 3 GB of RAM, RED HAT Linux 5.5. We used the MWS algorithm and its lazy version as implemented in the Alchemy software (Kok *et al.*, 2007), and took advantage of the SP code[5] in implementing our WSP-χ as an extension to the Alchemy software (Kok *et al.*, 2007).

[5] Available: http://users.ictp.it/ zecchina/SP/

6.4.2 Metrics

In order to compare the performance and scalability of the testbed algorithms we considered three metrics:

- The quality of MAP solution as a function of the running time.
- The quality of MAP solution as a function of cooling parameter.
- The average percentage of fixed (frozen) ground atoms.

where the quality of MAP solution is measured by computing the average cost of unsatisfied ground clauses by the obtained MAP solution. It is worth noting that solving the MAP inference here is equivalent to solving a weighted MAX-SAT problem where the goal is to find the MAP solution that maximizes the total weight of the satisfying clauses (which is identical to minimize the total weight of the unsatisfying clauses). Thus considering the average cost of unsatisfied ground clauses serves as a quite good measurement to test the quality of the obtained MAP solution.

6.4.3 Results

We conducted our experimental evaluations through three experiments.

Experiment I

Figures (6.2, 6.3, 6.4), (6.5, 6.6, 6.7) and (6.8, 6.9, 6.10) display the average cost of unsatisfied clauses (smaller is better) as a function of time for the six underlying inference algorithms at three different numbers of objects in the domains of Cora, UW-CSE, and Yeast datasets respectively. Notation used to label each of these figures is as: MLN-*number-of-objects* (*number of ground clauses in the propositional MLN*). The absence of some algorithms in some plots means that no results could be obtained since they ran out of memory.

In terms of the quality of solutions, the results show that the WSP-Dec algorithm at thresholds 0.2 and 0.5 finishes at least 43% more accurately than the propositional MaxWAlkSAT on both WebKB and Yeast datasets, and 51% more accurately on the Cora dataset. It was also very competitive with Lazy MaxWalkSAT inference on Cora and WebKB datasets, and at least 19.5% more accurate on Yeast datasets, even though Lazy MaxWalkSAT inference handles approximately half the number of ground clauses that are tackled by WSP-Dec (*answering Q1*). In addition, lazy-WSP-Dec at thresholds 0.2, 0.5 dominates both the propositional MaxWalkSAT and the Lazy MaxWalkSAT by wide margins on all underlying tested

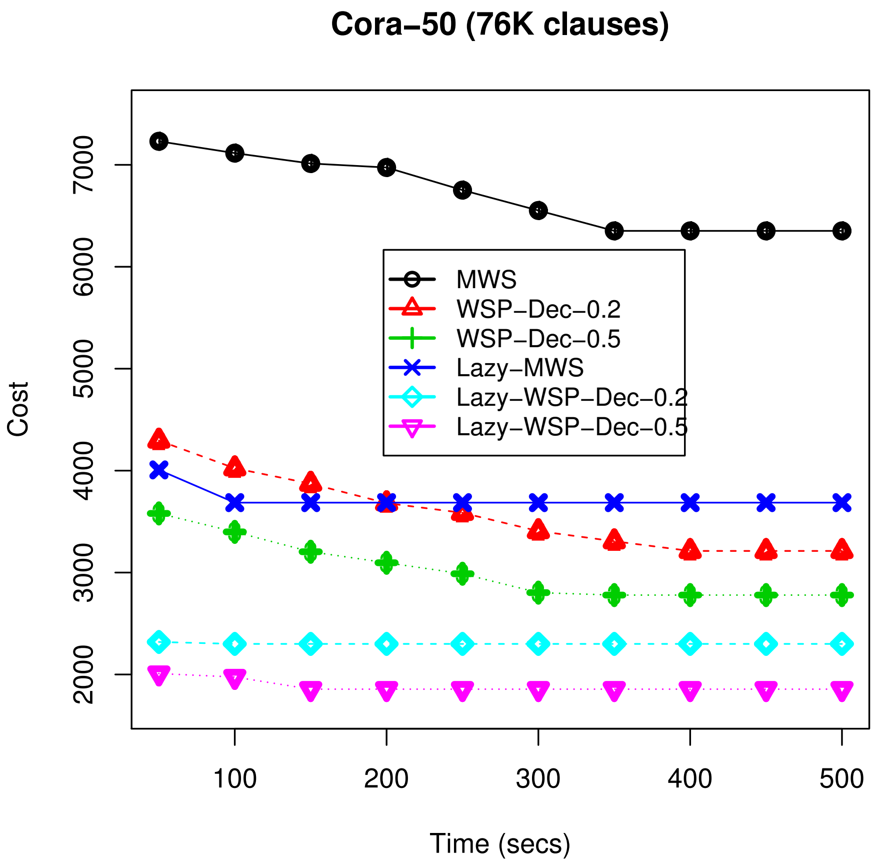

Figure 6.2 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for Cora at 50 objects.

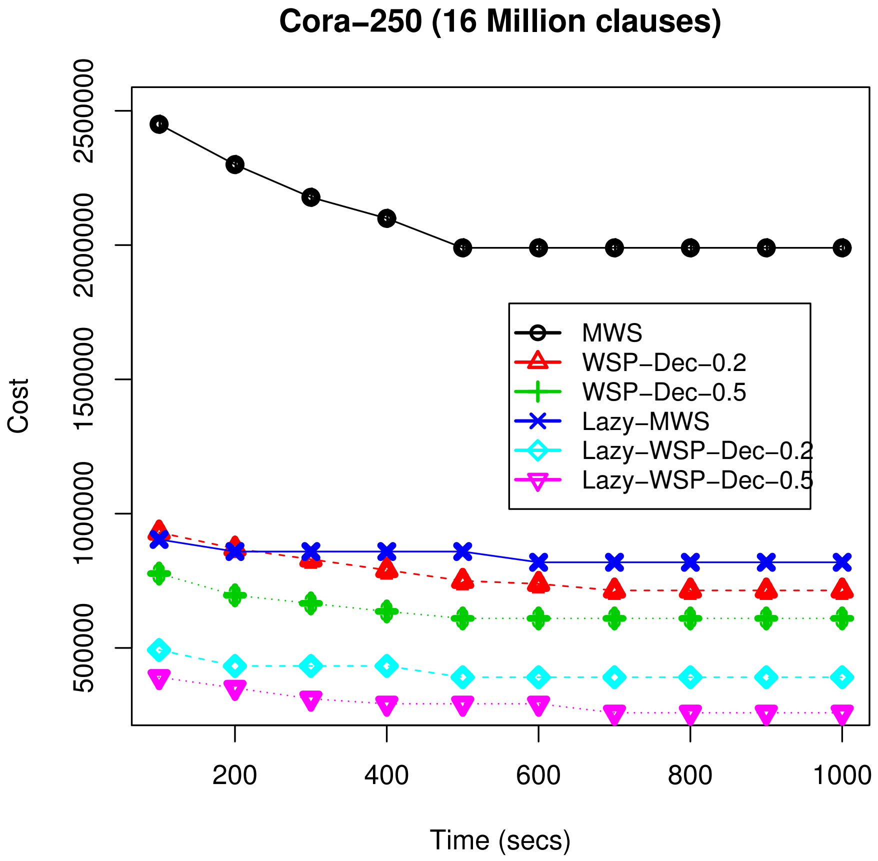

Figure 6.3 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for Cora at 250 objects.

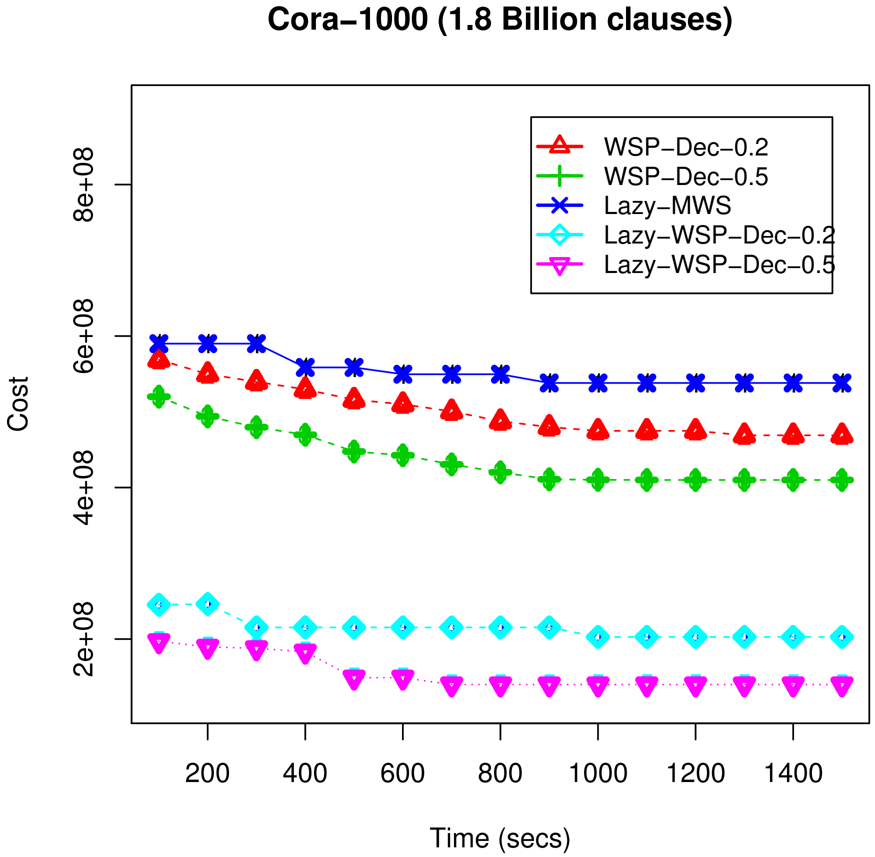

Figure 6.4 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for Cora at 1000 objects.

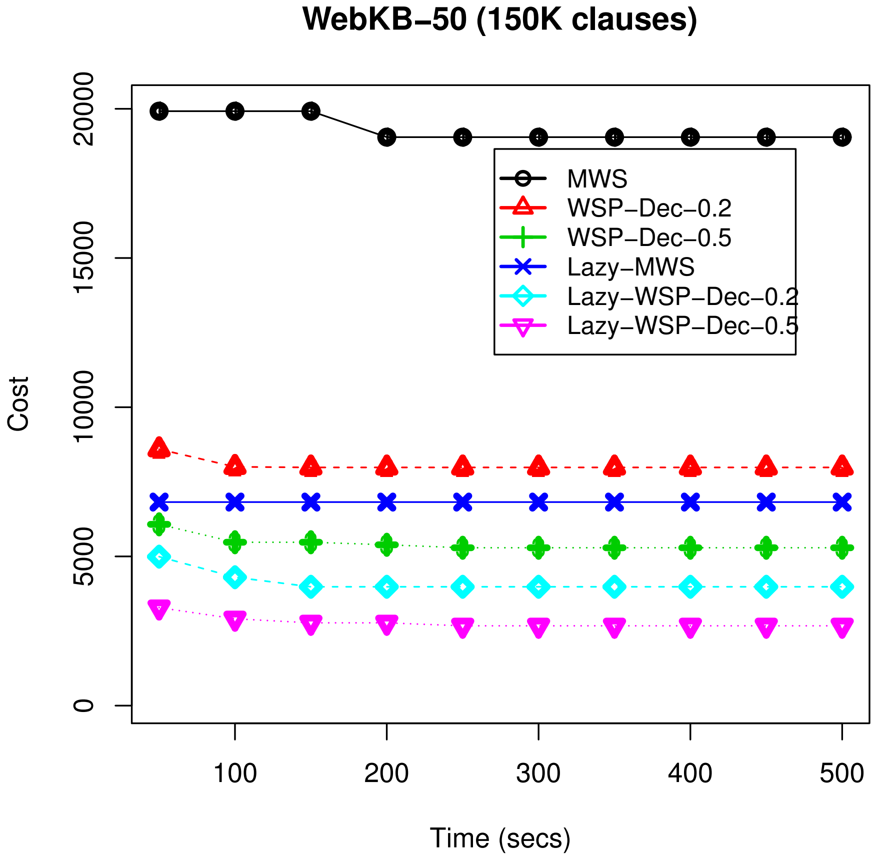

Figure 6.5 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for WebKB at 50 objects.

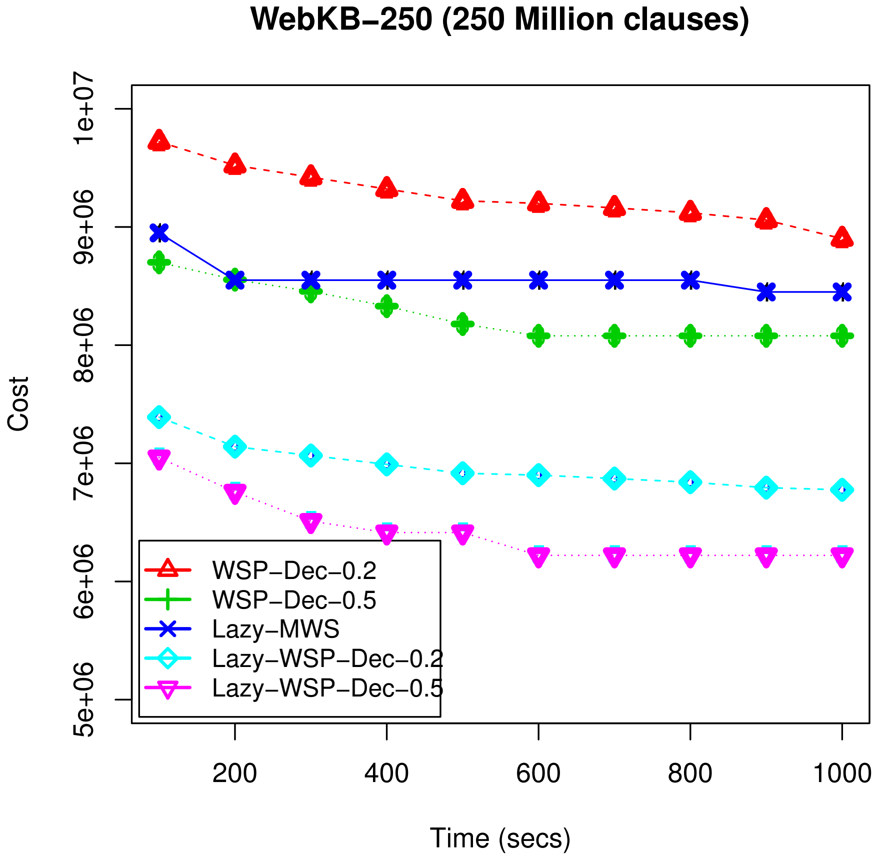

Figure 6.6 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for WebKB at 250 objects.

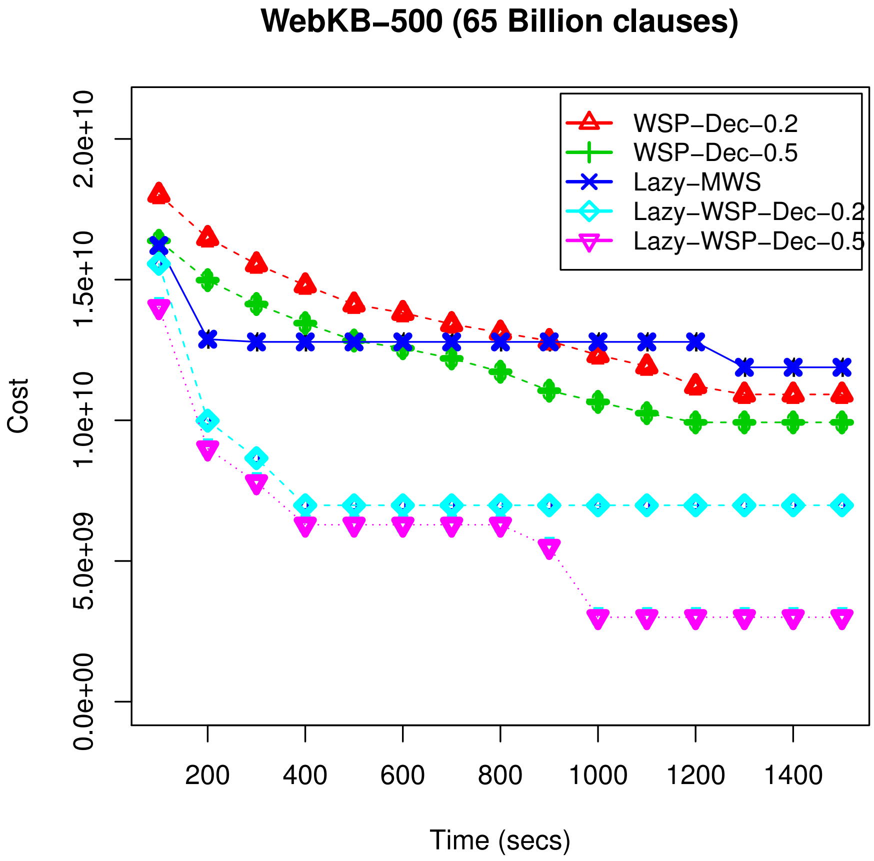

Figure 6.7 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for WebKB at 1000 objects.

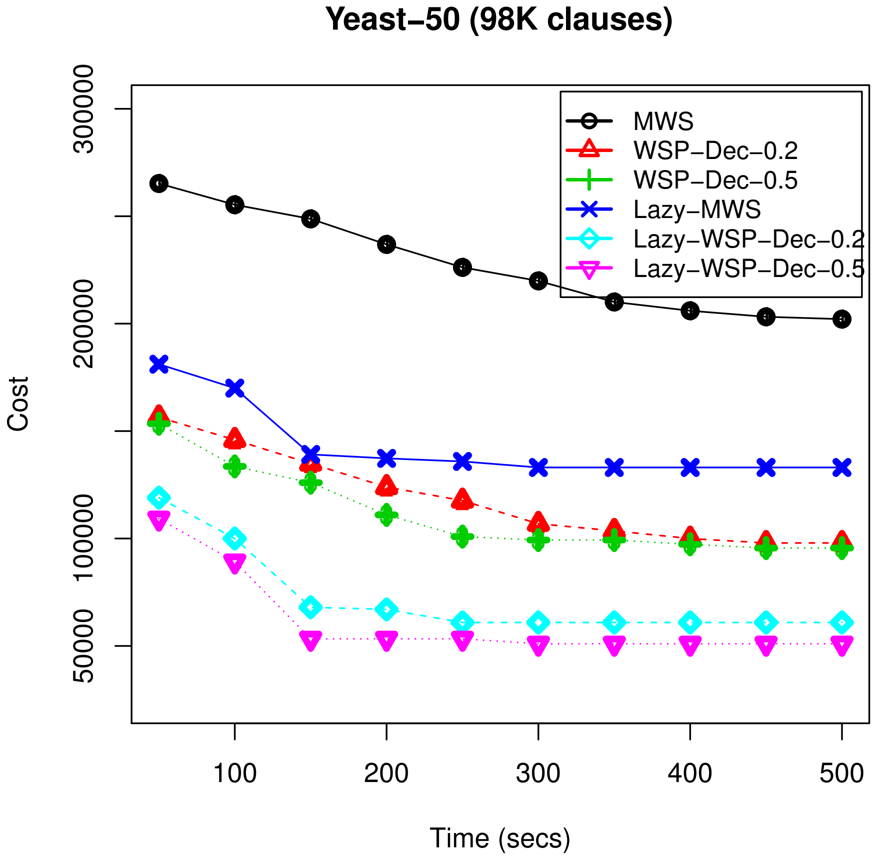

Figure 6.8 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for Yeast at 50 objects.

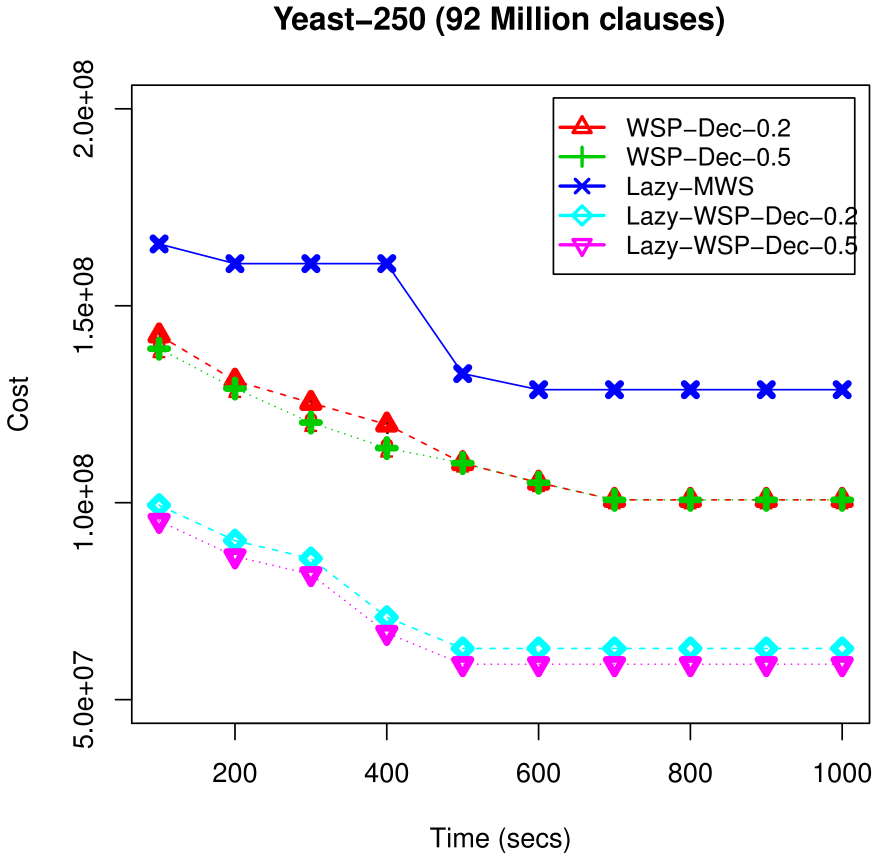

Figure 6.9 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for Yeast at 250 objects.

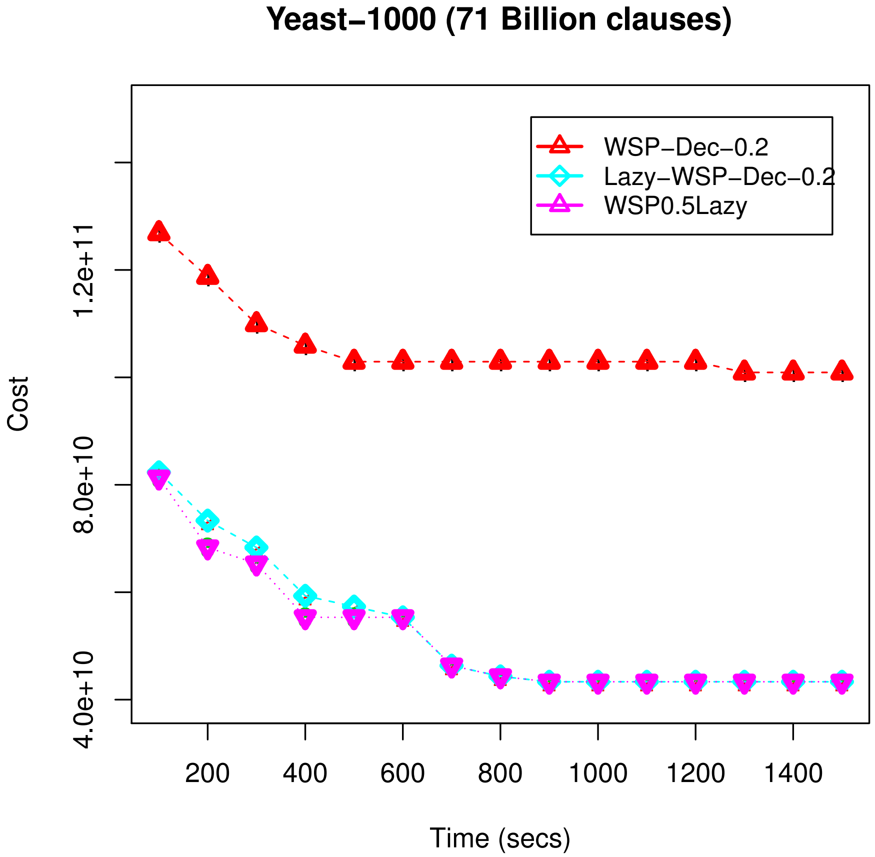

Figure 6.10 Cost vs. Time: average cost of unsatisfied clauses (smaller is better) against time for Yeast at 1000 objects.

datasets (*answering Q5*). Clearly, the WSP-Dec algorithms at threshold 0.5 were marginally more accurate than the WSP-Dec algorithms at threshold 0.2 on both Cora and WebKB datasets (*answering the part of Q3 related to magnetization threshold*). In terms of scalability, both Lazy-WSP-Dec at thresholds 0.2, 0.5 and WSP-Dec at threshold 0.2 were able to handle all full datasets, whereas Lazy MaxWalkSAT and WSP-Dec at threshold 0.5 ran out of memory with 1000 objects in the Yeast dataset (*answering Q2*). Additionally, both Lazy MaxWalkSAT and WSP-Dec at threshold 0.5 dominated propositional MaxWalkSAT, which ran out of with 1000 objects in both Cora and Yeast datasets, 500 objects in the Cora dataset, and 250 objects in Yeast dataset.

Experiment II

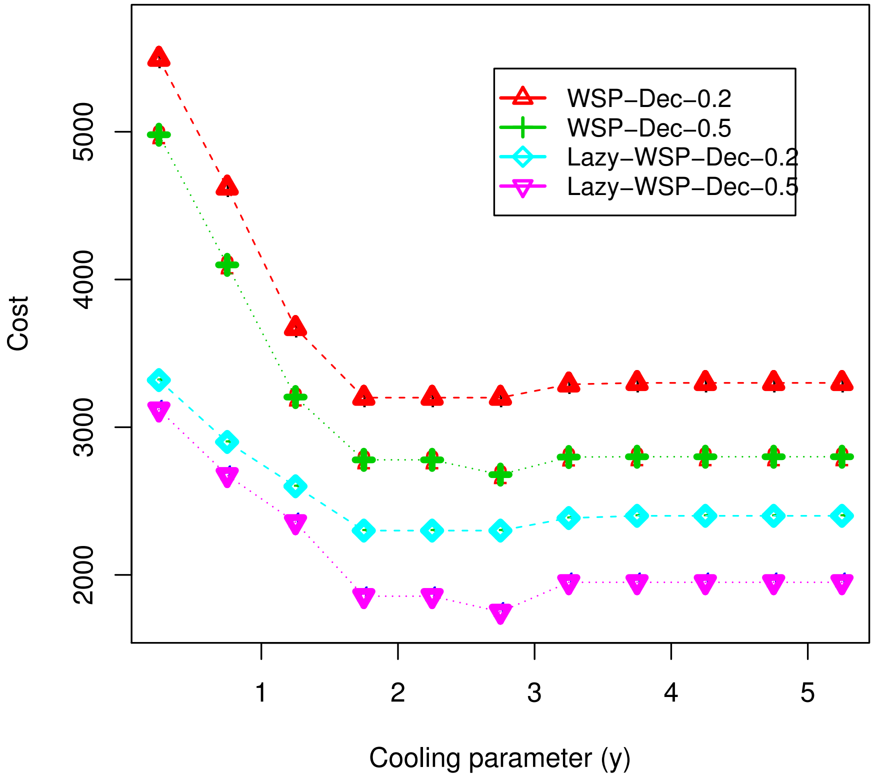

Figure 6.11 Cost vs. cooling parameter: average cost of unsatisfied clauses (smaller is better) against different values of cooling parameter y of WSP-Dec algorithm for Cora.

Figures 6.11, 6.12 and 6.13 display the average cost of unsatisfied clauses as a function of

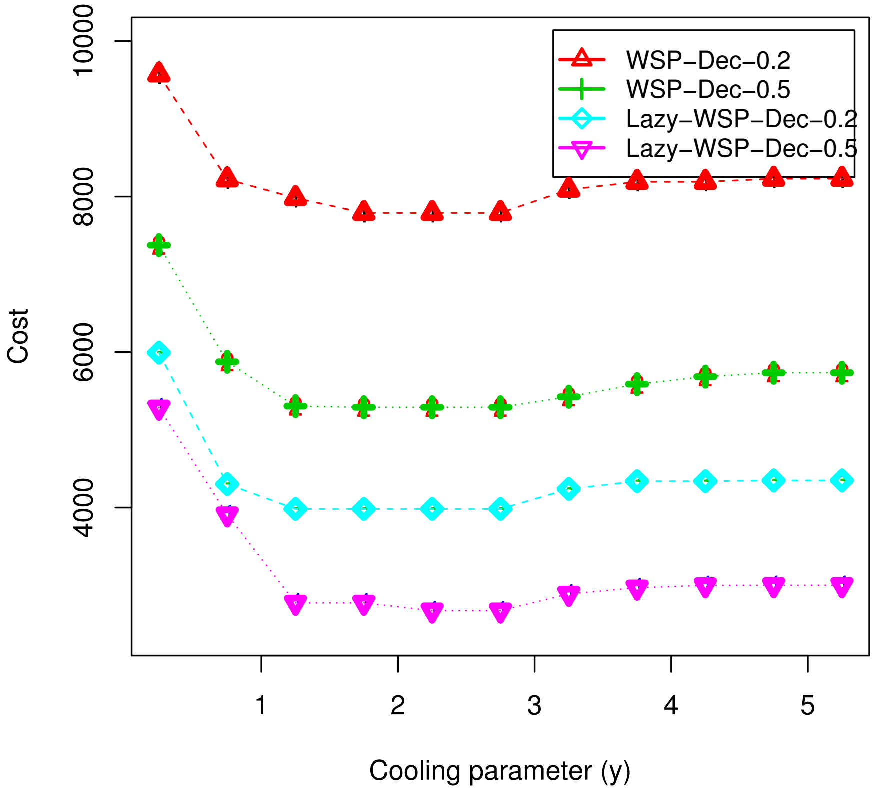

Figure 6.12 Cost vs. cooling parameter: average cost of unsatisfied clauses (smaller is better) against different values of cooling parameter y of WSP-Dec algorithm for WebKB.

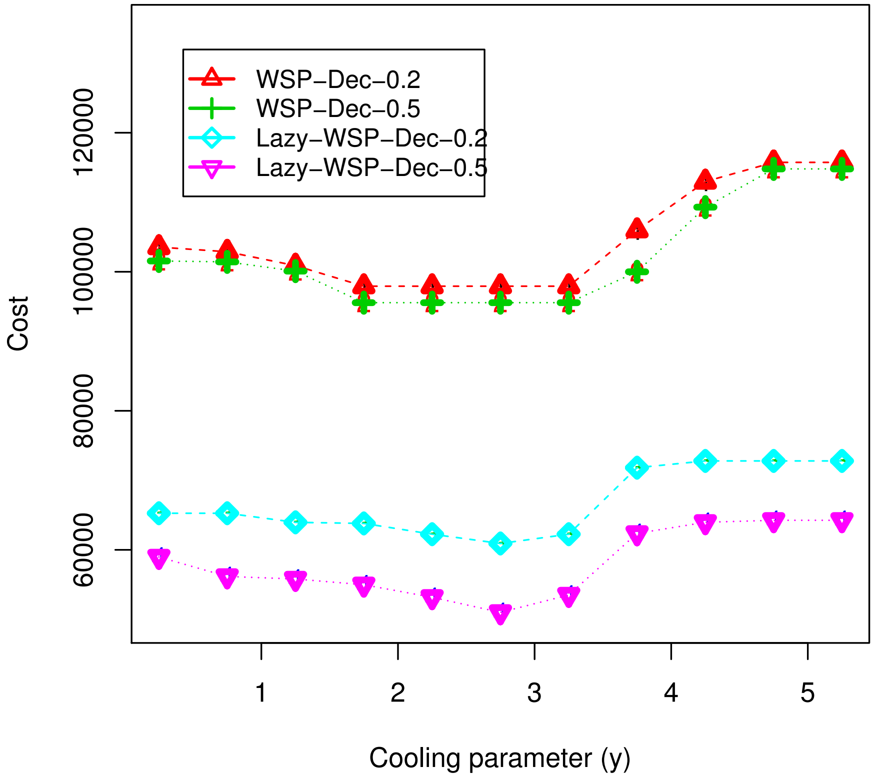

Figure 6.13 Cost vs. cooling parameter: average cost of unsatisfied clauses (smaller is better) against different values of cooling parameter y of WSP-Dec algorithm for Yeast.

different cooling parameter's values ($y \in [0.25, 5.25]$) of the WSP-Dec algorithm and the Lazy-WSP-Dec algorithm at thresholds 0.2, 0.5 for 50 objects in the domains of the three underlying datasets. The result shows that the WSP-Dec algorithm reaches a slope-and-plateau region where its quality of solution increases and then starts to decease. In the three tested datasets, this slope-and-plateau region occurred when the cooling y parameter's value is approximately between 1.5 and 3 (*answering the part of Q3 related to cooling parameter*).

Experiment III

Table 6.2 The percentage of the frozen ground atoms (i.e., cluster backbones) that are fixed (fixed%) and the average cost of unsatisfied clauses (Cost) for a family of WSP-Dec at different choices of smoothing pairs (χ,γ) on Cora, Web-KB, and Yeast. The cooling parameter y assigned a value 2 and the threshold takes a value 0.2.

		WSP-Dec algorithms							
		$\chi = 1$, $\gamma = 0$		$\chi = 0.5$, $\gamma = 0.5$		$\chi = 0.25$, $\gamma = 0.75$		$\chi = 0$, $\gamma = 1$	
Datasets	No. Obj.	fixed%	Cost	fixed%	Cost	fixed%	Cost	fixed%	Cost
Cora	50	43.1	3.2×10^3	30.8	4.4×10^3	21.1	5.4×10^3	6.4	6.6×10^3
	250	40.5	7.2×10^5	29.7	9.2×10^5	18.8	1.2×10^6	5.6	1.5×10^6
	1000	51.7	4.7×10^7	36.5	6.1×10^7	24.3	7.9×10^7	10	9.7×10^7
Web-KB	50	35.5	8.0×10^3	26.2	1.0×10^4	16.0	1.4×10^4	7.0	1.6×10^4
	250	41.0	9.0×10^6	30.0	1.2×10^7	17.9	1.5×10^7	8.3	1.9×10^7
	1000	48.2	1.1×10^{10}	35.3	1.4×10^{10}	22.5	1.8×10^{10}	9.9	2.3×10^{10}
Yeast	50	47.0	9.8×10^4	34.0	1.3×10^5	21.6	1.8×10^5	9.0	2.6×10^5
	250	38.4	1.0×10^8	28.9	1.2×10^8	16.7	1.6×10^8	7.8	2.1×10^8
	1000	30.5	1.1×10^{11}	22.6	1.3×10^{11}	13.4	1.7×10^{11}	4.3	2.1×10^{11}

Table 6.2 records both the average cost of unsatisfied clauses and the percentage of the frozen ground atoms that are fixed by the WSP-Dec family of algorithms at four pairs of settings of (χ,γ) on Cora, Web-KB, and Yeast. Overall, the most successful pair of WSP-Dec algorithm was ($\chi = 1$,$\gamma = 0$) in all tested datasets. For this setting, the decimation step fixed approximately $30\% - 51\%$ of the query ground atoms (i.e., it obtains a portion that is $30 - 51\%$ of the optimal MAP solution). By contrast, the poorest pair setting was ($\chi = 0$, $\gamma = 1$), the decimation step of the WSP-Dec algorithm fixed at most 10% of the query ground atoms, reaching to small portions of the MAP solutions. In addition, the algorithm with the

pair ($\chi = 0.5$, $\gamma = 0.5$) was more accurate than the pair ($\chi = 0.25$, $\gamma = 0.75$) in terms of both quality of MAP solution, and it has a larger amount of fixed frozen atoms (*conclusive answer to Q4*).

6.5 Discussion

Overall the results clearly show that WSP-χ based algorithms substantially improve the accuracy and scalability of the propositional MaxWalkSAT algorithm for MAP inference. This is due to, first, finding the frozen ground atoms, which provides a large portion of the optimal MAP solution. Second, fixing the frozen atoms simplifies the MAP inference task into another one that can be solved accurately using any conventional MAP inference algorithm.

WSP-Dec algorithms were also very competitive with respect to Lazy MAP Inference whenever a substantial amount of frozen atoms were obtained. This can be attributed to the fact that fixing the frozen atoms enlarges the evidence database and shrinks the query set which provides great implications for reducing the effective size of the grounded network. Moreover, the WSP-Dec algorithm dominated both propositional MaxWalkSAT and Lazy-MaxWalkSAT on all tested data sets when it combined with Lazy Inference. This is because the result of such combination is the exploitation of both sparseness in Lazy and frozen atoms from WSP to scale up the MAP inference.

The magnetization threshold $\mathcal{T}$ offers a trade-off for WSP-Dec algorithm in terms of the amount of frozen atoms/the quality of obtained MAP solution: on one hand decreasing $\mathcal{T}$'s value enables one to obtain a large amount of frozen ground atoms and therefore improve the scalability, but on the other hand some of those frozen atoms could be inaccurate and that can effect the quality of the final obtained MAP solution. Thus, in the presence of a huge grounded network, one can choose to slightly sacrifice the quality of the solution by decreasing $\mathcal{T}$ just to enable the WSP-Dec algorithm to find a MAP solution. For instance, with 1000 objects in the domain of Yeast dataset - on which its MLN features diversity in the weights) - WSP-Dec 0.5 has difficulties reaching a MAP solution, whereas WSP-Dec 0.2 algorithm can find one.

The results in Experiment II show that the behaviour of the WSP-Dec algorithm over varying cooling parameter y is consistent with Theorem 3, ensuring that as long as WSP-χ converges, its performance improves as cooling parameter y increases. That is to say, the marginals computed by the WSP-Dec algorithm will prefer MAP solutions that satisfy the clauses with a maximum total weight. However, at a certain point of increasing y, WSP can fail to converge to accurate results which may be attributed to overshooting the optimal solution. Also, choosing a small value of y can slow the convergence. Experimentally, we find that we need to take it to a sufficiently large value between 1.5 and 3 to obtain highly convergence to an accurate result. This values of y is very close to the work of Battaglia et al. (2004), who found that $y = 2.5$ is one best value for SP-y algorithm. Although one can use a bisection method to numerically obtain the ideal cooling value y beforehand, we believe that the value of y can be dynamically tuned to favor the convergence of the WSP-Dec algorithm, which will be an interesting analysis for future research.

Furthermore, the performance of WSP-χ algorithms on the extended factor graph significantly effected by the choice of smoothing parameter χ, and this appears clearly in the results of Table 6.2. That is to say, given the pool of WSP-χ algorithms, increasing the value of χ (or equivalently decreasing the value of γ) allows the algorithm to obtain more frozen ground atoms, which results in enlarging the evidence database and shrinking the query set, and therefore improving the scalability besides finding a larger portion for the optimal MAP solution. Thus, when setting χ to the most (i.e., $\chi = 1$ and $\gamma = 0$), a call to MaxWalkSAT might in fact not be needed or only needed to solve an easy MAP inference on a scalable grounded network. On the contrary, decreasing χ to the most (i.e., $\chi = 0$ and $\gamma = 1$) reduces WSP-χ to traditional BP, and this makes the calling of MaxWalkSAT often faces an hard MAP inference on a simplified grounded network that is very close to the propositional one. In addition, the success of pure WSP-1 for finding the most accurate results, supports the conjecture that MAP solutions of relational problems typically do possess non-trivial max-cores for large structured formulas.

CHAPTER 7 CONCLUSION AND FUTURE WORK

In this thesis, we have proposed at least three major contributions. Below, we revisit those contributions briefly, and then sketch our thoughts about future research directions.

7.1 GEM-MP Inference Approach

In Chapter 4, 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 this work lies in the creation 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 fixed point 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 derives as 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 potency 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.

7.2 Preference Relaxation Scaling Strategy

In Chapter 5, we proposed Preference Relaxation, a two-stage strategy that exploits determinism to scale up relational inference. Preferences are first ignored in order to shrink the query set and enlarge the evidence database. The novelty here appears in exploiting determinism again, but this time for scaling inference. Experiments on real-world domains show that PR is able to greatly reduce the time and space requirements of inference by several orders of magnitude when applied to propositional MC-SAT and is twice as fast as its lazy version.

7.3 WSP-χ Family of Algorithms

It is widely known that many real-world problems can be formulated using expressive SRL models that feature logical structures with very high densities, and this renders them identical to hard satisfiability instances. We believe a clear gap in the present literature on MAP inference in SRL exists with respect to taking into consideration the fact that the solution space of such problems is frequently clustered when the density of the underlying model is high or close to a critical threshold. Ignoring the clustering that occurs in the solution space can engender intricacies of getting stuck in a metastable cluster at local optimum when handling the inference using some state-of-the-art techniques like local search, max-product message-passing or LP relaxation based algorithms. The novelty of the work, presented in Chapter 6, is two-fold. First, we present a new family of extended factor graphs WSP-χ associated with a family of Weighted Survey Propagation algorithms applicable to SRL models. The objective of WSP-χ is to identify the backbones of a cluster containing potentially optimal MAP solutions. This introduces the WSP-χ family as a set of pre-processing methods that can help in finding the optimal solution in the presence of clustered solution spaces. Second, we propose lazy variants of the WSP-χ family of algorithms to improve scalability for MAP inference. Using real-world domains such as protein interaction, hyperlink analysis and entity resolution, we have experimentally shown that WSP-χ and its lazy variants are able to greatly improve quality and scalability of MAP inference when integrated with propositional MaxWalkSAT and its lazy version. To conclude, the approach of WSP-χ represents an improvement in relational MAP inference: By obtaining cluster backbones that determine which particular clusters contain potentially optimal solutions, not only is a large portion of the optimal solution provided in the cluster, but also fixing them helps getting inside the cluster by enlarging the evidence database and shrinking the query set. This therefore reduces the graph network into a scalable one that can be solved accurately using any conventional

MAP inference method.

7.4 Note on some of the thesis's applications

With the use of the proposed approaches in this thesis, we now have the ability to perform inference and data mining tasks in much larger relational models in an accurate and efficient way, even if these models are difficult to solve (e.g., because of the presence of large amounts of determinism or hard constraints and cycles). As a concrete example, in the entity resolution application, we can integrate both PR and GEM-MP to de-duplicate more than three billion citations (say, for example the citations of the articles in the engineering field that are published in the last 10 years from universities in Europe and Asia). Another application is protein interactions, we can now determine accurately the interactions among more than two billion genes and proteins (e.g., breast cancer proteins of women living in North America and Africa), specifying if they have similar functions and structures. This puts other similar large scale, real-world (scientific and industrial) applications within reach. Specific examples include many information extraction applications (Poon and Domingos, 2007), the mining of knowledge-sharing sites for Viral marketing (Richardson *et al.*, 2003), and certain mobile robot mapping applications (Wang and Domingos, 2008).

7.5 Future Work

The research in this thesis points to a number of promising directions for future work. In particular, some issues and directions still warrant consideration, including:

- **Evaluation of the use of GEM-MP as an inference subroutine for learning.** It is well known that there is a strong interaction between learning algorithms, which estimate the parameters of a model from data, and inference algorithms which use a model to make predictions about data. This makes the choice of an efficient and accurate inference technique during learning often has a great influence on the resulting model. In the world of SRL, exact inference is intractable. Instead, we can perform approximate inference using Markov chain Monte Carlo, and in particular Gibbs sampling. However, when we have deterministic dependencies in the model, the ergodicity breaks down in Gibbs sampling. This remains true even for more sophisticated alternatives like simulated tempering and MC-SAT. This in fact motivates the use of our GEM-MP — which has the ability to frequently converge to accurate results in the presence of cycles and determinism — as a much more efficient alternative.

- **Investigation of the lifted (cf. Ahmadi *et al.*, 2013; Singla *et al.*, 2010) and the lazy (cf. Poon *et al.*, 2008) versions of GEM-MP to enhance its scalability.** At a high level, GEM-MP can be seen as akin to message-passing inference methods. The presence of some lifting message-passing techniques like lifted BP — which has been implemented in some publicly available software like alchemy 2.0 (Kok *et al.*, 2007) — makes the derivation of lifted GEM-MP is straightforwardly doable.

- **The possibility of increasing the accuracy of GEM-MP** by re-deriving new update rules that apply a global approximation for the posterior distribution $q(\mathcal{Y}; \mathcal{T}_{\mathcal{Y}})$ in the $M_{q(\mathcal{Y})}$-step of GEM-MP.

- **The application of PR to other inference algorithms such as Lifted Inference.** One can combine PR with lifted BP to obtain Lifted-PR-BP, which maybe differ from PR-BP (see subsection 5.2.1) with starting by relaxing soft factors and construct the *lifted factor graph* for factor nodes and variable nodes that represent the awake ground hard clauses and their awake ground atoms, respectively.

- **The combination of WSP-χ with other inference algorithms such as Lifted Inference.** One can obtain a Lifted-WSP-χ by lifting the re-parameterized extended factor graph (see section 6.1) and then slightly modifying WSP-χ to run over it.

- **Derive an online MAP inference scenario for WSP-χ.** In some applications like object tracking (Song *et al.*, 2013) and the development of markets (Domingos and Richardson, 2001), it is often necessary to incorporate time for systems that change dynamically and information from past states can be carried over by means of the dynamic MLNs. In such cases, the MLNs have been frequently applied in an offline mode, where the inference is done over a defined time frame. For real applications this solution is often insufficient and a method is needed that can run online and in real-time.

We also plan to:

- **Use WSP-Dec for solving CSPs**, since a decimation combined with WSP-χ-based message-passing can be viewed as a depth-first search combined with a “highest bias” heuristic.

- **Try to dynamically tune the smoothing and cooling parameters for WSP-χ** to favor a better convergence for the WSP-Dec algorithm.

- **Apply WSP-χ to the exact parallelized integer linear programming (ILP) solver Gurobi and approximate solver Tuffy**

- **Perform an on-line inference scenario of PR-based algorithms.**

- **Try to combine WSP-χ with the most recent variants of MaxWalkSAT.**

- **Evaluate the use of PR, GEM-MP and WSP-χ for solving inference on small datasets that are at the limit of what recent exact solvers (Vlasselaer *et al.*, 2015) can handle, and then compare the accuracy and scalability with the state-of-the-art inference algorithms.**

- **Improve MC-SAT by using the more recent uniform sampling methods (Chakraborty *et al.*, 2014a,b) instead of SampleSAT.**

REFERENCES

Achlioptas, Dimitris and Ricci-Tersenghi, Federico (2009). Random formulas have frozen variables. *SIAM Journal on Computing*, *39*(1), SIAM, 260–280.

Ahmadi, Babak and Kersting, Kristian and Mladenov, Martin and Natarajan, Sriraam (2013). Exploiting symmetries for scaling loopy belief propagation and relational training. *Machine learning*, *92*(1), 91–132.

Allen, David and Darwiche, Adnan (2003). New advances in inference by recursive conditioning. *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann Publishers Inc., 2–10.

Altarelli, Fabrizio and Monasson, Remi and Semerjian, Guilhem and Zamponi, Francesco (2009). A review of the statistical mechanics approach to random optimization problems. *Handbook of Satisfiability, volume 185 of the Series "Frontiers in Artificial Intelligence and Applications", IOS Press.*

Andrieu, Christophe and De Freitas, Nando and Doucet, Arnaud and Jordan, Michael I (2003). An introduction to mcmc for machine learning. *Machine learning*, *50*(1-2), 5–43.

Battaglia, Demian and Kolář, Michal and Zecchina, Riccardo (2004). Minimizing energy below the glass thresholds. *Physical Review E*, *70*, 36107–36118.

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

Besag, Julian (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 259–302.

Biroli, Giulio and Cocco, Simona and Monasson, Rémi (2002). Phase transitions and complexity in computer science: an overview of the statistical physics approach to the random satisfiability problem. *Physica A: Statistical Mechanics and its Applications*, *306*, 381–394.

Braunstein, A. and Mézard, M. and Zecchina, R. (2005). Survey propagation: An algorithm for satisfiability. *Random Structures and Algorithms*, *27*(2), 201–226.

Braunstein, Alfredo and Zecchina, Riccardo (2004). Survey and belief propagation on random k-sat. *Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing, Vancouver (BC), Canada.* Springer, vol. 2919, 519–528.

Bui, Hung B and Huynh, Tuyen N and de Salvo Braz, Rodrigo (2012). Exact lifted inference with distinct soft evidence on every object. *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, Toronto, Ontario, Canada.* AAAI Press, 1875–1881.

Chakraborty, Supratik and Fremont, Daniel J and Meel, Kuldeep S and Seshia, Sanjit A and Vardi, Moshe Y (2014a). Distribution-aware sampling and weighted model counting for sat. *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, Québec, Canada, July 27 –31.* AAAI Press, 1722–1730.

Chakraborty, Shiladri and Meel, Kuldeep S and Vardi, Moshe Y (2014b). Balancing scalability and uniformity in sat witness generator. *Proceedings of 51st Design Automation Conference (DAC), Austin, Texas, June 5-9.* IEEE, 1–6.

Chavas, Joël and Furtlehner, Cyril and Mézard, Marc and Zecchina, Riccardo (2005). Survey-propagation decimation through distributed local computations. *Journal of Statistical Mechanics: Theory and Experiment*, *2005*(11), IOP Publishing, 11016–11027.

Chieu, Hai Leong and Lee, Wee Sun (2009). Relaxed survey propagation for the weighted maximum satisfiability problem. *Journal of Artificial Intelligence Research (JAIR)*, *36*, 229–266.

Chieu, Hai L and Lee, Wee S and Teh, Yee W (2007). Cooled and relaxed survey propagation for mrfs. *Proceedings of the 21st Annual Conference on Neural Information Processing Systems: Advances in Neural Information Processing Systems 20, Vancouver, British Columbia, Canada.* Curran Associates, Inc., 297–304.

Dauwels, Justin and Korl, Sascha and Loeliger, Hans-Andrea (2005). Expectation maximization as message passing. *Proceedings of IEEE International Symposium on Information Theory (ISIT 2005), Adelaide Convention Centre Adelaide, Australia.* IEEE computer society, 583–586.

Davis, Jesse and Domingos, Pedro (2009). Deep transfer via second-order markov logic. *Proceedings of the 26th International Conference on Machine Learning (ICML-09).* Montreal, Canada.

De Salvo Braz, Rodrigo and Amir, Eyal and Roth, Dan (2005). Lifted first-order probabilistic inference. *Proceedings of the 19th International joint conference in artificial intelligent.* AAAI Press, 1319–1325.

De Salvo Braz, Rodrigo and Amir, Eyal and Roth, Dan (2006). Mpe and partial inversion in lifted probabilistic variable elimination. *Proceedings Of The Twenty-first National Conference On Artificial Intelligence, July 16–20, 2006, Boston, Massachusetts.* AAAI press, vol. 6, 1123–1130.

De Salvo Braz, Rodrigo and Natarajan, Sriraam and Bui, Hung and Shavlik, Jude and Russell, Stuart (2009). Anytime lifted belief propagation. *Proceedings of 6th International Workshop on Statistical Relational Learning, Leuven, Belgium.* vol. 9, 1–3.

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

Domingos, Pedro and Richardson, Matt (2001). Mining the network value of customers. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, San Francisco, CA, USA.* ACM, 57–66.

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

Forney, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, *61*(3), IEEE computer society, 268–278.

Frey, Brendan J and MacKay, David JC (1998). A revolution: Belief propagation in graphs with cycles. *Advances in neural information processing systems*, Morgan Kaufmann, 479–485.

Friedgut, Ehud (2005). Hunting for sharp thresholds. *Random Structures & Algorithms*, *26*(1-2), 37–51.

Friedgut, Ehud and Bourgain, Jean and others (1999). Sharp thresholds of graph properties, and the k-sat problem. *Journal of the American mathematical Society*, *12*(4), 1017–1054.

Geman, Stuart and Geman, Donald (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6), IEEE computer society, 721–741.

Getoor, Lise and Taskar, Ben (2007). *Introduction to Statistical Relational Learning: Adaptive Computation and Machine Learning*. The MIT Press.

Globerson, Amir and Jaakkola, Tommi (2007). Convergent propagation algorithms via oriented trees. R. Parr and L. C. van der Gaag, editors, *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22*. AUAI Press, 133–140.

Gogate, Vibhav and Domingos, Pedro (2011). Probabilistic theorem proving. *Proceedings of the Twenty-Seventh Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-11)*. AUAI Press, Corvallis, Oregon, 256–265.

Gogate, Vibhav and Jha, Abhay Kumar and Venugopal, Deepak (2012). Advances in lifted importance sampling. *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press, 1910–1916.

Gomes, Carla and Hogg, Tad and Walsh, Toby and Zhang, Weixiong (2002). Tutorial - phase transitions and structure in combinatorial problems. *Proceedings Of The Eighteenth National Conference On Artificial Intelligence, Edmonton, Canada*. AAAI Press.

Granville, Vincent and Kriválek, Mirko and Rasson, J-P (1994). Simulated annealing: A proof of convergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(6), IEEE computer society, 652–656.

Hartmann, Alexander K and Weigt, Martin (2006). *Phase transitions in combinatorial optimization problems: basics, algorithms and statistical mechanics*. John Wiley & Sons.

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

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

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

Heskes, Tom (2004). On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, *16*(11), MIT Press, 2379–2413.

Hoeve, Willem Jan van and Pesant, Gilles and Rousseau, Louis-Martin (2006). On global warming: Flow-based soft global constraints. *Journal of Heuristics*, *12*(4-5), Springer, 347–373.

Hopcroft, John E. and Motwani, Rajeev and Ullman, Jeffrey D. (2006). *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Horsch, Michael C and Havens, William S (2000). Probabilistic arc consistency: A connection between constraint reasoning and probabilistic reasoning. *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 282–290.

Hsu, Eric I and Kitching, Matthew and Bacchus, Fahiem and McIlraith, Sheila A (2007). Using expectation maximization to find likely assignments for solving csp's. *Proceedings of 22nd National Conference on Artificial Intelligence (AAAI '07), Vancouver, Canada*. AAAI Press, vol. 22, 224–232.

Hsu, Eric I. and Muise, Christian and Beck, J. Christopher and McIlraith, Sheila A. (2008). Probabilistically estimating backbones and variable bias. *Proceedings of 14th International Conference on Principles and Practice of Constraint Programming (CP '08), Sydney, Australia*. Springer, 613–617.

Huynh, Tuyen N and Mooney, Raymond J (2009). Max-margin weight learning for markov logic networks. *Machine Learning and Knowledge Discovery in Databases*, Springer, vol. 5781. 564–579.

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

Ibrahim, Mohamed-Hamza and Pal, Christopher and Pesant, Gilles (2015). Exploiting determinism to scale relational inference. *Proceedings of the Twenty-Ninth National Conference on Artificial Intelligence (AAAI'15), January 25 –30, 2015, Austin, Texas, USA*. AAAI Press, 1756–1762.

Kambhampati, Soumya C and Liu, Thomas (2013). Phase transition and network structure in realistic sat problems. *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14–18, 2013 in Bellevue, Washington, USA*. AAAI Press, 1619–1620.

Kautz, Henry and Selman, Bart and Jiang, Yueyen (1997). A general stochastic approach to solving problems with hard and soft constraints. *The Satisfiability Problem: Theory and Applications*, *17*, 573–586.

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

Kersting, Kristian and Ahmadi, Babak and Natarajan, Sriraam (2009). Counting belief propagation. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence.* AUAI Press, 277–284.

Khosla, Megha and Melhorn, Kurt and Panagiotou, Konstantinos (2009). *Message Passing Algorithms.* PhD Thesis, Citeseer.

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

Kilby, Philip and Slaney, John and Thiébaux, Sylvie and Walsh, Toby (2005). Backbones and backdoors in satisfiability. *Proceedings of the The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, Pittsburgh, Pennsylvania, USA*. AAAI Press, vol. 5, 1368–1373.

Kok, Stanley and Singla, Parag and Richardson, Matthew and Domingos, Pedro and Sumner, Marc and Poon, Hoifung and Lowd, Daniel (2007). *The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering,University of Washington, Seattle, WA.* http://alchemy.cs.washington.edu.

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

Kolmogorov, Vladimir (2006). Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(10), 1568–1583.

Kroc, Lukas and Sabharwal, Ashish and Selman, Bart (2007). Survey propagation revisited. *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22.* AUAI Press, 217–226.

Kroc, Lukas and Sabharwal, Ashish and Selman, Bart (2008). Counting solution clusters in graph coloring problems using belief propagation. *Proceedings of 22nd Conference on Neural Information Processing Systems: Advances in Neural Information Processing Systems 21, Vancouver, British Columbia, Canada.* Curran Associates Inc., 873–880.

Kroc, Lukas and Sabharwal, Ashish and Selman, Bart (2009). Message-passing and local heuristics as decimation strategies for satisfiability. *Proceedings of the 2009 ACM symposium on Applied Computing.* ACM, 1408–1414.

Kschischang, Frank and Member, Senior and Frey, Brendan J. and Loeliger, Hans-andrea (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, *47*, IEEE computer society, 498–519.

Kumar, M Pawan and Torr, Philip HS (2008). Efficiently solving convex relaxations for map estimation. *Proceedings of the 25th international conference on Machine learning, Helsinki, Finland, July 5-9.* ACM, 680–687.

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

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

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

Maneva, Elitza and Mossel, Elchanan and Wainwright, Martin J. (2007). A new look at survey propagation and its generalizations. *Journal of the ACM (JACM)*, *54*(4), ACM, 17–21.

Mann, Alexander and Hartmann, AK (2010). Numerical solution-space analysis of satisfiability problems. *Physical Review E, 82*(5), APS, 056702–56707.

Marinescu, Radu and Dechter, Rina (2005). Advances in and/or branch-and-bound search for constraint optimization. *Proceedings of the 7th International Workshop on Preferences and Soft Constraints of the Eleventh International Conference on Principles and Practice of Constraint Programming, October 1, 2005 Melia Sitges Hotel, Sitges, Spain*. Springer, 1457–1491.

Mateescu, Robert and Kask, Kalev and Gogate, Vibhav and Dechter, Rina (2010). Join-graph propagation algorithms. *Journal of Artificial Intelligence Research (JAIR)*, *37*, 279–328.

Robert J. Mceliece and David J. C. Mackay and Jung-fu Cheng (1998). Turbo decoding as an instance of pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communications*, *16*, IEEE computer society, 140–152.

Meltzer, Talya and Globerson, Amir and Weiss, Yair (2009). Convergent message passing algorithms - a unifying view. J. Bilmes and A. Y. Ng, editors, *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21*. AUAI Press, 393–401.

Milch, Brian and Zettlemoyer, Luke S and Kersting, Kristian and Haimes, Michael and Kaelbling, Leslie Pack (2008). Lifted probabilistic inference with counting formulas. *Proceedings of the Twenty Third Conference on Artificial Intelligence, Chicago, Illinois, USA*. AAAI Press, vol. 8, 1062–1068.

Montanari, Andrea and Parisi, Giorgio and Ricci-Tersenghi, Federico (2004). Instability of one-step replica-symmetry-broken phase in satisfiability problems. *Journal of Physics A: Mathematical and General*, *37*(6), IOP Publishing, 2073–2079.

Mooij, Joris M. and Kappen, Hilbert J. (2005). Sufficient conditions for convergence of loopy belief propagation. *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*. AUAI Press, 396–403.

Murphy, Kevin and Weiss, Yair and Jordan, Michael (1999). Loopy belief propagation for approximate inference: An empirical study. *Proceedings of the Fifteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-99), Stockholm, Sweden*. Morgan Kaufmann, 467–476.

Neal, Radford M. and Hinton, Geoffrey E. (1999). *Learning in Graphical Models*. MIT Press, chapter *A View of the EM Algorithm That Justifies Incremental, Sparse, and Other Variants*. 355–368.

Nguyen, XuanLong and Wainwright, Martin J and Jordan, Michael I (2004). Decentralized detection and classification using kernel methods. *Proceedings of the twenty-first international conference on Machine learning, (ICML), Banff, Canada.* ACM, vol. 69, 80–88.

Papai, Tivadar and Kautz, Henry A. and Stefankovic, Daniel (2012). Slice normalized dynamic markov logic networks. *Proceedings of 26th Conference on Neural Information Processing Systems. Annual Conference: Advances In Neural Information Processing Systems 25.* Curran Associates Inc., 1916–1924.

Papai, Tivadar and Singla, Parag and Kautz, Henry (2011). Constraint propagation for efficient inference in markov logic. *Proceedings of 17th International Conference on Principles and Practice of Constraint Programming (CP 2011), Perugia, Italy, 12-16 September.* springer publishing, 691–705.

Park, James D (2002). Using weighted max-sat engines to solve mpe. *Proceedings of the Eighteenth National Conference on Artificial Intelligence, Menlo Park, CA, USA.* AAAI Press, 682–687.

Parkes, Andrew J (1997). Clustering at the phase transition. *Proceedings of the 14th National Conference on Artificial Intelligence, July 27–31, at the convention center in Providence, Rhode Island.* AAAI Press, 340–345.

Pearl, Judea (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Poole, David (2003). First-order probabilistic inference. *Proceedings of the 18th International Joint Conference on Artificial Intelligence IJCAI'03, Acapulco, Mexico.* Morgan Kaufmann Publishers Inc., vol. 3, 985–991.

Poon, Hoifung and Domingos, Pedro (2006). Sound and efficient inference with probabilistic and deterministic dependencies. *Proceedings of the 21st national conference on Artificial intelligence, Vol.(1).* AAAI Press, AAAI'06, 458–463.

Poon, Hoifung and Domingos, Pedro (2007). Joint inference in information extraction. *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07), Vancouver, British Columbia, July 22–26.* vol. 7, 913–918.

Poon, Hoifung and Domingos, Pedro and Sumner, Marc (2008). A general method for reducing the complexity of relational inference and its application to mcmc. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, Chicago, Illinois, July 13–17.* AAAI Press, 1075–1080.

Potetz, Brian (2007). Efficient belief propagation for vision using linear constraint nodes. *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07, Minneapolis, MN, USA*. IEEE computer society, 1–8.

Ravikumar, Pradeep and Lafferty, John (2006). Quadratic programming relaxations for metric labeling and markov random field map estimation. *Proceedings of the 23rd international conference on Machine learning*. ACM, 737–744.

Richardson, Matthew and Agrawal, Rakesh and Domingos, Pedro (2003). Trust management for the semantic web. *Proceedings of the Second International Semantic Web Conference (ISWC2003)*. Springer, 351–368.

Richardson, Matthew and Domingos, Pedro (2006). Markov logic networks. *Machine Learning*, *62*(1-2), Kluwer Academic Publishers, 107–136.

Riedel, Sebastian (2008). Improving the accuracy and efficiency of map inference for markov logic. *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI 2008), 9-12 July, Helsinki, Finland*. AUAI Press, 468–475.

Roosta, Tanya and Wainwright, Martin J. and Sastry, Shankar S. (2008). Convergence analysis of reweighted sum-product algorithms. *IEEE Transactions on Signal Processing*, *56*(9), IEEE computer society, 4293–4305.

Rossi, Francesca and Van Beek, Peter and Walsh, Toby (2006). *Handbook of constraint programming*. Elsevier.

Sarkhel, Somdeb and Gogate, Vibhav (2013). Lifting walksat-based local search algorithms for map inference. *Proceedings of Statistical Relational Artificial Intelligence Workshop at the Twenty-Seventh AAAI Conference on Artificial Intelligence, Bellevue, Washington, USA*. AAAI Press, 64–67.

Sarkhel, Somdeb and Venugopal, Deepak and Singla, Parag and Gogate, Vibhav (2014). Lifted MAP inference for markov logic networks. *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, Reykjavik, Iceland*. JMLR: W & CP, vol. 33, 859–867.

Saul, Lawrence K. and Jaakkola, Tommi and Jordan, Michael I. (1996). Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research, AAAI Press, 4*(1), 61–76.

Selman, Bart and Kautz, Henry and Cohen, Bram and others (1993). Local search strategies for satisfiability testing. *Cliques, coloring, and satisfiability: Second DIMACS implementation challenge*, *26*, 521–532.

Semerjian, Guilhem and Monasson, Rémi (2003). Relaxation and metastability in a local search procedure for the random satisfiability problem. *Physical Review E*, *APS, 67*(6), 066103–66109.

Sen, Prithviraj and Deshpande, Amol and Getoor, Lise (2009). Bisimulation-based approximate lifted inference. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, Canada on June 18-21.* AUAI Press, 496–505.

Shavlik, Jude and Natarajan, Sriraam (2009). Speeding up inference in markov logic networks by preprocessing to reduce the size of the resulting grounded network. *Proceedings of the 21 International Joint Conference on Artificial Intelligence, Pasadena, California, USA.* Morgan Kaufmann Publishers Inc., 1951–1956.

Shi, Xiangqiong and Schonfeld, Dan and Tuninetti, Daniela (2010). Message error analysis of loopy belief propagation. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, 14-19 March, Dallas, Texas, USA.* IEEE computer society, 2078–2081.

Singla, Parag (2012). Markov logic networks: theory, algorithms and applications. *Proceedings of the 18th International Conference on Management of Data.* Computer Society of India, 15–150.

Singla, Parag and Domingos, Pedro (2006a). Entity resolution with markov logic. *ICDM.* IEEE Computer Society, 572–582.

Singla, Parag and Domingos, Pedro (2006b). Memory-efficient inference in relational domains. *Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06), Boston, Massachusetts, July 16–20.* AAAI Press, vol. 6, 488–493.

Singla, Parag and Domingos, Pedro (2008). Lifted first-order belief propagation. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, Chicago, Illinois, July 13–17.* AAAI Press, 1094–1099.

Singla, Parag and Nath, Aniruddh and Domingos, Pedro (2010). Approximate lifted belief propagation. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, Georgia, USA, July 11–15, 2010.* AAAI Press, 92–97.

Slaney, John and Walsh, Toby (2001). Backbones in optimization and approximation. *Proceedings of the 17th International Joint Conference on Artificial Intelligence, Seattle, WA, USA*. Morgan Kaufmann Publishers Inc., vol. 1, 254–259.

Smith, David and Gogate, Vibhav (2014). Loopy belief propagation in the presence of determinism. *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, April 22-25, Reykjavik, Iceland.* JMLR: W & CP, vol. 33, 895–903.

Song, Young Chol and Kautz, Henry and Allen, James and Swift, Mary and Li, Yuncheng and Luo, Jiebo and Zhang, Ce (2013). A markov logic framework for recognizing complex events from multimodal data. *Proceedings of the 15th ACM on International conference on multimodal interaction.* ACM, 141–148.

Szeliski, Richard (2006). Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, *2*(1), Now Publishers Inc., 1–104.

Van den Broeck, Guy and Taghipour, Nima and Meert, Wannes and Davis, Jesse and De Raedt, Luc (2011). Lifted probabilistic inference by first-order knowledge compilation. *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence, Barcelona, Catalonia, Spain, 16–22 July.* AAAI Press, 2178–2185.

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

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

Vlasselaer, Jonas and Van den Broeck, Guy and Kimmig, Angelika and Meert, Wannes and De Raedt, Luc (2015). Anytime inference in probabilistic logic programs with Tp-compilation. *Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI).* AAAI Press, 1852–1858.

Wainwright, Martin and Jaakkola, Tommi and Willsky, Alan (2003). Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, *49*(5), IEEE computer society, 1120–1146.

Wainwright, Martin and Jaakkola, Tommi and Willsky, Alan (2004). Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing*, *14*(2), Springer, 143–166.

Wainwright, Martin and Jaakkola, Tommi and Willsky, Alan (2005). MAP estimation via agreement on (hyper)trees: Message-passing and linear programming approaches. *IEEE Transactions on Information Theory*, *51*, IEEE computer society, 3697–3717.

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

Wang, Jue and Domingos, Pedro (2008). Hybrid markov logic networks. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, Chicago, Illinois, July 13–17*. vol. 8, 1106–1111.

Wei, Wei and Erenrich, Jordan and Selman, Bart (2004). Towards efficient sampling: Exploiting random walk strategies. *Proceedings of the Nineteenth National Conference On Artificial Intelligence, July 25–29, 2004, San Jose, California*. AAAI Press, vol. 4, 670–676.

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

Weiss, Yair and Freeman, William T (2001). On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, *47*(2), IEEE computer society, 736–744.

Winn, John Michael (2004). *Variational message passing and its applications*. PhD Thesis, University of Cambridge.

Winn, John M. and Bishop, Christopher M. (2005). Variational message passing. *Journal of Machine Learning Research*, *6*, JMLR. org, 661–694.

Yanover, Chen and Meltzer, Talya and Weiss, Yair (2006). Linear programming relaxations and belief propagation–an empirical study. *Journal of Machine Learning Research*, *7*, JMLR. org, 1887–1907.

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

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

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

Yuille, Alan L. (2001). A double-loop algorithm to minimize the bethe free energy. *Proceedings of the Third International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer-Verlag, 3–18.

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

Zhang, Weixiong (2004). Phase transitions and backbones of the asymmetric traveling salesman problem. *Journal of Artificial Intelligence Research (JAIR)*, *21*, AAAI Press, 471–497.

APPENDIX A Proofs of Theorems and Propositions

Proof of Theorem 2

Theorem 2. The underlying joint distribution defined by the extended factor graph, $\hat{\mathcal{G}}$ with $\chi = 1$, is positive only over valid max-cores.

Proof. The complete assignment X in extended factor graph $\hat{\mathcal{G}}$ that is not max-core will be either invalid or will involve unconstrained ground atoms set to value 1 or 0. For invalid complete assignments, the distribution is zero because of the definition of ξ in Eq. (6.3). Additionally, for complete assignments with unconstrained ground atoms set to value 1 or 0 the distribution will be zero because of the definition of φ_j in Eq. (6.9). Thus, from Eq. (6.6), for each valid max-core X we have a joint probability of the form:

$$p(X) \propto \prod_{f_i \in S(X)} e^{\hat{w}_i \cdot y} \tag{A.1}$$

where $S(X)$ represents only the set of the ground clauses satisfied by X. This means that the joint probability is always positive for valid max-core X. □

Proof of Theorem 3

Theorem 3. When $y \to \infty$, then WSP-1 estimates marginals corresponding to the stationary point of the Bethe free energy on a uniform distribution over max-cores.

Proof. Assume that we have a max-core (i.e., $\mathcal{W}$-core) of total weight $\mathcal{W}$, and a more optimal one (($\mathcal{W} + \epsilon$)-core) of larger weight $\mathcal{W} + \epsilon$. According to Eq. (A.1), the ratio between the two max-cores is:

$$\frac{P(\mathcal{W} + \epsilon\text{-core})}{P(\mathcal{W}\text{-core})} = e^{\epsilon \cdot y} \tag{A.2}$$

Now, as y tends to ∞, the distribution of Eq. (A.2) is still positive only over max-cores. This means that each max-core will have the same joint probability which is e^{∞}. This in turn implies that the extended factor graph defines a uniform joint distribution over valid max-cores. Hence, running WSP-1's message-passing over the extended factor graph representing Eq. (A.1) estimates marginals over uniformly distributed max-cores. □

Proof of Proposition 1

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 $\mathcal{G}$.

$$\sum_{Y_i} \hat{f}_i(X_1, \ldots, X_n, Y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} = f_i(X_1, \ldots, X_n),\ \forall \hat{f}_i \in \hat{\mathcal{F}} \tag{A.3}$$

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:

$$\begin{aligned} \textstyle\sum_{Y_i} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} &= \textstyle\sum_{Y_i:\mathcal{Z}(y_i=x)} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} \\ &+ \textstyle\sum_{Y_i:\mathcal{Z}(y_i\neq x)} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} \end{aligned} \tag{A.4}$$

The first and the second parts in the right hand side of Eq. (A.4) 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. (4.3) 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:

$$\begin{aligned} \textstyle\sum_{Y_i} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} &= \textstyle\sum_{Y_i:\mathcal{Z}(y_i=x)} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1} \\ &+ \textstyle\sum_{Y_i:\mathcal{Z}(y_i\neq x)} \overbrace{\hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i)\Big|_{\bar{O}_i=1}}^{\text{Zero}} \end{aligned} \tag{A.5}$$

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. (4.2) 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:

$$\sum_{Y_i} \hat{f}_i(x_1, \ldots, x_n, Y_i = y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = f_i(x_1, \ldots, x_n) \tag{A.6}$$

Now since Eq. (A.6) 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:

$$\sum_{Y_i} \hat{f}_i(X_1, \ldots, X_n, Y_i, \bar{O}_i) \Big|_{\bar{O}_i=1} = f_i(X_1, \ldots, X_n),\ \forall \hat{f}_i \in \hat{\mathcal{F}} \tag{A.7}$$

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

Proof of Proposition 2

Proposition 2. Any arbitrary factor graph $\mathcal{G}$ is equivalent, i.e., defining 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:

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

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, $\{X_1, \ldots, X_N\}$.[1] 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:

$$P(X_1, \ldots, X_N) = \prod_{a=1}^{M} f_a(\mathcal{X}_{f_a}) \tag{A.8}$$

Now 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 $\hat{f}_a(\mathcal{X}_{f_a}, O_a, Y_a)$. Now the extended factor graph $\hat{\mathcal{G}}$ includes the original variables $\{X_1, \ldots, X_N\}$, activation variables $\{O_1, \ldots, O_M\}$, and mega-node variables $\{Y_1, Y_2, \ldots, Y_M\}$. It accommodates M extended factors, $\{\hat{f_1}(\mathcal{X}_{f_1}, O_1, Y_1), \ldots, \hat{f_M}(\mathcal{X}_{f_M}, O_M, Y_M)\}$.

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

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

$$P(X_1, \ldots, X_N, Y_1, \ldots, Y_M, O_1, \ldots, O_M) = \prod_{a=1}^{M} \hat{f}_a(\mathcal{X}_{f_a}, O_a, Y_a) \tag{A.9}$$

Now 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 $\bar{O_a} = 1$:

$$P(X_1, .., X_N, Y_1, .., Y_M, \bar{O_1}, .., \bar{O_M}) \Big|_{\bar{O_a}=1, \forall a \in \{1,..,M\}} = \prod_{a=1}^{M} \hat{f}_a(\mathcal{X}_{f_a}, \bar{O_a}, Y_a) \Big|_{\bar{O_a}=1} \tag{A.10}$$

Now, eliminating the auxiliary mega-node variables from Eq. (A.10) by marginalizing each extended factor $\hat{f}_a$ over its mega-node Y_a, we then have:

$$\begin{aligned} P(X_1, \ldots, X_N) &= \textstyle\sum_{Y_1,..,Y_M} P(X_1, .., X_N, Y_1, .., Y_M, \bar{O_1}, .., \bar{O_M}) \Big|_{\bar{O_a}=1, \forall a \in \{1,..,M\}} \\ &= \textstyle\sum_{Y_1,\ldots,Y_M} \prod_{a=1}^{M} \hat{f}_a(\mathcal{X}_{f_a}, \bar{O_a}, Y_a) \Big|_{\bar{O_a}=1} \end{aligned} \tag{A.11}$$

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. (A.11), and we obtain:

$$P(X_1, \ldots, X_N) = \prod_{a=1}^{M} \sum_{Y_a} \hat{f}_a(\mathcal{X}_{f_a}, \bar{O_a}, Y_a) \Big|_{\bar{O_a}=1} \tag{A.12}$$

However, from Proposition 1 we have that:

$$\sum_{Y_a} \hat{f}_i(\mathcal{X}_{f_a}, \bar{O_i}, Y_i) \Big|_{\bar{O_i}=1} = f_i(\mathcal{X}_{f_a}) \tag{A.13}$$

By applying Eq. (A.13) for each reduced extended factor in Eq. (A.12), we obtain the joint probability of the extended factor graph $\hat{\mathcal{G}}$ over the variables $\{X_1, , \ldots, X_N\}$ as follows:

$$P(X_1, , \ldots, X_N) = \prod_{a=1}^{M} f_a(\mathcal{X}_{f_a}) \tag{A.14}$$

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. (A.8), which implies the equivalence between the two factor graphs. □

Proof of Proposition 3

Proposition 3 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. 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:

$$\mathcal{S}_3 = |\mathcal{X}_h| \times T_h + |\mathcal{X}_s| \times T_s \tag{A.15}$$

Now we need to calculate the computational complexity required by both T_h and T_s. For T_h we first calculate $\left|\mathcal{F}^h_{X_j}\right|$ once which requires $\Theta(m_h)$ and then calculate T_l for the set of hard ground clauses that involve X_j as positive and negative respectively, which requires $\Theta(m_h T_l)$. Now to calculate 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:

$$T_h \in \Theta(m_h) + \Theta(m_h T_l) \in O(m_h r) \tag{A.16}$$

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

$$T_s \in \Theta(m_s) + \Theta(m_s T_l) \in O(m_s r) \tag{A.17}$$

Now we take Eqs. (A.16) and (A.17) and substitute for Eq. (A.15) to obtain the computational complexity of $\mathcal{S}_3$:

$$\mathcal{S}_3 \in |\mathcal{X}_h| \times O(m_h r) + |\mathcal{X}_s| \times O(m_s r) \tag{A.18}$$

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

$$\mathcal{S}_3 \in O(nmr) \tag{A.19}$$

Now using Eq. (A.19), the total complexity of the three stages of the algorithm can be bounded as:

$$\mathcal{S}_1 + \mathcal{S}_2 + \mathcal{S}_3 \in O(\max[n, nm, nmr]) = O(nmr) \tag{A.20}$$

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