LA-UR- 09-00306

Approved for public release;
distribution is unlimited.

Title:	Approximate Inference on Planar Graphs Using Loop Calculus and Belief Propagation
Author(s):	Michael Chertkov/146900/T-4/LANL Vicenc Gomez/Radboud University Nijmegen Hilbert J. Kappen/Radboud University Nijmegen
Intended for:	Journal of Machine Learning Research

Los Alamos
NATIONAL LABORATORY
EST. 1943

Form 836 (7/06)

Approximate inference on planar graphs using Loop Calculus and Belief Propagation

Vicenç Gómez VGOMEZ@IUA.UPF.EDU
Hilbert J. Kappen B.KAPPEN@SCIENCE.RU.NL
Department of Biophysics
Radboud University Nijmegen
6525 EZ Nijmegen, The Netherlands

Michael Chertkov CHERTKOV@LANL.GOV
Theoretical Division and Center for Nonlinear Studies
Los Alamos National Laboratory
Los Alamos, NM 87545

Editor:

Abstract

We introduce novel results for approximate inference on planar graphical models using the loop calculus framework. The loop calculus (Chertkov and Chernyak, 2006a) allows to express the exact partition function of a graphical model as a finite sum of terms that can be evaluated once the belief propagation (BP) solution is known. In general, full summation over *all* correction terms is intractable. We develop an algorithm for the approach presented in Chertkov et al. (2008) which represents an efficient truncation scheme and a new representation of the series in terms of pfaffians of a matrix for planar graphs. We analyze the performance of the algorithm for the partition function approximation on models with binary variables and pairwise interactions on grids and on random structure. We study in detail both the loop series and the equivalent pfaffian series and show that the first term of the pfaffian series for the general, intractable planar model, can provide very accurate approximations. The algorithm outperforms previous truncation schemas of the loop series and is competitive with other state-of-the-art methods for approximate inference.

Keywords:

1. Introduction

Graphical models are popular tools widely used in many areas which require modeling of uncertainty. They provide an effective approach through a compact representation of the joint probability distribution. The two most common types of graphical models are bayesian Networks (BN) or markov random fields (MRFs).

The partition function of a graphical model, which plays the role of normalization constant in a MRF or probability of evidence (likelihood) in a BN is a fundamental quantity which arises in many contexts such as hypothesis testing or parameter estimation. Exact computation of this quantity is only feasible when the graph is not too complex, or equivalently, when its tree-width is small. Currently many methods are devoted to approximate this quantity.

The Belief Propagation (BP) algorithm (Pearl, 1988) is at the core of many of these approximate inference methods. Initially thought as an exact algorithm for tree graphs, it is widely used as an approximate method for loopy graphs (Murphy et al., 1999; Frey and MacKay, 1997). The exact partition function is explicitely related with the BP approximation through the loop calculus framework introduced by Chertkov and Chernyak (2006a). loop calculus allows to express the exact partition function as a finite sum of terms (loop series) that can be evaluated once the BP solution is known. Each term maps uniquely to a subgraph, also denoted as a generalized loop, where all nodes have *at least* degree two. Summation of the entire loop series is a hard combinatorial task because the number of generalized loops can be enourmously large. However, different approximations can be obtained by considering different subsets of generalized loops in the graph.

It has been shown empirically (Gómez et al., 2007; Chertkov and Chernyak, 2006b) that truncating this series may provide efficient corrections to the initial BP approximation. More precisely, whenever BP performs satisfactorily which occurs in the case of sufficiently weak interactions between variables or short influcence of loops, only a small quantity of terms is sufficient to recover the exact result (Gómez et al., 2007). On the other hand, for those cases where BP requires many iterations to converge, many terms of the series are required to improve substantially its approximation. A formal characterization of the classes of tractable problems via loop calculus still remains as an open question.

A step toward this goal has been done in Chertkov et al. (2008) where it is shown that for any graphical model, summation of a certain subset of terms can be mapped to a summation of weighted perfect matchings on an extended graph. For planar graphs (graphs that can be embedded in a plane without crossing edges), this problem can be solved in polynomial time evaluating the pfaffian of a skew-symmetric matrix associated to the extended graph. Furthermore, the full loop series can be expessed as a sum over so-called pfaffian terms. Each pfaffian term may account for a large number of loops and is solvable in polynomial time as well.

Their approach builds on work developed in the 1960s by Kasteleyn (1963); Fisher (1966) and other physicists who addressed the question of how many perfect matchings exist in a graph, also known as the dimer problem in the statistical physics literature (a dimer represents an edge between two nearest neighbors in a graph). Their key result can be summarized as follows: the partition function of a *planar graphical model defined on binary variables* can be mapped to a weighted perfect matching problem and calculated in polynomial time under the following two restrictions:

1. Symmetric pairwise interactions: Interactions exist only between two variables and only depend on disagreement between their values.

2. Single variable potentials (local fields) are zero for all variables.

Such a model is known in statistical physics as the binary Ising model *without external field*. Notice that exact inference on a general binary planar is intractable (Barahona, 1982).

Recently, other methods for inference on graphical models which are based in the work of Kasteleyn and Fisher have been introduced. Globerson and Jaakkola (2007) obtained upper bounds on the partition function for non-planar graphs with binary variables by decomposition of the partition function into a weighted sum over partition functions of

spanning planar models which fulfill the two previous conditions. The resulting problem is a convex optimization problem and, since exact inference can be done in each planar *sub*-model, the bound can be calculated in polynomial time.

Another example is the work of Schraudolph and Kamenetsky (2008) which provides a framework for exact inference on a restricted class of planar graphs using the approach of Kasteleyn and Fisher. More precisely, they showed that any joint probability function defined on binary variables can be expressed in a functional form which fulfills the two previous conditions after addition of a new auxiliary node linked to all the existing nodes. Under this transformation, the two conditions can be relaxed at the cost of restricting the graphical model to be $\mathcal{B}-$outerplanar, which means that there must exist a planar embedding in which a subset $\mathcal{B}$ of the nodes lie on the same face. In this way, they can allow for instance single-variable potentials for the subset $\mathcal{B}$ of variables.

Contrary to the two aforementioned approaches which rely on exact inference on a tractable planar model, the loop calculus directly leads to a framework for approximate inference on general planar graphs. Truncating the loop series according to Chertkov et al. (2008) already gives the exact result if the planar graph fulfills the two previous constraints. In the general planar case, however, this truncation may result into an accurate approximation that can be incrementally corrected by considering subsequent terms in the series.

In the next Section we review the main theoretical results on loop calculus and planar graphs and introduce the algorithm used in this work. In Section 3 we provide experimental results on approximation of the partition function for grids and randomly structured graphs. We focus on a planar-intractable binary model with symmetric pairwise interactions but nonzero single variable potentials. The source code used for this results is freely available at `http://www.mbfys.ru.nl/staff/v.gomez/`. We end this manuscript with conclusions and future work in Section 4.

2. Belief Propagation and loop Series for Planar Graphs

We consider the Forney graph representation, also called general vertex model (Forney, 2001; Loeliger, 2004), of a probability distribution $p(\boldsymbol{\sigma})$ defined over a vector $\boldsymbol{\sigma}$ of binary variables (vectors are denoted using bold symbols). Forney graphs are general graphical models which subsume factor graphs and therefore both bayesian networks and markov random fields. In appendix A we show how to convert a factor model to its equivalent Forney graph representation.

A binary Forney graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ consists of a set of nodes $\mathcal{V}$ where each node $a \in \mathcal{V}$ represents an interaction and each edge $(a, b) \in \mathcal{E}$ represents a binary variable ab which take values $\sigma_{ab} := \{\pm 1\}$. We denote $\bar{a}$ the set of neighbors of node a. Interactions $f_a(\boldsymbol{\sigma}_a)$ are arbitrary functions defined over typically small subsets of variables where $\boldsymbol{\sigma}_a$ is the vector of variables associated with node a, i.e. $\boldsymbol{\sigma}_a := (\sigma_{ab_1}, \sigma_{ab_2}, \dots)$ where $b_i \in \bar{a}$.

The joint probability distribution of such a model factorizes as:

$$p(\boldsymbol{\sigma}) = Z^{-1} \prod_{a \in \mathcal{V}} f_a(\boldsymbol{\sigma}_a), \qquad Z = \sum_{\boldsymbol{\sigma}} \prod_{a \in \mathcal{V}} f_a(\boldsymbol{\sigma}_a), \tag{1}$$

where Z is the normalization factor or partition function.

From a variational perspective, a fixed point of the BP algorithm represents a stationary point of the Bethe "free energy" approximation under proper contraints (Yedidia et al., 2000). In Forney style notation:

$$Z^{BP} = \exp\left(-F^{BP}\right), \quad F^{BP} = \sum_{a}\sum_{\boldsymbol{\sigma}_a} b_a\left(\boldsymbol{\sigma}_a\right)\ln\left(\frac{b_a(\boldsymbol{\sigma}_a)}{f_a(\boldsymbol{\sigma}_a)}\right) - \sum_{b\in\bar{a}}\sum_{\sigma_{ab}} b_{ab}\left(\sigma_{ab}\right)\ln b_{ab}\left(\sigma_{ab}\right), \tag{2}$$

where $b_a(\boldsymbol{\sigma}_a)$ and $b_{ab}(\sigma_{ab})$ are the beliefs (pseudo-marginals) associated to each node $a \in \mathcal{V}$ and variable ab. For graphs without loops, Equation (2) coincides with the Gibbs "free energy" and therefore Z^{BP} coincides with the exact partition function Z. If the graph contains loops, Z^{BP} is just an approximation more or less accurate depending on how strong the influence of the loops is.

We introduce now some convenient definitions related to the loop calculus framework.

Definition 1 *A **generalized loop** in a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}\rangle$ is any subgraph $C = \langle V', E'\rangle$, $V' \subseteq \mathcal{V}, E' \subseteq (V' \times V') \cap \mathcal{E}$ such that each node in V' has degree two or larger.*

For simplicity, we will use the term loop in rest of this manuscript. loop calculus allows to explicitely represent Z in terms of the BP approximation via the loop series expansion:

$$Z = Z^{BP}\cdot z, \qquad z = \left(1 + \sum_{C\in\mathcal{C}} r_C\right), \qquad r_C = \prod_{a\in C}\mu_{a;\bar{a}_C}, \tag{3}$$

where $\mathcal{C}$ is the set of all loops in the graph. Each loop term r_C is a product of terms $\mu_{a,\bar{a}_C}$, each of them associated to every node a of the loop. $\bar{a}_C$ denotes the set of neighbours of a which appear the loop C:

$$\mu_{a;\bar{a}_C} = \frac{\sum_{\boldsymbol{\sigma}_a} b_a\left(\boldsymbol{\sigma}_a\right)\prod_{b\in\bar{a}_C}\left(\sigma_{ab} - m_{ab}\right)}{\prod_{b\in\bar{a}_C}\sqrt{1-m_{ab}^2}}, \qquad m_{ab} = \sum_{\sigma_{ab}}\sigma_{ab} b_{ab}\left(\sigma_{ab}\right). \tag{4}$$

In this work we consider planar graphs where all nodes have degree no larger than 3. We denote by *triplet* a node with degree three in the graph. In appendix A we show that a graphical model can be converted to this representation at the cost of introducing auxiliary nodes.

Definition 2 *A **2-regular loop** is a loop in which all nodes have degree* exactly *two.*

Definition 3 *The **2-regular partition function** $Z_\emptyset$ is the truncated form of* (3) *which sums all* 2*-regular loops only:* [1]

$$Z_\emptyset = Z^{BP}\cdot z_\emptyset, \qquad z_\emptyset = 1 + \sum_{C\in\mathcal{C}|d(a)=2,\forall a\in C} r_C. \tag{5}$$

As an example, Figure 1a shows a small Forney graph and Figure 1b the seven loops included in its corresponding 2-regular partition function.

1. Notice that this part of the series is denoted as *single-connected partition function* in Chertkov et al. (2008). We prefer to use 2-regular partition function because loops with more than one connected component are also included in this part of the series.

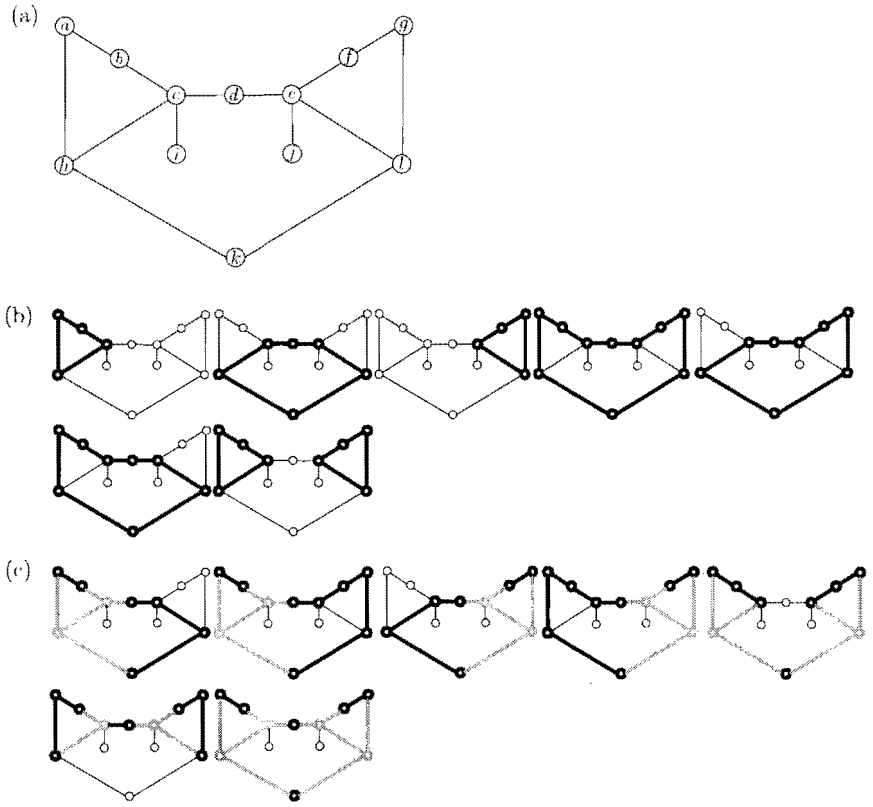

Figure 1: Example. **(a)** A Forney graph. **(b)** Loops (in bold) included in the 2-regular partition function. **(c)** Loops (in bold and red) not included in the 2-regular partition function. Marked in red, the triplets associated with each loop.

2.1 Computing the 2-regular Partition Function Using Perfect Matching

In Chertkov et al. (2008) it has been shown that computation of $z_\emptyset$ can be mapped to a dimer matching problem, more precisely, to the computation of the sum of all weighted perfect matchings in a graph. A perfect matching is a subset of edges such that each node is met by exactly one edge in the subset: The weight of a matching is the product of the weights of the edges in the matching. The key idea of this mapping is to extend the original Forney graph $\mathcal{G}$ into an new graph $\mathcal{G}_{ext}$ in such a way that each perfect matching in $\mathcal{G}_{ext}$ corresponds to a 2-regular loop in $\mathcal{G}$. Under the planarity condition, the sum of all weighted perfect matchings can be done in polynomial time following Kasteleyn's argument. Here we reproduce these results with little variations and more emphasys in algorithmic aspects.

Given a Forney graph $\mathcal{G}$ and the BP approximation, we simplify $\mathcal{G}$ and obtain the 2-core by removing nodes of degree 1 recursively. After this step, either $\mathcal{G}$ is the null graph (and then BP is exact) or $\mathcal{G}$ is composed of vertices of degree 2 or 3 only. At this point, all terms

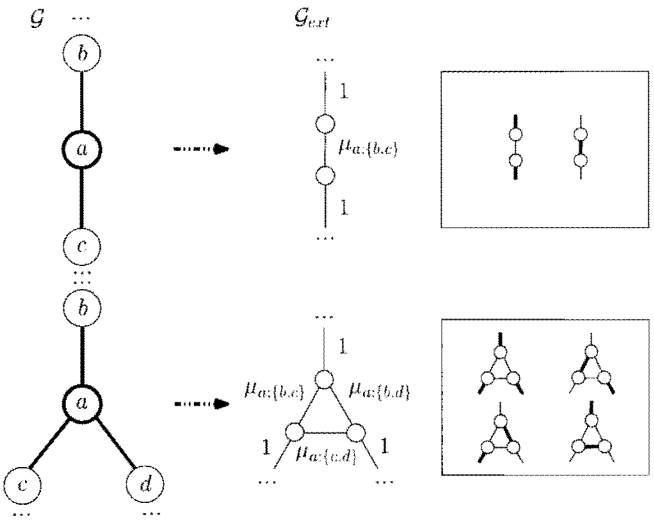

Figure 2: Fisher's rules. The original Forney graph $\mathcal{G}$ is converted to an extended graph $\mathcal{G}_{ext}$ where each node is split in two or three nodes according to its degree. Note that rules preserve planarity.

appearing in equation (4) can be computed. Also, it is convenient for subsequent steps to have a biconnected graph (a graph that remains connected after removal of any vertex). If necessary, we add dummy edges with zero weight which do not alter the partition function.

To construct the extended graph $\mathcal{G}_{ext}$ we split each node with degree 2 or 3 in $\mathcal{G}$ according to the rules introduced by Fisher (1966), see Figure 2. Edge weights associated to external edges are set to unity and new internal edges are weighted according to the corresponding terms $\mu_{a;\bar{a}_C}$ of Equation (4). It is easy to see that each 2-regular loop in $\mathcal{G}$ is associated to a perfect matching in $\mathcal{G}_{ext}$ and, furthermore, this correspondence is *unique*. Consider, for instance, the vertex of degree three in the bottom of Figure 2. Given a 2-regular loop C, vertex a can appear in four different configurations: either node a does not appear in C, or C contains one of the following three *sub*-paths: -*b*-*a*-*c*-, -*b*-*a*-*d*- or -*c*-*a*-*d*-. These four cases correpond to *sub*-terms in a loop with values 1, $\mu_{a;\{b,c\}}$, $\mu_{a;\{b,d\}}$ and $\mu_{a;\{c,d\}}$ respectively and coincide with the matchings shown within the box on the left. A simpler reasoning holds for the vertex of degree two of the top of Figure 2.

This correspondence allows to express the 2-regular partition function $Z_\emptyset$ as the sum over all weighted perfect matchings defined on $\mathcal{G}_{ext}$. Kasteleyn (1963) provided a method to compute this sum in polynomial time for planar graphs. We follow his approach: first, we orient edges of a planar embedding of $\mathcal{G}_{ext}$ in such a way that for every face (except possibly the external face) the number of clockwise oriented edges is odd. For a planar graph this orientation can be easily calculated. Algorithm 1 produces such an orientation (Karpinski and Rytter, 1998).

Algorithm 1 Pfaffian orientation

Arguments: undirected biconnected graph $\mathcal{G}_{ext}$.

Construct a planar embedding $\bar{\mathcal{G}}_{ext}$ of $\mathcal{G}_{ext}$.
Construct a spanning tree T of $\bar{\mathcal{G}}_{ext}$.
Construct a faces tree H having vertices corresponding to the faces of $\bar{\mathcal{G}}_{ext}$:
connect two vertices in H if the respective face boundaries share an edge not in T.
for all face (vertex in H) **do**
    Orient all its edges in T arbitrarily (but keep the number of clock-wise oriented edges).
end for
for all face (vertex in H) traversed in postorder **do**
    Add the unique edge not in T.
    Orient it such that the number of clock-wise oriented edges is odd.
end for
RETURN directed extended graph T

Next, denote μ_{ij} the weight of the edge between nodes i and j in $\mathcal{G}_{ext}$. After orienting the graph we create the following skew-symmetric matrix $\hat{A} = -\hat{A}^t$:

$$\hat{A}_{ij} = \begin{cases} +\mu_{ij} & \text{if } (i,j) \in \mathcal{E}_{\mathcal{G}_{ext}} \\ -\mu_{ij} & \text{if } (j,i) \in \mathcal{E}_{\mathcal{G}_{ext}} \\ 0 & \text{otherwise} \end{cases}.$$

This matrix is known as the Tutte matrix of $\mathcal{G}_{ext}$ and the pfaffian of $\hat{A}$ gives the desired sum. The pfaffian is an algebraic object very related with the determinant. More precisely $\mathrm{Pf}(\hat{A})^2 = \mathrm{Det}(\hat{A})$. Calculation of $z_\emptyset$ can therefore be performed in time $\mathcal{O}(N^3)$ where N is the number of nodes of $\mathcal{G}_{ext}$ (Galbiati and Maffioli, 1994).

In our case, however, $z_\emptyset$ can be either positive or negative, and computing the value of the pfaffian up to the sign is not enough for our purposes. Furthermore, since each element $\hat{A}_{ij}$ can be negative not only because of the pfaffian orientation but also because of Equation (4), the sign of the pfaffian needs to be *corrected.* Our solution considers the original Kasteleyn's binary matrix:

$$\hat{B}_{ij} = \begin{cases} +1 & \text{if } (i,j) \in \mathcal{E}_{\mathcal{G}_{ext}} \\ -1 & \text{if } (j,i) \in \mathcal{E}_{\mathcal{G}_{ext}} \\ 0 & \text{otherwise} \end{cases}.$$

If the sign of $\mathrm{Pf}(\hat{B})$ is negative then the sign of $\mathrm{Pf}(\hat{A})$ is changed. The absolute value $\mathrm{Pf}(\hat{B})$ coincides with the number of perfect matchings or the number of loops included in the sum. The sign represents the correction. The corrected value of $z_\emptyset$ is:

$$z_\emptyset = \mathrm{sign}\left(\mathrm{Pf}\left(\hat{B}\right)\right) \cdot \mathrm{Pf}\left(\hat{A}\right).$$

For the special case of binary planar graphs with pure interaction potentials (symmetric and zero local fields) the 2-regular partition function coincides with the exact partition function $Z = Z_\emptyset = Z^{BP} \cdot z_\emptyset$ since the other terms in the series vanish.

2.2 Computing the Full Loop Series Using Perfect Matching

Chertkov et al. (2008) established that $z_\emptyset$ is just the first term of a finite sum involving pfaffians. We briefly reproduce their results here and provide an algorithm to compute the full loop series as a pfaffian series.

Consider $\mathcal{T}$ the set of all possible triplets (vertices with degree 3 in the original graph $\mathcal{G}$). For each possible subset $\Psi \in \mathcal{T}$ including an *even* number of triplets there exists a unique correspondence between loops in $\mathcal{G}$ including the triplets in Ψ and perfect matchings in another extended graph $\mathcal{G}_{ext_\Psi}$ constructed after removal of the triplets Ψ in $\mathcal{G}$. Using this representation the full loop series can be represented as a pfaffian series, where each term Z_Ψ is tractable and is a product of the respective pfaffian with the $\mu_{a;\bar{a}}$ terms associated with each triplet of Ψ:

$$z = \sum_\Psi Z_\Psi \qquad Z_\Psi = z_\Psi \prod_{a \in \Psi} \mu_{a;\bar{a}} \qquad (6)$$

$$z_\Psi = \text{sign}\left(\text{Pf}\left(\hat{B}_\Psi\right)\right) \cdot \text{Pf}\left(\hat{A}_\Psi\right).$$

The 2-regular partition function thus corresponds to $\Psi = \emptyset$. We refer to the remaining terms of the series as higher-order pfaffian terms. Notice that matrices $\hat{A}_\Psi$ and $\hat{B}_\Psi$ depend on the removed triplets and therefore each z_Ψ requires different matrices and different edge orientations. In addition, after removal of vertices in $\mathcal{G}$ the resulting graph may be disconnected. As before, in these cases we add dummy edges with zero weight to make the graph biconnected again. Notice then that $|\text{Pf}(\hat{B}_\Psi)|$ is no longer the number of loops included in the pfaffian but an upper bound.

Figure 1c shows the loops corresponding to the higher-order pfaffian terms of the example. The first and second subsets of triplets (grouped in grey squares) include summation over two loops and the remaining pfaffian terms include uniquely one loop.

Exhaustive enumeration of all subsets of triplets leads to a $2^{|\mathcal{T}|}$ time algorithm, which is prohibitive. However, many triplet combinations may lead to forbidden configurations. Experimentally, we found that a principled way to look for higher order pfaffian terms with large contribution is to search first for pairs of triplets, then groups of four, and so on. For large graphs, this also becomes intractable. Actually, the problem is very similar to the problem of selecting loop terms r_C with largest contribution. The advantage of the pfaffian representation, however, is that $Z_\emptyset$ is always the pfaffian term that accounts for the largest number of loop terms and is the most contributing term in the series. In this work we do not derive any heuristic to search pfaffians terms with larger contributions. Instead, we study the full pfaffian series and in particular we focus on the accuracy of $Z_\emptyset$.

Algorithm 2 describes the full procedure to compute all terms using the representation of expression (6). The main loop can be stopped at to obtain an anytime algorithm which produces corrections incrementally.

3. Experiments

In this section we study numerically the proposed algorithm. To facilitate the evaluation and the comparison with other algorithms we focus on the the binary Ising model, a particular case of the model (1) where factors only depend on the disagreement between two variables

Algorithm 2 Pfaffian series

Arguments: Forney graph $\mathcal{G}$

1: $z := 0$.
2: **for all** $(\Psi \in \mathcal{T})$ **do**
3: Build extended graph $\mathcal{G}_{ext_\Psi}$ applying rules of Figure 2.
4: Set pfaffian orientation in $\mathcal{G}_{ext_\Psi}$ according to Algorithm 1
5: Build matrices $\hat{A}$ and $\hat{B}$.
6: Compute pfaffian with sign correction z_Ψ according to Equation (3).
7: $z := z + z_\Psi \prod_{a \in \Psi} \mu_{a;\bar{a}}$.
8: **end for**
9: **RETURN** $Z^{BP} \cdot z$

and take the form $f_a(\sigma_{ab}, \sigma_{ac}) = \exp\left(J_{a;\{ab,ac\}}\sigma_{ab}\sigma_{ac}\right)$. We consider also nonzero local potentials parametererized by $f_a(\sigma_{ab}) = \exp\left(J_{a;\{ab\}}\sigma_{ab}\right)$ in all variables so that the model becomes planar-intractable.

We create different inference problems by choosing different interactions $\{J_{a;\{ab,ac\}}\}$ and local field parameters $\{J_{a;\{ab\}}\}$. To generate them we draw independent samples from a Normal distribution $\{J_{a;\{ab,ac\}}\} \sim \mathcal{N}(0, \beta/2)$ and $\{J_{a;\{ab\}}\} \sim \mathcal{N}(0, \beta\Theta)$, where Θ and β determine how difficult the inference problem is. Generally, for $\Theta = 0$ the planar problem is tractable. For $\Theta > 0$, small values of β result in weakly coupled variables (easy problems) and large values of β in strongly coupled variables (hard problems). Larger Θ result in easier inference problems.

In the next subsection we analyze the full pfaffian series using a small example and compare it with the original representation based on the loop series. Next, we compare the following algorithms for approximation of the partition function: [2]

Truncated Loop-Series for BP (TLSBP) (Gómez et al., 2007), which accounts for a certain number of loops by perfoming depth-first-search on the factor graph and then merging the found loops iteratively. We adapted TSLBP as an anytime algorithm (**anyTLSBP**) such that only the length of the loop is used as argument instead of the two parameters S and M, see Gómez et al. (2007) for details. This approach does not compute all possible loops (in particular, complex loops are not included), but is more efficient than TLSBP since merging two loops only involve bitwise operations.

Cluster Variation Method (**CVM-Loopk**) A double-loop implementation of CVM (Heskes et al., 2003). This algorithm is a special case of generalized belief propagation (Yedidia et al., 2005) with convergence guarantees. We use as outer clusters all (maximal) factors together with loops in the factor graph that consist of up to **k** different variables. We experimented with different values of $\mathbf{k} \in \{4, 6, 8\}$.

Tree-Structured Expectation Propagation (**TreeEP**) (Minka and Qi, 2004). This method performes exact inference on a base tree of the graphical model and approximates the other interactions. This method yields good results if the graphical model is very sparse, which is the case of planar models. We generalized the method of

2. We use the libDAI library (Mooij, 2008) for algorithms **CVM-Loopk**, **TreeEP** and **TRW**.

choosing the base tree described in Minka and Qi (2004) to multiple variable factors as follows: when estimating the mutual information between x_i and x_j , we take the product of the marginals on $\{i, j\}$ of all the factors that involve x_i and/or x_j. Other generalizations of TreeEP to higher- order factors are possible (e.g., by clustering variables), but it is not clear how to do this in general in an optimal way.

When possible, we also compare with the following two methods which provide upper bounds of the partition sum:

Planar graph decomposition (PDC) (Globerson and Jaakkola, 2007) which decomposes the vector of factors into a mixture over planar graphs.

Tree Reweighting (TRW) (Wainwright et al., 2005) which decomposes the vector of factors into a mixture over spanning trees.

To evaluate the accuracy of the approximations we consider errors in Z and, when possible, computational cost as well. As shown in Gómez et al. (2007), errors in Z obtained from a truncated form of the loop series are very similar to errors in single variable marginal probabilities, which can be obtained by conditioning over the variables under interest. We only consider tractable instances for which Z can be computed via the junction tree algorithm (Lauritzen and Spiegelhalter, 1988). Given an approximation Z' of Z, the error measure used in this manuscript is:

$$\text{error} Z' = \frac{|\log Z - \log Z'|}{\log Z}.$$

As in Gómez et al. (2007), we use four different schedulings of the messages for BP: fixed and random sequential updates, parallel (or synchronous) updates, and residual belief propagation (RBP), a method proposed by Elidan et al. (2006) which selects the next message to be updated which has maximum *residual*, a quantity defined as an upper bound on the distance of the current messages from the fixed point. We report non-convergence when none of the previous methods converged. We report convergence at iteration t when the maximum absolute value of the updates of all the messages from iteration $t-1$ to t is smaller than a threshold $\vartheta = 10^{-14}$.

3.1 Full Pfaffian Series

In the previous section we have described two equivalent representations of the exact partition function in terms of loop series and pfaffian series. Here we analyze numerically how these two representations differ using an example, shown in Figure 3 as a factor graph, for which all terms of both series can be computed. We analyze a single instance parameterized using $\Theta = 0.1$ and different pairwise interactions $\beta \in \{0.1, 0.5, 1.5\}$. We expect similar results for larger instances.

We use TLSBP to retrieve all loops, 8123 for this example, and Algorithm 2 to compute all pfaffian terms. To compare the two approximations we sort all contributions, either loops or pfaffians, by their absolute value in decreasing order, and then analyze how the errors are corrected as more terms are included in the approximations. We define partition

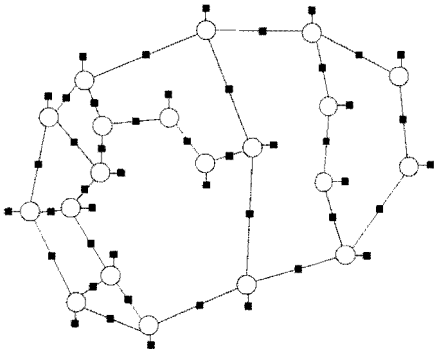

Figure 3: Planar graph used for the example in factor graph representation. Circles and black squares denote variables and factors respectively.

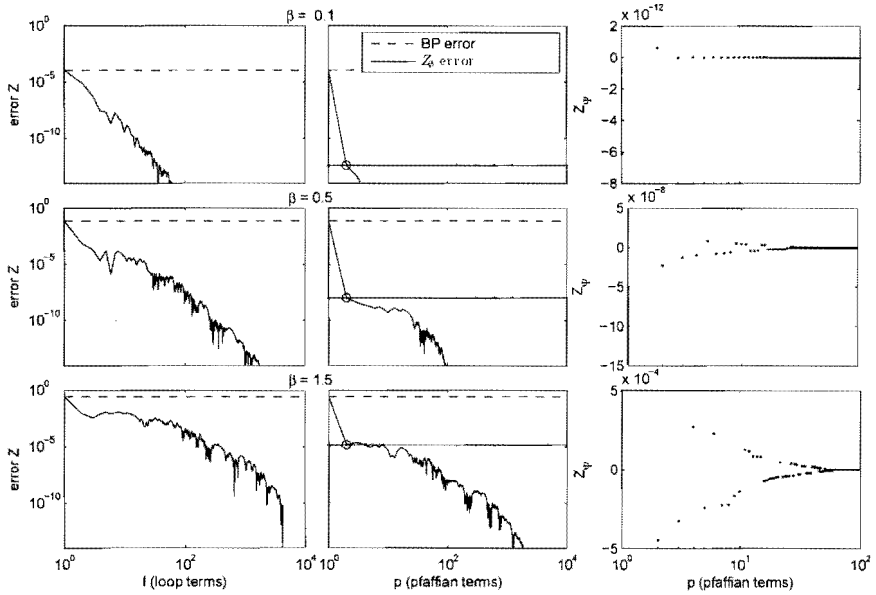

Figure 4: Comparison between full loop series and full pfaffian series. Each row corresponds to a different value of the interaction strength β. **Left column** shows the error considering loop terms $Z^{TLSBP}(l)$ in log-log scale. Shaded regions include all loop terms (not necesarily 2-regular loops) to reach the same (or better) accuracy than the accuracy of the 2-regular partition function $Z_\emptyset$. **Middle column** shows the error considering pfaffian terms $Z^{Pf}(p)$ also in log-log scale. The first pfaffian term corresponds to $Z_\emptyset$ and is marked with a circle. **Right column** shows the values of the first 100 pfaffians terms sorted by $|Z_\Psi|$ and excluding $z_\emptyset$.

functions for the truncated series in the following way:

$$Z^{TLSBP}(l) = Z^{BP}\left(1 + \sum_{i=1\ldots l} r_{C_i}\right), \qquad Z^{Pf}(p) = Z^{BP}\left(\sum_{i=1\ldots p} Z_{\Psi_i}\right).$$

In this way $Z^{TLSBP}(l)$ accounts from the l most contributing loops and $Z^{Pf}(p)$ accounts for the p most contributing pfaffian terms. In all cases, the pfaffian term with largest absolute value Z_{Ψ_1} corresponds to $z_\emptyset$.

Figure 4 shows the error Z^{TLSBP} (first column) and Z^{Pf} (second column) of both representations. For weak interactions ($\beta = 0.1$) BP converges fast and provides an accurate approximation with an error of order 10^{-4}. Summation of less than 50 loop terms (top-left panel) is enough to obtain machine precision accuracy. Notice that the error is almost reduced totally with the $z_\emptyset$ correction (top-middle panel). In this scenario, higher order terms of the pfaffian series are negligible (top-right panel).

As we increase the complexity of the problem, the quality of the BP approximation decreases. The number of loop corrections required to correct the BP error then increases. In this example, for intermediate interactions ($\beta = 0.5$) the first pfaffian term $z_\emptyset$ improves considerably, more than 5 orders of magnitude, on the BP error. Note that approximately 10^2 loop terms are required to achieve the same correction as the one obtained by $z_\emptyset$ (grey region of middle-left panel).

For strong interactions ($\beta = 1.5$) BP converges after many iterations and gives a poor approximation. In this scenario also a larger proportion of loop terms (bottom-left panel) is necessary to correct the BP result up to machine precision. For the particular example we see (bottom-left panel) that approximately 200 loop terms are required to achieve the same correction as the one obtained by $z_\emptyset$. The $z_\emptyset$ is quite accurate (bottom-middle panel).

As the right panels show, higher order pfaffian contributions tend to change progressively from a flat sequence with a few relevant terms to a alternating sequence of positive and negative terms which grow in absolute value as the difficulty of the problem increases. These oscillations are present in both representations and can be explained by the mixed type of interactions.

In general, we can conclude that the $z_\emptyset$ correction to the BP approximation can be significant even in hard problems for which BP converges after many iterations, and does not require and explicit search for loops.

3.2 Grids

After analyzing the full pfaffian series on a small random example we now restrict to the $Z_\emptyset$ approximation using Ising grids (nearest neighbor connectivity). First, we compare that approximation with other inference methods for different types of interactions (attractive or mixed) and then we study the scalability of the method in function of the size of the graphs.

3.2.1 ATTRACTIVE INTERACTIONS

We first focus on binary models with interactions that tend to align the coupled variables to the same value, $J_{a;\{ab,ac\}} > 0$. If local fields are also positive $J_{a;\{ab\}} > 0, \forall a \in \mathcal{V}$, Sudderth

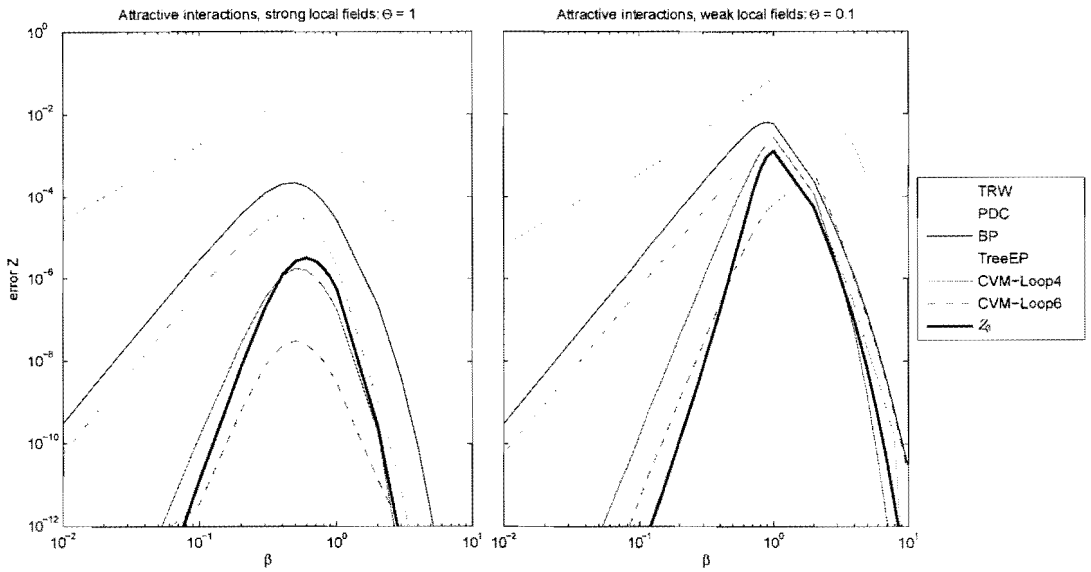

Figure 5: 7x7 grid attractive interactions and positive local fields. Curves show error averages over 50 random instances in function of the difficulty of the problem.

et al. (2008) showed that the BP approximation is a *lower-bound* of the exact partition function and all loops (and therefore pfaffian terms too) have the same sign[3]. Although this is not formally proven for general models with attractive interactions regardless the sign of the local fields, numerical results suggest that this propery holds as well for that type of models.

We generate grids with positive interactions and local fields, i.e. $|\{J_{a;bc}\}| \sim \mathcal{N}(0, \beta/2)$ and $|\{J_{a;b}\}| \sim \mathcal{N}(0, \beta\Theta)$ and study the performance for various values of β and for strong $\Theta = 1$ and weak $\Theta = 0.1$ local fields.

Figure 5 shows the average error over 50 instances reported by different methods. Using this setup, BP converged in all instances using sequential updates of the messages. The error curves of all methods show an initial growth and a subsequent decrease, a fact explained by the phase transition occurring in this model for $\Theta = 0$ and $\beta \approx 1$ (Mooij and Kappen, 2005). As the difference between the two plots suggest, errors are larger as Θ approaches zero. Notice, however, that $Z_\emptyset = Z$ for the limit case of $\Theta = 0$.

We observe that in all instances $Z_\emptyset$ *always improves* on the BP approximation. Corrections are more significant for weak interactions $\beta < 1$ and strong local fields. For strong interactions $\beta > 1$ and weak local fields the improvement is less significant, although this occurs for all methods under consideration.

The $Z_\emptyset$ approximation turns to perform better than TreeEP in all cases except for very strong couplings, where they show very similar results. Interestingly, $Z_\emptyset$ performs very

3. The condition is that all single variable beliefs at the BP fixed point must satisfy $m_{ab} = b_{ab}(+1) - b_{ab}(-1) > 0, \forall (a,b) \in \mathcal{E}$

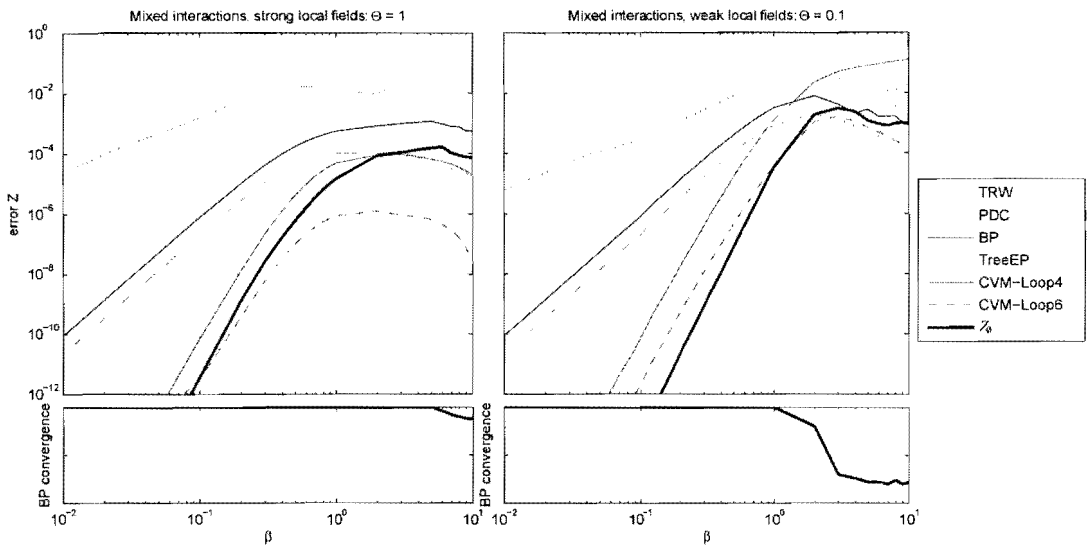

Figure 6: 7x7 grid mixed interactions. Curves show error averages over 50 random instances in function of the difficulty of the problem. Bottom panels show percentage of cases in which BP converged.

similar to CVM-Loop4 which is known to be a very accurate approximation for this type of model, see Yedidia et al. (2000) for instance. We observe that in order to obtain better average results than $Z_\emptyset$ using CVM, we need to select larger outer-clusters such as loops of length 6, which increases dramatically the computational cost.

The methods which provide upper bounds of Z (PDC and TRW) report the largest average error. PDC performs slightly better than TRW, as has been shown in Globerson and Jaakkola (2007) for the case of mixed interactions. We remark that the worse performance of PDC for stronger couplings and weak local fields might be attributed to implementation artifacts, since for $\beta > 4$ we have numerical precision errors. In general, both upper bounds are significantly less tight than the lower bounds provided by BP and $Z_\emptyset$.

3.2.2 Mixed Interactions

We now analyze a more general Ising grid model where interactions and local fields can have mixed signs. In that case, Z^{BP} and $Z_\emptyset$ are no longer lower bounds of Z and loop terms can be positive or negative. Figure 6 show results using this setup. Top panels show average errors and bottom panels show percent of instances in which BP converged using at least one of the methods described in section.

For strong local fields (left panels), as β grows we see that the improvements of $Z_\emptyset$ on BP are on average less significant. It is important to note that $Z_\emptyset$ corrects always the BP result, even for very strong couplings ($\beta = 10$) where a small percentage of the graphs failed to converge. $Z_\emptyset$ performs slightly better than CVM-Loop4 and substantially better than

TreeEP for small β . All three methods show similar results for strong couplings $\beta > 2$. As in the case of attractive interactions, the best results are attained using CVM-loop6.

For the case of weak local fields (right panels), BP fails to converge near the transition to the spin-glass phase. For $\beta = 10$, BP converges only in less than 25% of the instances. In the difficult configurations, $\beta > 22$, all methods under consideration give similar results (all comparable to BP). Moreover, it may happen that $Z_\emptyset$ degrades the Z^{BP} approximation because loops of alternating signs have major influence in the series. This result was also reported in Gómez et al. (2007) when loop terms instead of pfaffian terms where considered.

3.2.3 SCALING WITH GRAPH SIZE

We now study how the $Z_\emptyset$ approximation accuracy varies as we increase the size of the grid. We generate random grids with mixed couplings for $\sqrt{N} = \{4, ..., 19\}$ and focus on a regime of intermediate local fields $\Theta = 0.5$ and strong couplings $\beta = 1$. We compare also with anyTLSBP, a variant of our previous algorithm for truncating the loop series. We run anyTLSBP by selecting all loops shorter than a given length, which we increase progresively. To provide a fair comparison between both methods, we run anyTLSBP for the same amount of cpu time as the one required to obtain $Z_\emptyset$ and take the last valid result.

Figure 7 (left) shows the errors of different methods. Since variability in the errors was larger than before, we took the median for comparison. We can see that results are similar to previous experiments. In order of increasing accuracy we have BP, TreeEP, CVM-Loop4 and $Z_\emptyset$. We note that larger clusters in CVM would have given better results.

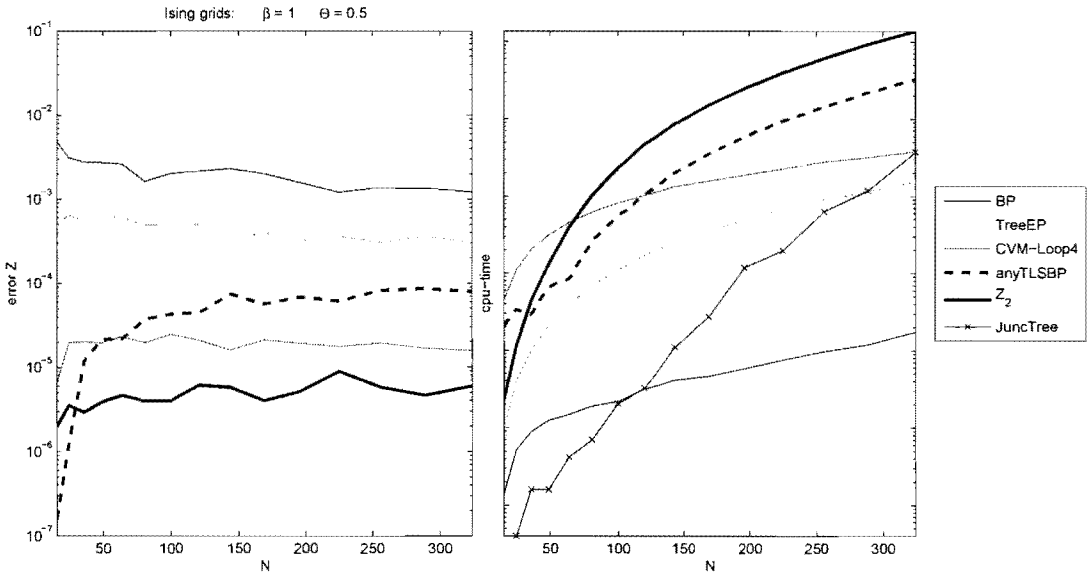

Figure 7: Scaling with grid size for $\Theta = 0.5$ and $\beta = 0.1$. Results are error medians over 50 instances. BP converged in all cases.

Overall, we can see that the error stays approximately constant as we increase N in almost all methods that we compare. The error of anyTLSBP starts being the smallest but soon increases because the proportion of loops captured decreases very fast. For $N > 100$ anyTLSBP performs worse than CVM-Loop4. The $Z_\emptyset$ correction, on the other hand, stays more constant and we can conclude that for this configuration the $Z_\emptyset$ correction scales reasonably well. Note that, although anyTLSBP is systematically worse than $Z_\emptyset$, they show similar scaling behaviour for large graphs, $N > 150$, which is worth to mention because both truncation methods are based on different principles: whereas $Z_\emptyset$ truncates all non 2-regular loops, anyTLSBP truncates loops of size larger than a certain length.

Figure 7 (right) shows the cpu time of all tested approaches. The cpu time required to compute $Z_\emptyset$ scales with $O(N^3_{G_{ext}})$ but turns out to be the least efficient approach. We in section 4. The cpu time of the Junction-tree method quickly increases with the tree-width of the graphs. For large enough N exact solution via the Junction-tree method is no longer feasible and $Z_\emptyset$ represents an improvement over BP.

3.3 Random Graphs

In the previous subsection we have studied the $Z_\emptyset$ correction on graphs with grid structure. It is interesting to analyze the $Z_\emptyset$ correction in graphs with no nearest neighbor connectivity. For that we consider random planar graphs which we generate using the method described in Schaeffer (1999). This method allows to generate efficiently random planar graph with a given number of edges. We focus on 3-connected planar graphs. The resulting graphs are

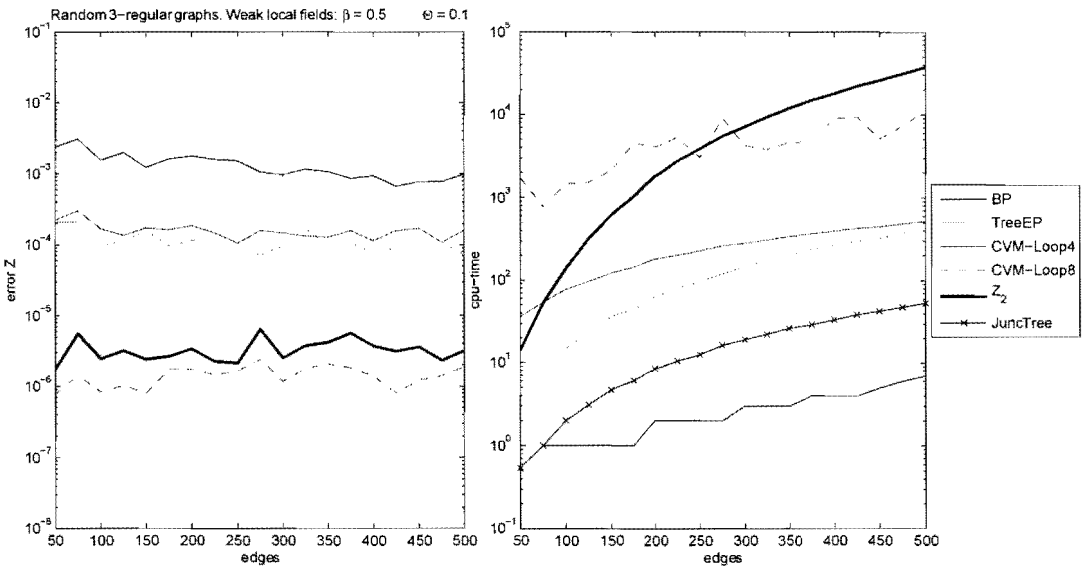

Figure 8: Scaling with number of edges for moderate interactions and weak local fields. Results are error medians over 50 instances. BP converged in all cases.

interpreted as factor graphs with pairwise interactions which are converted to an equivalent Forney graph representation (see appendix A for details).

Inference problems are now parameterized by the number of edges of the graphical model and the interaction and local field parameters. We choose weak local field potentials $\Theta = 0.1$ and intermediate couplings $\beta = 0.5$ and analyze the error varying the number of edges. This setup generates extended graphs $\mathcal{G}_{ext}$ (after application of Fisher rules) of sizes between $2 \cdot 10^2$ and $2.3 \cdot 10^3$ nodes. We note that this is a particular challenging problem for approximate inference methods but not for exact inference, because although the number of nodes grows fast, the graphs are still sparse and their tree-widths do not grow as dramatically as in the grids case.

Figure 8 (left) shows errors of different methods. First, we remark that the improvement of CVM-Loop4 over BP is less significant than in the grids scenario. In this case, CVM-Loop4 performs very similarly to TreeEP. This is probably explained because CVM-Loop4 exploits symmetries in the structure of the networks which do not occur in the graphs considered here. Second, the $Z_\emptyset$ approximation performs better than these two approaches. However, selecting larger clusters such as in CVM-Loop8 the $Z_\emptyset$ correction can be improved.

Computational costs are shown in 8 (right). The $Z_\emptyset$ and the CVM-Loop8 are the most expensive approaches. $Z_\emptyset$ is more efficient than CVM-Loop8, but only for number of edges up to 250. TreeEP and CVM-Loop4 have very similar scaling behavior. Although TreeEP is significantly more efficient than CVM-Loop4, their costs get closer as the size of the graphs increases. JuncTree is more efficient than all approximate methods except BP, which is the most efficient algorithm.

4. Discussion

We have presented an algorithm based in the loop calculus framework for approximate inference on planar graphical models defined on binary variables. The proposed approach corrects the estimate of the partition function provided by BP.

We have illustrated the algorithm using the Ising model. This model allows to analyze how the performance of the method varies according to the complexity of the problem. Given that exact results are obtained in polynomial time for the case of zero local fields, one would expect less accurate approximations for problems with larger local fields. However, the outcomes of our experiments on grids show that significant improvements over BP are always obtained for large enough single variable potentials. The quality of this correction degrades as external fields become smaller. This suggests that in the Ising model, the difficulty of the inference task changes abruptly from very easy problems with no local fields to very hard problems with small local fields and then decays again as external fields become larger.

The $Z_\emptyset$ correction turns out to be competitive with other state of the art methods for approximate inference of the partition function. In first place, we have shown that $Z_\emptyset$ is much more accurate than methods which provide upper bounds of Z such as TRW or PDC. This illustrates that such methods come at the cost of less accurate approximations. We have also shown that for the case of grids with attractive interactions, the lower bound provided by $Z_\emptyset$ is the most accurate.

Secondly, we found that $Z_\emptyset$ performs much better than treeEP for weak and intermediate couplings and shows competitive results for strong interactions. This is a good result given that TreeEP exploits the sparsity of the underlying graph and planar graphs are known to be sparse.

Generalized belief propagation turned out to be the best algorithm in our numerical experiments. Using a double-loop implementation of CVM and large enough outer clusters we could always improve on the $Z_\emptyset$ approximation. Notice, however, that although in this study larger regions gave better results, this not necessary true in general. For grid graphs, we have shown that to improve the $Z_\emptyset$ correction, one has to select regions including at least loops of length 6 in the graph. For planar graphs with random structure we found more variability in the solutions, but in general using CVM-Loop8 we could improve the results of $Z_\emptyset$.

Finally, we have presented results comparing $Z_\emptyset$ with TLSBP, which is another algorithm for the loop series of BP which considers the length of the loops as the tuncation principle. On one hand, the calculation of $Z_\emptyset$ involves a *resummation* of many loop terms which in the case of TLSBP are summed individually. This favours the $Z_\emptyset$ approach. On the other hand, $Z_\emptyset$ is restricted to the class of 2-regular loops whereas TLSBP also accounts for terms corresponding to more complex loop structures in which nodes can have degree larger than 2. This favours the TLSBP approach. Overall, for planar graphs, we have shown evidence that the $Z_\emptyset$ approach is better than TLSBP when the size of the graphs is not very small. We emphasize, however, that TLSBP can be applied to non-planar binary graphical models too.

Currently, the shortcoming of the presented approach is the implementation constant which determines the final computational cost. However, since the critical part of the algorithm is the pfaffian calculation and not the algorithms used to obtain the extended graphs and the associated matrices, it is easy to devise more efficient methods than the one used here. First, methods exist for the calculation of the determinant/pfaffian which can reduce the cost from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^{3/2})$. Second, the pfaffian involving the binary matrix $\hat{B}$ could be computed more efficienly using a bit-matrix representation as in (Schraudolph and Kamenetsky, 2008). Another posibility would be to correct the sign using other strategy which does not require the pfaffian of $\hat{B}$, thus reducing the cpu time to the half. All these technical issues are under current research.

We have focus on inference problems defined on planar graphs with symmetric interactions and to make them difficult we have introduced local field potentials. It is important to emphasize that the presented algorithm can be directly applied also to models where interactions between variables are not symmetric. This implies that both conditions mentioned in the introduction section can be relaxed and still having an approximate algorithm and differs from other approaches that use the Fisher & Kasteleyn approach (Globerson and Jaakkola, 2007; Schraudolph and Kamenetsky, 2008). Although planarity may appear a severe restriction, we emphasize that planar graphs appear in many contexts such as computer vision and image processing, magnetic and optical recording, or network routing and logistics.

Of interest are extensions of this work for non-planar graphs. In the spirit of Globerson and Jaakkola (2007) one can think about other types of spanning subgraphs more general than "easy" planar graphs for which exact computation can be performed using perfect

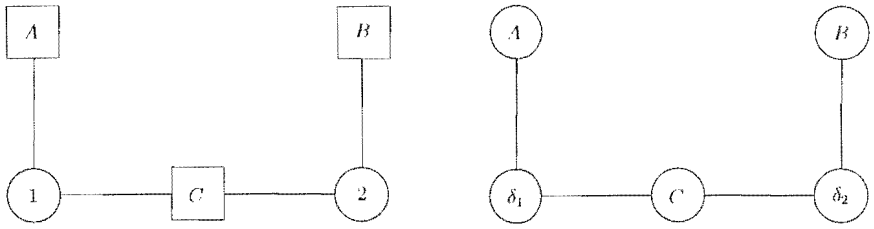

Figure 9: **(a)** A small factor graph $\mathcal{G}_{\mathcal{F}}$ and **(b)** an equivalent Forney graph $\mathcal{G}$.

matching. The correction $Z_{\emptyset}$ can be an accurate approximation of this spanning subgraphs and the resulting approximation method would also provide bounds of the exact result.

Acknowledgments

We acknowledge J. M. Mooij for providing the libDAI framework and A. Windsor for the planar graph functions of the boost graph library. We also thank V. Chernyak, Jason K. Johnson and N. Schraudolph for interesting discussions and A. Globerson for providing the matlab sources of PDC. This research is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant BSIK03024. The work at LANL was carried out under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396.

Appendix A: Converting a factor graph to a Forney Graph.

A probabilistic model is usually represented as a bayesian network or a markov random field. Since bipartite factor graphs subsume both models, we show here how to convert a factor graph model defined on binary variables to a more general Forney graph representation for which the presented algorithm can be directly applied.

In a bipartite factor graph $\mathcal{G}_{\mathcal{F}} = (\mathcal{V}_{\mathcal{F}}, \mathcal{E}_{\mathcal{F}})$ the set $\mathcal{V}_{\mathcal{F}}$ is composed of a set of variable nodes $\mathcal{I}$ and a set of factor nodes $\mathcal{J}$. Each variable node $i \in \mathcal{I}, i := \{1, 2, \dots\}$ represents a variable which takes values $\sigma_i = \{\pm 1\}$. We label factor nodes using capital letters so that $a = \{A, B, \dots\}, a \in \mathcal{J}$ denotes a factor node which has an associated function $f_a(\boldsymbol{\sigma}_a)$ defined on a subset of variables $\bar{a} \in \mathcal{I}$. An (undirected) edge exists between two nodes $(a, i) \in \mathcal{E}_{\mathcal{F}}$ if $i \in \bar{a}$.

Given $\mathcal{G}_{\mathcal{F}}$, a direct way to obtain an equivalent Forney graph $\mathcal{G}$ is: first, we create a node $\delta_i \in \mathcal{V}$ for each variable node $i \in \mathcal{V}_{\mathcal{F}}$. Second, we associate a new binary variable $\delta_i a$ with values $\sigma_{\delta_i a} = \{\pm 1\}$ to edges $(\delta_i, a) \in \mathcal{E}$. Nodes $\delta_i \in \mathcal{V}$ are *equivalent factor nodes* and have associated the characteristic function $\delta_i(\boldsymbol{\sigma}_a) = 1$ if $\sigma_{\delta_i a} = \sigma_{\delta_i b}, \forall a, b \in \bar{\delta}_i$ and zero otherwise. Finally, factor nodes $c \in \mathcal{V}_{\mathcal{F}}$ correspond to the same factor nodes c in $\mathcal{V}$ but defined on the new variables $\delta_i c, \forall i \in \bar{c}$.

Figure 9 shows an example of this transformation. Notice that, although we impose an direction in the edge labels, they remain undirected: $(\delta_i, a) = (a, \delta_i)$, $\forall \delta_i, a \in \mathcal{V}$. The joint distribution of $\mathcal{G}_\mathcal{F}$ is related to the joint distribution of $\mathcal{G}$ by:

$$\frac{1}{Z} f_A(\sigma_1) f_B(\sigma_2) f_C(\sigma_1, \sigma_2) \equiv \frac{1}{Z} f_A(\sigma_{\delta_1 A}) f_B(\sigma_{\delta_2 B}) f_C(\sigma_{\delta_1 C}, \sigma_{\delta_2 C}) f_{\delta_1}(\sigma_{\delta_1 A}, \sigma_{\delta_1 C}) f_{\delta_2}(\sigma_{\delta_2 B}, \sigma_{\delta_2 C}).$$

Once $\mathcal{G}$ has been generated following the previous procedure it may be the case that nodes $\delta_i \in \mathcal{V}$ have degree 3 or larger. This happens if a variable i appears in more than 3 factor nodes in $\mathcal{G}_\mathcal{F}$. It is easy to convert $\mathcal{G}$ to a graph were all δ_i nodes have maximum degree 3 by introducing new auxiliary variables $\delta_{i_1}, \delta_{i_2}, ...$ and equivalent nodes. For instance, if variable $i \in \mathcal{V}_\mathcal{F}$ appears in 4 factors A, B, C, D:

$$f_{\delta_i}(\sigma_{\delta_i A}, \sigma_{\delta_i B}, \sigma_{\delta_i C}, \sigma_{\delta_i D}) \equiv f_{\delta_{i_1}}(\sigma_{\delta_i A}, \sigma_{\delta_i B}, \sigma_{\delta_{i_1}}) f_{\delta_{i_2}}(\sigma_{\delta_{i_1}}, \sigma_{\delta_i C}, \sigma_{\delta_i D}).$$

Also notice that $\mathcal{G}$ may have more loops than $\mathcal{G}_\mathcal{F}$.

References

F. Barahona. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241–3253, 1982. URL `http://stacks.iop.org/0305-4470/15/3241`.

M. Chertkov and V. Y. Chernyak. Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(06):P06009, 2006a.

M. Chertkov and V. Y. Chernyak. Loop calculus helps to improve belief propagation and linear programming decodings of LDPC codes. In *invited talk at 44th Allerton Conference*, September 2006b.

M. Chertkov, V. Y. Chernyak, and R. Teodorescu. Belief propagation and loop series on planar graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(05): P05003 (19pp), 2008. URL `http://stacks.iop.org/1742-5468/2008/P05003`.

G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, Boston, Massachussetts, July 2006. AUAI Press.

M. Fisher. On the dimer solution of the planar Ising model. *Journal of Mathematical Physics*, 7(10):1776–1781, 1966.

Jr. Forney, G.D. Codes on graphs: normal realizations. *Information Theory, IEEE Transactions on*, 47(2):520–548, Feb 2001. ISSN 0018-9448. doi: 10.1109/18.910573.

B. J. Frey and D. J. C. MacKay. A revolution: Belief propagation in graphs with cycles. In *NIPS*, 1997.

G. Galbiati and F. Maffioli. On the computation of Pfaffians. *Discrete Appl. Math.*, 51(3): 269–275, 1994. ISSN 0166-218X. doi: http://dx.doi.org/10.1016/0166-218X(92)00034-J.

A. Globerson and T. S. Jaakkola. Approximate inference using planar graph decomposition. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 473–480. MIT Press, Cambridge, MA, 2007.

V. Gómez, J. M. Mooij, and H. J. Kappen. Truncating the loop series expansion for belief propagation. *J. Mach. Learn. Res.*, 8:1987–2016, 2007. ISSN 1533-7928.

T. Heskes, K. Albers, and H. J. Kappen. Approximate inference and constrained optimization. In *Proceedings of the 19th Annual conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 313–320, San Francisco, CA, 2003. Morgan Kaufmann Publishers.

M. Karpinski and W. Rytter. Fast parallel algorithms for graph matching problems. pages 164–170. Oxford University Press, USA, 1998.

P. W. Kasteleyn. Dimer statistics and phase transitions. *Journal of Mathematical Physics*, 4(2):287–293, 1963. doi: 10.1063/1.1703953. URL `http://link.aip.org/link/?JMP/4/287/1`.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical society. Series B-Methodological*, 50(2):154–227, 1988.

H.-A. Loeliger. An introduction to factor graphs. *Signal Processing Magazine, IEEE*, 21 (1):28–41, Jan. 2004. ISSN 1053-5888. doi: 10.1109/MSP.2004.1267047.

T. Minka and Y. Qi. Tree-structured approximations by Expectation Propagation. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

J. M. Mooij. libDAI, 2008. `http://mloss.org/software/view/77/`.

J. M. Mooij and H. J. Kappen. On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(11):P11012, 2005.

K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 467–475, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco, CA, 1988. ISBN 1558604790.

G. Schaeffer. Random sampling of large planar maps and convex polyhedra. In *Annual ACM Symposium on Theory of Computing*, pages 760–769, Atlanta, GA, 1999. ACM.

N. Schraudolph and D. Kamenetsky. Efficient exact inference in planar Ising models. In *Advances in Neural Information Processing Systems 22*. MIT Press, Cambridge, MA, 2008.

E. Sudderth, M. Wainwright, and A. Willsky. Loop series and bethe variational bounds in attractive graphical models. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1425–1432. MIT Press, Cambridge, MA, 2008.

M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *IEEÈ Transactions on Information Theory*, 51(7):2313–2335, July 2005.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13 (NIPS-00)*, pages 689–695, December 2000.

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