Information Processing Letters 112 (2012) 449–456

Contents lists available at SciVerse ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Recursive sum–product algorithm for generalized outer-planar graphs

Qiang Cheng [a,*], Feng Chen [a], Wenli Xu [a], Song Wang [b]

[a] Tsinghua National Laboratory for Information Science and Technology, Department of Automation, Tsinghua University, Beijing 100084, PR China
[b] Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29208, USA

ARTICLE INFO

Article history:
Received 26 June 2011
Received in revised form 1 March 2012
Accepted 2 March 2012
Available online 6 March 2012
Communicated by J. Torán

Keywords:
Graphical models
Inference algorithm
Approximation algorithms
Generalized outer-planar graph
Recursive sum–product algorithm

ABSTRACT

Inference on cyclic graphs is one of the most important problems in the applications of graphical models. While exact inference is NP-hard on general cyclic graphs, it has been found that exact inference can be achieved with a computational complexity as low as $\mathcal{O}(Nm^3)$ on the outer-planar graph, which is a special kind of cyclic graph. In this paper, we introduce a new kind of cyclic graph, the generalized outer-planar (GOP) graph, which is more general than the outer-planar graph and show that the exact inference on the GOP graphs can be achieved in $\mathcal{O}(Nm^3)$ by a recursive sum–product (RSP) algorithm. RSP exploits the property of GOP graphs that the faces are reducible, and brings a "face elimination" procedure to compute the marginals exactly. Furthermore, RSP can be implemented on general cyclic graphs to obtain approximate marginals. Experimental results show the effectiveness of approximate RSP on various graphical models.

1. Introduction

Graphical models have been widely studied and used as powerful tools in many fields for modeling uncertainty, such as image processing [1], natural language processing [2], computational biology [3], and statistical machine learning [4]. Inference problems on graphical models, including computing partition functions, marginal distributions and most probable configurations, play an important role in many of these applications [5]. Exact inference on general graphs is NP-hard, and therefore it is usually infeasible to achieve in practice. However, for some special cyclic graphs, exact inference can be done with efficient algorithms. The belief propagation algorithm (BP) is an exact inference algorithm, with $\mathcal{O}(Nm^2)$ complexity (N is the number of variables, and m is the maximal number of states of the variables), for tree-structured graphical models [6]. The junction tree algorithm is an exact inference approach for graphs with cycles, with complexity exponential to the size of the maximal clique in the induced junction tree [4]. Its complexity is $\mathcal{O}(Nm^3)$ for outer-planar graphs and may be higher for other cyclic graphs. There are many approximate algorithms, such as the loopy belief propagation algorithm, the generalized belief propagation algorithm [7] and the junction graph method [8], which can obtain approximate inference results with lower complexity compared with that of exact inference.

In this paper, we introduce a new kind of cyclic graph, the *generalized outer-planar* (GOP) graph, and develop an exact inference algorithm for such graphs. GOP graphs are more general than outer-planar graphs. The computational complexity of off-the-shelf exact inference methods may be more than $\mathcal{O}(Nm^3)$ on GOP graphs. We propose a *recursive sum–product* (RSP) algorithm for exact inferences on GOP graphs, which can keep the lowest computational complexity, i.e., $\mathcal{O}(Nm^3)$. The property of a GOP graph is that its variables can be eliminated by using two simple operations. Making use of this property, RSP is developed as a "face elimination" procedure, which eliminates the faces of GOP graphs one after another recursively. More specifically, the inference is obtained from each individual face and later integrated by using the structural overlap among different faces in RSP. RSP obtains exact inference on GOP graphs with the complexity of $\mathcal{O}(Nm^3)$, which is

[*] Corresponding author. Tel.: +86 1062797145; fax: +86 1062786911.
E-mail address: cheng-q09@mails.tsinghua.edu.cn (Q. Cheng).

0020-0190/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.ipl.2012.03.001

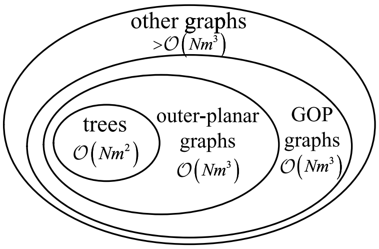

Fig. 1. Relations of different kinds of graphs and the computational complexity on these graphs for exact inference.

lower than the complexities of other exact inference methods.

The relations of different kinds of graphs and the computational complexity on these graphs for exact inference are shown in Fig. 1. Specifically, RSP can obtain exact inference on those GOP graphs, with a lower complexity.

The remainder of this paper is organized as follows. The background and some notations are introduced in Section 2. The sum–product procedure for one cycle is introduced in Section 3. Generalized outer-planar graphs are introduced in Section 4. The recursive sum–product algorithm (RSP) on GOP graphs is developed in Section 5. In Section 6, we develop an approximate version of RSP for general cyclic graphs. In Section 7, experiments on a variety of graphs are conducted to show the effectiveness of approximate RSP.

2. Preliminaries

In this section, we overview the concepts of *outer-planar graph* and *Markov random field* that are important for our algorithm development. We denote an undirected graph by $G = (V, E)$, where $V = \{1, 2, \dots, n\}$ is the set of nodes and $E \subset V \times V$ is the set of edges. When G is used for probabilistic inference, each node in this graph represents a random *variable*.

2.1. Face and outer-planar graph

We first give the description of the *face* of a planar graph. A planar graph G partitions the rest of the plane into a number of connected open sets, which are called the **faces** of G. Each planar graph has exactly one unbounded face, called the **outer face** or **external face** [9]. In our paper, "face" is used to denote the bounded face if there is no confusion, and "face" also denotes the set of variables on its boundary. Fig. 2 shows four faces f_1, f_2, f_3, f_4 and an outer face f_0 in a planar graph. The outer-planar graph is defined as follows.

An **outer-planar graph** G is a planar graph, where all nodes lie on the outer face [10]. The graph shown in Fig. 5(a) is an outer-planar graph, while the graph in Fig. 5(b) is not an outer-planar graph, since node 5 does not lie on the outer face.

2.2. Graphical models

An undirected graphical model (also known as a Markov random field (MRF)) is a set of random variables (as well as a set of undirected links) that have a Markov property described by the graph G. Let X_s be the random variable related to node $s \in V$ in graph G, and x_s take values in the discrete space $\mathcal{X}_s = \{0, 1, \dots, m_s - 1\}$. Let $\mathbf{x} = \{x_s \mid s \in V\}$ take values in the product space $\mathcal{X}^N = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_N$, where $N = |V|$ is the number of nodes in G. $\mathbf{x}$ is called a state of $\mathbf{X}$, where $\mathbf{X} = \{X_s \mid s \in V\}$. A *clique* of a graph G is a subset of nodes in which there exists an edge between each pair of nodes in the subset, and a clique potential is a non-negative real valued function defined on the set of nodes in a clique [5]. The pairwise MRF is an MRF where all the cliques have at most two variables. The probability distribution of a pairwise MRF $G = (V, E)$ can be expressed as a product of clique potentials:

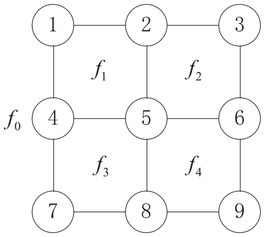

Fig. 2. A planar graph and its faces.

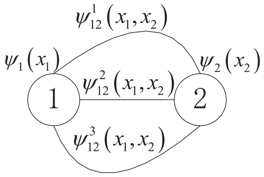

Fig. 3. Illustrations of parallel edges.

$$p(\mathbf{x}) = \frac{1}{Z_\psi} \prod_{s \in V} \psi_s(x_s) \prod_{(s,t) \in E} \psi_{st}(x_s, x_t), \tag{1}$$

where Z_ψ denotes the partition function, $\psi_s(x_s)$ and $\psi_{st}(x_s, x_t)$ denote the potentials corresponding to the clique $\{x_s\}$ and $\{x_s, x_t\}$.

3. Sum–product on one cycle

If a graph contains only one face (not including the outer face), there is just one cycle in the graph. When using the sum–product procedure for the graph with only one cycle, we first reduce the cycle to parallel edges between two variables of this cycle, and then compute the distribution of these two variables. If there are two or more edges connecting the same two nodes, such edges are called **parallel edges**. One example of parallel edges is shown in Fig. 3, where there are three parallel edges between nodes 1 and 2. The potential functions on parallel edges can be reduced to one potential function on one edge by the dot product:

$$\psi_{st}(x_s, x_t) = \prod_{i \in PE(st)} \psi_{st}^i(x_s, x_t), \tag{2}$$

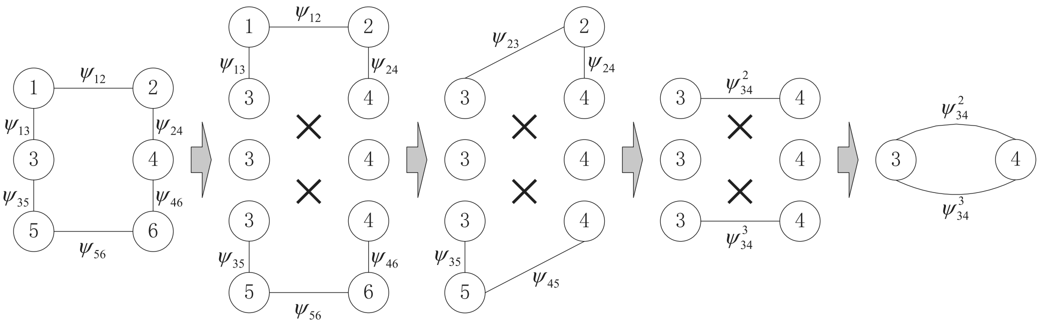

Fig. 4. The graph representation of the sum–product procedure on one cycle.

where $PE(st)$ denotes the set of parallel edges between nodes s and t. The computational complexity of Eq. (2) is linear to the size of the potential function.

In the following, we will discuss how to reduce the distributions on one cycle to those on parallel edges. Given a pairwise MRF with one cycle, its probability distribution is defined by Eq. (1). In the following discussion, we discard the branches adjacent to the cycle, since the computation on them can be accomplished similarly. We focus on computing the joint marginal distribution of the two nodes x_u and x_v on this cycle, and the following computation is needed.

$$p_{uv}(x_u, x_v) \propto \sum_{\mathbf{x}\setminus\{x_u,x_v\}} \prod_{s\in V} \psi_s(x_s) \prod_{(s,t)\in E} \psi_{st}(x_s, x_t). \qquad (3)$$

When the graph is a cycle, the above equation can be computed by the iteration

$$\psi_{s-1,s+1}(x_{s-1}, x_{s+1}) = \sum_{x_s} \psi_{s-1,s}(x_{s-1}, x_s)\psi_{s,s+1}(x_s, x_{s+1}). \qquad (4)$$

The above process eliminates variable x_s and the two edges adjacent to x_s, and yields a new edge between x_{s-1} and x_{s+1}. $\psi_{s-1,s+1}(x_{s-1}, x_{s+1})$ is a new potential defined on edge $\{s-1, s+1\}$. The computational complexity of the above equation is $\mathcal{O}(m^3)$. Fig. 4 illustrates the steps graphically.

To conclude the sum–product on one cycle, we define two operations:

Operation-A: joining parallel edges (i.e., Eq. (2)).
Operation-B: joining two edges which appear as the only neighbors of a given node (i.e., Eq. (4)).

By iterating the above two operations on a cycle, exact inference results can be obtained. This is concluded in the following theorem.

Theorem 1. *For the graphical model consisting of one cycle, marginal distributions (together with partition functions) can be computed exactly by iterating Operation-A and Operation-B with the computational complexity of $\mathcal{O}(Nm^3)$.*

We then introduce the AB-reducible property of a graph.

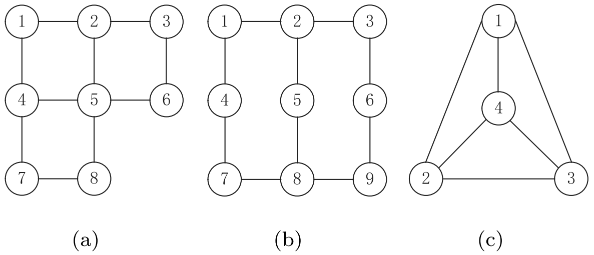

Fig. 5. (a) An outer-planar graph, (b) a generalized outer-planar graph, (c) a planar graph that is not GOP.

Definition 1. A graphical model has the AB-reducible property if exact results can be obtained by just iterating Operation-A and Operation-B on this graphical model.

Obviously, a graph with only one cycle has the AB-reducible property. A following question is: which kind of graph is also AB-reducible? This will be discussed in the following section.

4. Generalized outer-planar graphs

In this section, we will introduce a new kind of planar graph, generalized outer-planar (GOP) graph. GOP graphs are generalizations of outer-planar graphs, on which exact inference can be done with the same complexity as outer-planar graphs.

The generalized outer-planar graph is defined using the two operations: Operation-A and Operation-B.

Definition 2. Given a planar graph G, if exact inference can be done by iteratively and alternately implementing Operation-A and Operation-B, then this graph is a **generalized outer-planar (GOP) graph**.

Some graphs are given to illustrate this definition. The graph in Fig. 5(b) is a GOP graph since all the variables can be eliminated by running Operation-A and Operation-B. The graph shown in Fig. 5(a) is an outer-planar graph and it is also a GOP graph. The graphs in Fig. 5(c) and Fig. 2 are not GOP graphs. The following corollary shows that GOP graphs are generalization of outer-planar graphs.

Corollary 2. *All outer-planar graphs are generalized outer-planar graphs.*

Proof. We accomplish the proof by proving the chordal graph [11] of an outer-planar graph is a GOP graph. A chordal graph is a graph where each cycle with four or more nodes has a chord. A junction tree [8] can be constructed corresponding to an outer-planar graph, where each clique contains at most three nodes. We then construct a chordal graph w.r.t. the junction tree by adding some edges on the outer-planar graph. Implementing Operation-A and Operation-B on the chordal graph is the same as variable elimination on the junction tree, so exact results can be obtained. Then the chordal graph is a generalized outer-planar graph. □

There are many cases where the graphs are not outer-planar graphs but are GOP graphs. Exact inference on these graphical models can be accomplished by the junction tree algorithm with complexity of $\mathcal{O}(Nm^3)$. But additional operations should be implemented to construct a junction tree. In the following section, we will introduce a recursive sum–product algorithm for GOP graph, which can do exact inference on these graphs with lower computational complexity.

5. Recursive sum–product algorithm

In this section, we will introduce the recursive sum–product algorithm for exact inference on GOP graphs. The recursive sum–product algorithm (RSP) is developed by implementing Operation-A and Operation-B in a recursive form.

Our technique to compute the marginals on one cycle is first to reduce the cycle to parallel edges and then to compute the marginals of these parallel edges. The same technique can be used to compute the marginals on GOP graphs, i.e., we iteratively and alternately run Operation-A and Operation-B to reduce the faces into parallel edges. This can also be understood from "face elimination", where we eliminate faces one by one by implementing A- and B-operations.

This "face elimination" procedure is designed as a recursive algorithm, in which a face is eliminated only after the faces adjacent to it have been eliminated. The steps are as follows. We find one path on a face containing variables x_u, x_v, and then eliminate all the variables on this path except for x_u, x_v, such as the computation of Eq. (3). Before this, if any other faces are adjacent to this path, they must be eliminated first. This process forms a recursive scheme, since before eliminating one face, the faces adjacent to it must be eliminated first. The algorithm is summarized in Algorithm 1, and the following example is given to illustrate the algorithm.

Example 1. A graphical model is shown in Fig. 6, and we will compute the joint marginal distribution $p_{45}(x_4, x_5)$. The steps are as follows:

(1) Eliminate variables $x_3, x_7, x_2, x_8, x_1, x_9$ using Eq. (4).
(2) Compute the potential function $\psi_{46}(x_4, x_6)$ using Eq. (2).
(3) Eliminate variables x_6 using Eq. (4).
(4) Return $p_{45}(x_4, x_5)$ using Eq. (2).

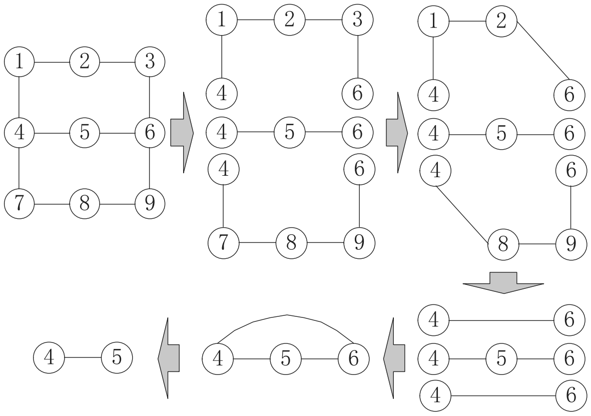

Fig. 6. The graph illustration of RSP.

Algorithm 1 Recursive sum–product algorithm.

```
Input: A GOP graphical model $G = (V, E)$.
Output: The joint marginal distribution $p_{uv}(x_u, x_v)$.
1: Let $S_{uv}$ denote the variables in the faces which contains $u$ and $v$.
   Eliminate variables in $S_{uv}$ using Eq. (4) and Eq. (2) if possible. Return
   $p_{uv}(x_u, x_v)$.
2: if $S_{uv} \neq \emptyset$ then
3:     For each edge $(i, j)$ in $S_{uv}$, let $u \leftarrow i$ and $v \leftarrow j$, go to Step 1.
4: end if
```

As described in Algorithm 1, RSP algorithm implements iterations on two operations: Operation-A and Operation-B. Thus, GOP graphs have the AB-reducible property. Algorithm 1 is represented in a recursive form. In fact, we can obtain $p_{uv}(x_u, x_v)$ by directly iterating Operation-A and Operation-B. The advantage of the recursive representation is that it will help explore the approximation version of RSP algorithm in the next section.

The above discussion discards the other branches connecting to the cycles, and can be extended to general cyclic graphs easily if each branch is regarded as a special "cycle". The computational complexity of RSP is $\mathcal{O}(Nm^3)$, where $m = \max\{m_s \mid s \in V\}$ and m_s is the number of states of variable x_s.

The recursion in RSP will be endless for the graphs that are not GOP. In the following section, we will discuss how to obtain approximate marginals on non-GOP graphs using the idea of RSP.

6. Approximate RSP

For the cyclic graphs which are not GOP graphs, such as the graph in Fig. 5(c), Algorithm 1 will be endless, and RSP cannot give a result. In this section, we further extend RSP, by ending Algorithm 1 properly, to achieve an approximate result on such graphs.

The reason for the failure of RSP on non-GOP graphs is that the condition for implementing Operation-B, i.e., no more than two edges appear as the neighbors of a given node, cannot be satisfied, such as the graph in Fig. 5(c). Thus, the variable cannot be eliminated using only Operation-B, i.e., Eq. (4). Our idea is to construct a GOP graph to approximate the original graph. We break up some edges and construct an approximate GOP graph directly from the original graph. This idea is similar to that

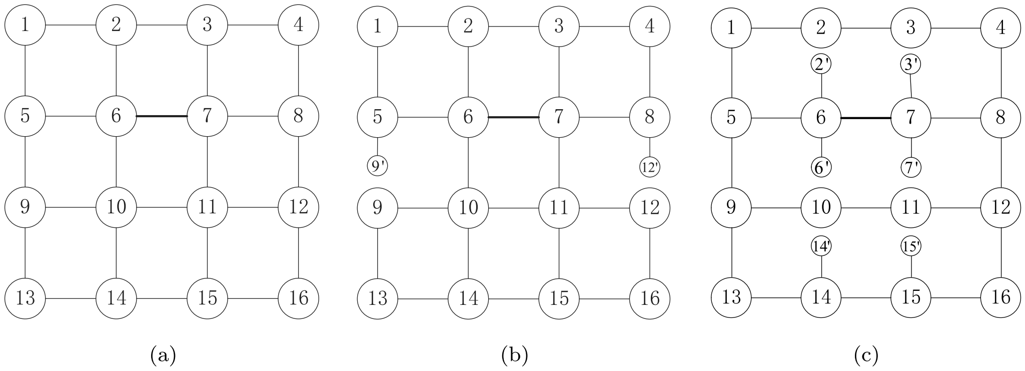

Fig. 7. The original graph (a) and the covering GOP graphs (b and c).

of covering tree [12], i.e., if the condition for implementing Operation-B is unsatisfied, we break up one or more edges by adding duplicate copies of some nodes as necessary. We name such approximate GOP graph as the covering GOP graph.

We give a heuristic method to construct a covering GOP graph. First, some edges are removed from the original graph to make all the nodes on the boundary of the outer face. Then, the removed edges are replaced by the corresponding auxiliary edges, one end of which has a copy of the original node. The potential functions on these auxiliary edges are the same as those of the removed edges. This way, we construct a covering outer-planar graph. Two covering GOP graphs of Fig. 7(a) are shown in Fig. 7(b) and Fig. 7(c). Generally speaking, removing fewer edges will lead to more accurate approximations. A greedy algorithm is given to construct a covering GOP graph, as shown in Algorithm 2.

Algorithm 2 Constructing a covering GOP graph.

```
Let $F_{in}$ be the set of nodes that are not on the boundary of the outer face.
while $F_{in}$ is not empty do
    Let $F_{out}$ be the set of nodes on the boundary of the outer face, and $F_{in}$ be the set of the other nodes.
    Add duplicate copies of a node in $F_{out}$ to makes the most nodes in $F_{in}$ on the boundary of the outer face.
end while
```

The covering GOP graph of G is not unique, and the two graphs in Fig. 7(b) and Fig. 7(c) both are covering GOP graphs of the graph shown in Fig. 7(a). When computing the joint marginal distribution of a pairwise nodes $\mu_{uv}(x_u, x_v)$ (we use $\mu_{uv}(x_u, x_v)$ to represent the approximate $p_{uv}(x_u, x_v)$), one can select a covering GOP graph with the most edges among x_u, x_v and their Markov boundary variables. The GOP graph in Fig. 7(c) has more edges among x_6, x_7 and their Markov boundary variables than the GOP graph in Fig. 7(b).

After the covering GOP graph is constructed, we can run RSP algorithm on it, and obtain approximate results. However, RSP can also be run on the original graph without constructing a covering GOP graph. We embed the process of constructing a covering GOP graph into the RSP algorithm, and yield an approximate version of RSP algorithm. The reason for the endlessness of RSP is that the condition for implementing Operation-B cannot be satisfied. To end RSP algorithm properly, the maximal number of iterations for Step 3 in Algorithm 1 is first defined, denoted by $I_{\max}$. Then, when Step 3 is iterated $I_{\max}$ times, Operation-B is approximately implemented even if three or more edges appear as the neighbors of a node. This approximate version of RSP is the same as running RSP on a covering GOP graph.

After the pairwise distributions $\mu_{uv}(x_u, x_v)$ are obtained, we need to compute the single variable distributions. Since the pairwise distributions are not exact, the marginalizations of different distributions w.r.t. the same variable may not equal to each other. Obviously, averaging these marginalizations is a direct method. Another approach to obtain the marginals of single node is by minimizing the variational free energy, such as the Bethe free energy or tree-reweighted Bethe free energy [13]. With the pairwise distributions known, the tree-reweighted Bethe free energy is:

$$\begin{aligned} F_{RSP} = \sum_{s\in V}\Bigg(\rho_s \sum_{j\in\chi_s} \mu_{s;j}\log\mu_{s;j} \\ - \sum_{t\in N(s)} \rho_{st} \sum_{(j,k)\in\chi_s\times\chi_t} \mu_{st;jk}\log\mu_{s;j}\Bigg) \\ - \sum_{s\in V}\sum_{j\in\chi_s}\mu_{s;j}\ln\psi_{s;j}(x_s) + C \end{aligned} \tag{5}$$

where $N(s)$ represents the set of the neighbors of nodes s, C is a constant, $\forall s \in V,\ \rho_s > 0,\ \forall(s,t) \in E,\ \rho_{st} > 0$, and

$$\mu_{s;j} = u_s(X_s = j), \quad \forall j \in \chi_s,$$

$$\mu_{st;jk} = u_{st}(X_s = j, X_t = k), \quad \forall (j,k) \in \chi_s \times \chi_t.$$

The minimization problem is:

$$\begin{aligned} \min \quad & F_{RSP}, \\ \text{s.t.} \quad & \sum_{x_s} \mu_s(x_s) = 1, \quad \forall s \in V, \end{aligned} \tag{6}$$

In F_{RSP}, ρ_s and ρ_{st} can be set as described in [14] to obtain better approximations. With pairwise distributions $\mu_{uv}(x_u, x_v)$ known, $\mu_s(x_s)$ can be computed by solving Eq. (6). F_{RSP} is a convex function w.r.t. $\mu_s(x_s), s \in V$, and

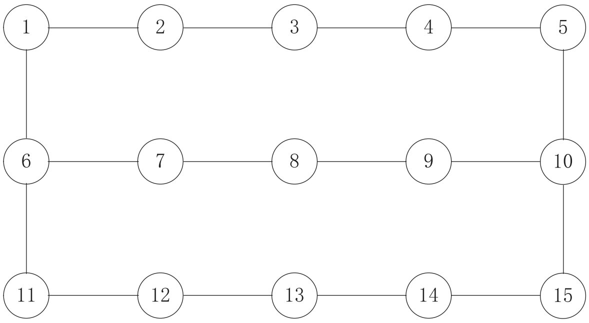

Fig. 8. A GOP graphical model.

there are many different algorithms for solving this problem, such as conditional gradient method [15].

7. Experiments

In this section, we first give an example to compare the exactness and the complexity of RSP and several other algorithms on GOP graphs, and then compare the performance of the approximate RSP to the performance of loopy belief propagation (LBP) and tree reweighted belief propagation (TRBP) algorithms, since they both are well-known efficient inference methods with computational complexity of the same order.

Take the GOP graphical model in Fig. 8 for example. RSP can obtain the exact marginals of this graph with complexity of $\mathcal{O}(m^3)$. The junction tree algorithm obtains the exact marginals with the complexity higher than $\mathcal{O}(m^3)$. LBP and TRBP can obtain approximate marginals with lower complexity, i.e., $\mathcal{O}(m^2)$. For GOP graphs, RSP always produces better results than LBP and TRBP, because LBP and TRBP only give approximate results on these graphs, while RSP gives exact results. In the following, we mainly compare the results of approximate RSP, LBP and TRBP on general cyclic graphs.

We use a 10 × 10 Ising model for experiments. This distribution has the form

$$p(\mathbf{x}) \propto \exp\left(\sum_{s\in V}\theta_s x_s + \sum_{(s,t)\in E}\theta_{st}x_s x_t\right),$$

where θ_s, θ_{st} are parameters, $x_s \in \{\pm 1\}$, V is the set of nodes and E is the set of edges. The parameters θ_s are drawn uniformly from $\mathcal{U}[-d_f, d_f]$ where $d_f \in \{0.05, 1\}$. The parameters θ_{ij} are drawn uniformly from $\mathcal{U}[-d_o, d_o]$ or $\mathcal{U}[0, d_o]$ to obtain mixed or attractive interaction potential respectively, $d_o \in \{0.2, 0.4, \ldots, 4\}$. The mean ℓ_1-error $\frac{1}{|V|}\sum_{s\in V}|\mu_{s;1} - p_{s;1}|$ is used to evaluate the results. (The ℓ_1-error for joint marginals is defined similarly.) The results are obtained after averaging 30 trials for each of the two conditions (attractive or mixed) and each setting of the edge strength. Parts of the experiments follow the works of [13,16].

We first compute the marginals of pairwise nodes of the 10 × 10 Ising model using approximate RSP. In order to reduce the computational complexity, we use a subgraph of the GOP graph, rather than covering GOP graph, to approximate the original graph, which removes the variables that are far from the target variables. When computing $\mu_{uv}(x_u, x_v)$, the proper GOP subgraph is shown in Fig. 9. Error comparison of the approximate marginals of pairwise variables is shown in Fig. 10. We can see that this GOP subgraph can give good approximations with high accuracy. The covering GOP graph will lead to more accurate results.

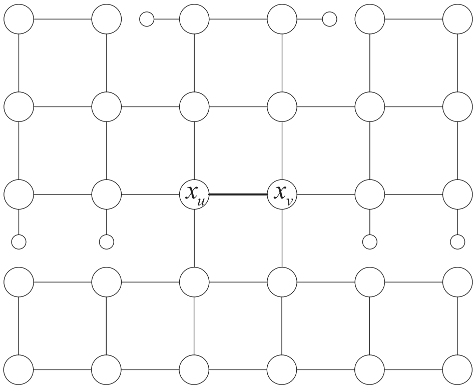

Fig. 9. The structure of subgraphs used for approximations.

The single marginals are computed based on the known approximate pairwise marginals. Since there are different methods to obtain single marginals from joint marginals, we first compare these different methods. We compare the following methods: marginalizing and averaging the edge distributions, minimizing Eq. (5) with Bethe approximation, and minimizing Eq. (5) with tree-reweighted approximation from experiments. The performance is shown in Fig. 11, where “Average”, “Bethe” and “TRW” represent these three methods respectively. We can see that the performances of these three methods are similar. This is because single node marginals are greatly influenced by the pairwise marginals, and the pairwise marginals are just approximations. The computational complexity of the last two methods is higher than the first one, since they need to solve optimization problems. Therefore, the marginalizing and averaging method is a better choice to compute the marginals of one node. Error comparison of different methods is shown in Fig. 12.

From the results of the above experiments, it can be seen that the approximate RSP outperforms TRBP method in most cases, in terms of both pairwise marginals and single marginals. RSP outperforms LBP method on graphical models with small field parameters θ_i when computing marginals. For the graphical models with large field parameters, the performance of RSP is better than that of LBP only under large intersection strengths.

8. Conclusion

This paper proposes a new kind of exact inference framework, based on a recursive sum–product algorithm (RSP), for graphical models with cycle. RSP algorithm can obtain exact marginals on generalized outer-planar graphical models, with low complexity. Moreover, approximate methods based on RSP are explored for those graphical models on which the baseline RSP cannot give a result. In the future, we plan to explore other approximate methods to obtain more accurate results under this framework.

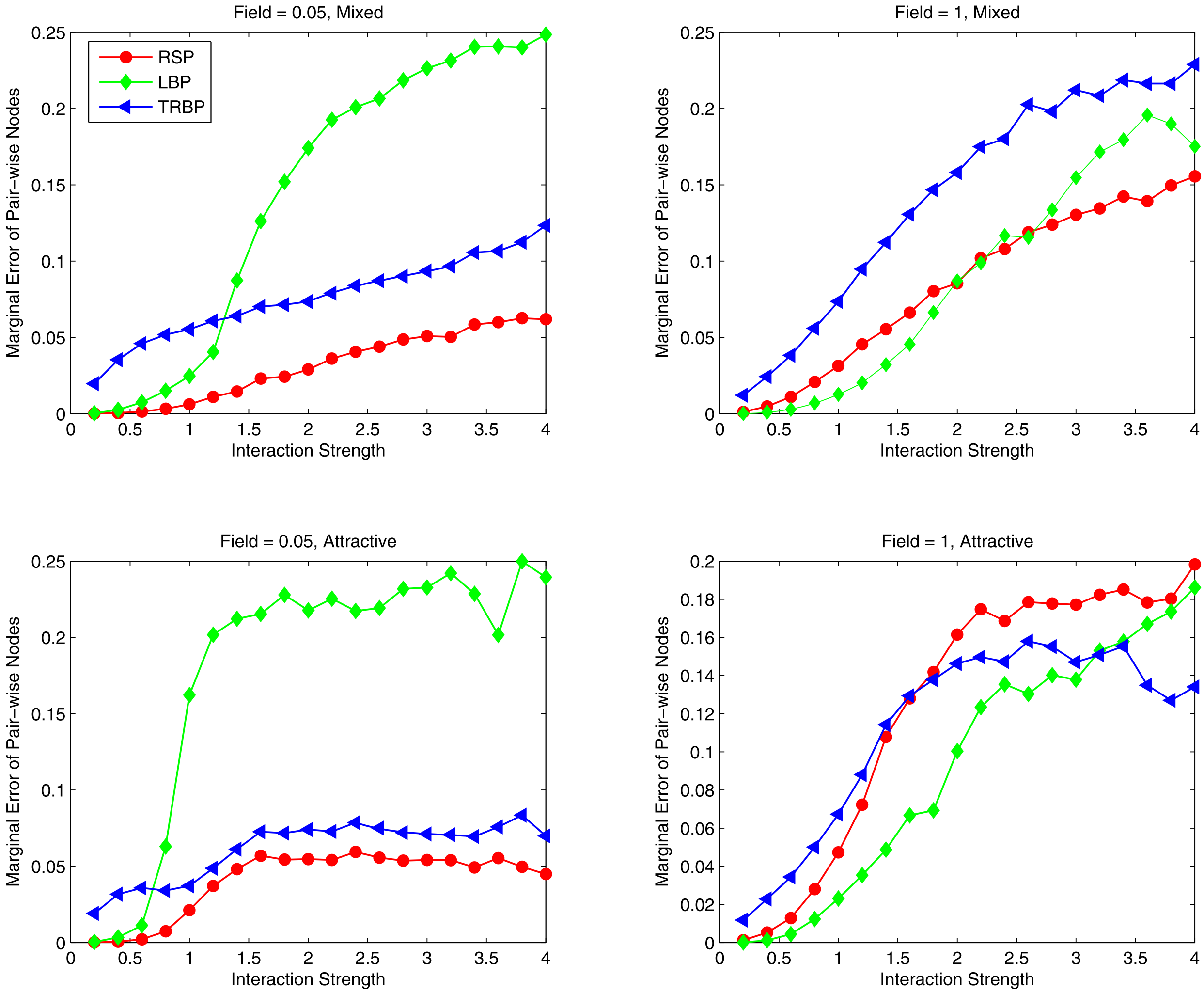

Fig. 10. Comparison of errors in the approximate marginals of pairwise variable of RSP and other approximate inference algorithms.

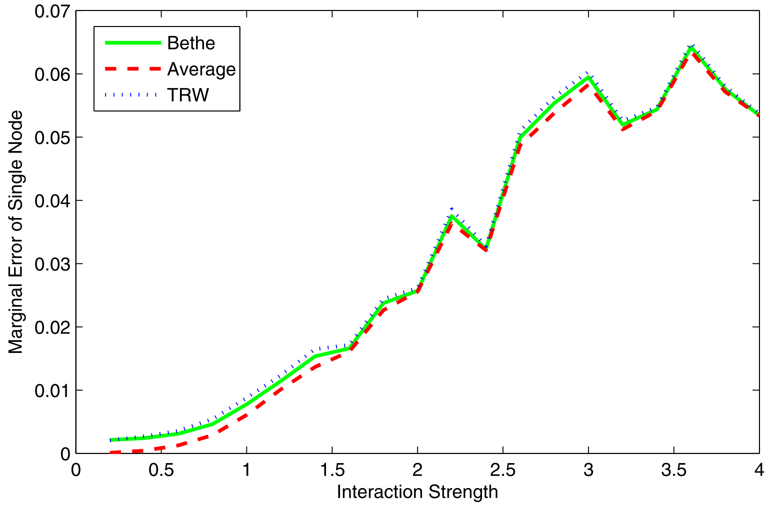

Fig. 11. Performances of different methods to compute the marginals of single node using the known approximate pairwise marginals.

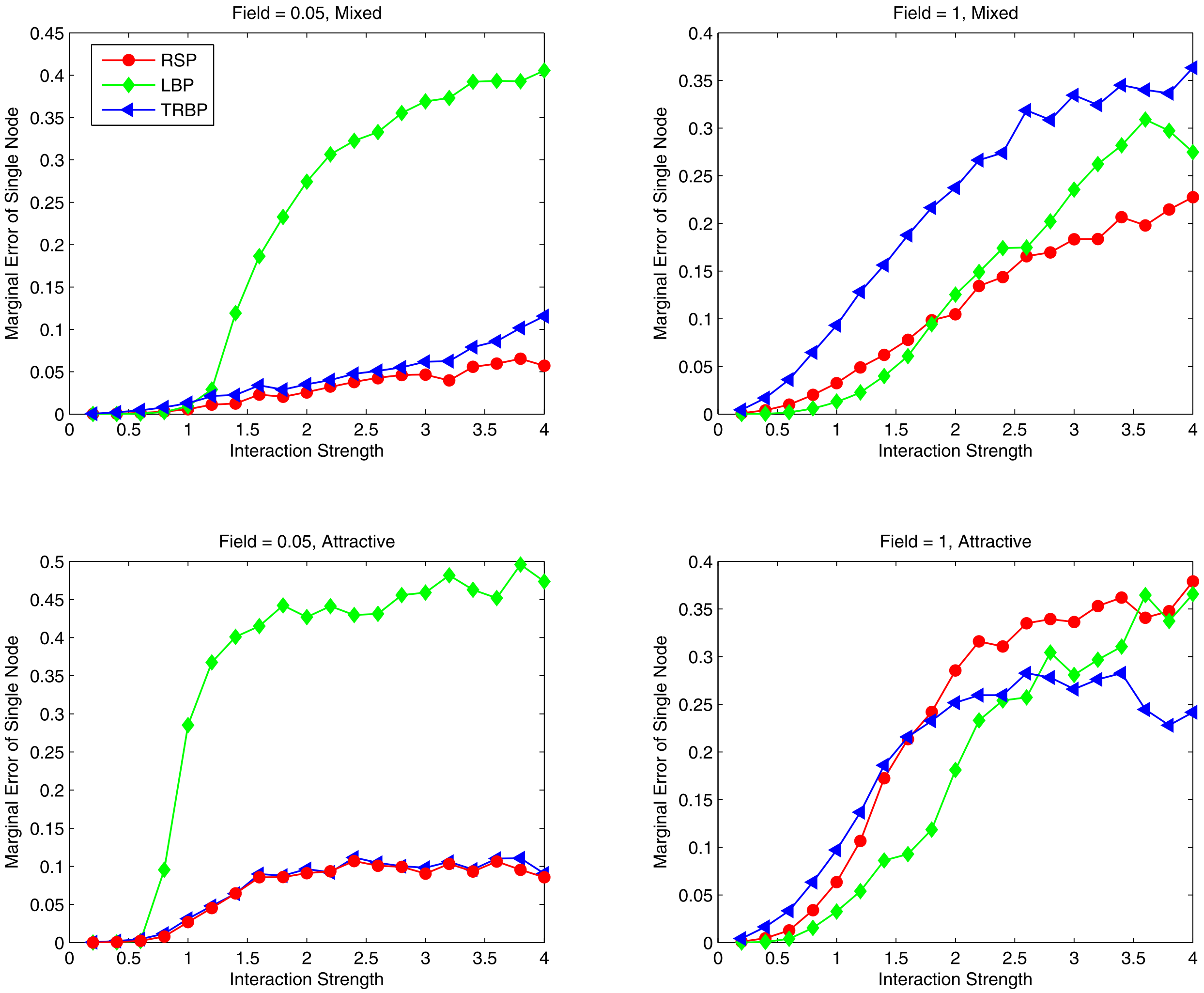

Fig. 12. Comparison of errors in the approximate marginals of single variable of RSP and other approximate inference algorithms.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Project No. 61071131), Beijing Natural Science Foundation (No. 4122040), National Key Basic Research and Development Program of China (No. 2009CB320602) and United Technologies Research Center (UTRC).

References

[1] M.F. Tappen, W.T. Freeman, Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters, in: ICCV, 2003, pp. 900–907.

[2] A. Rush, D. Sontag, M. Collins, T. Jaakkola, On dual decomposition and linear programming relaxations for natural language processing, in: EMNLP, Association for Computational Linguistics, 2010, pp. 1–11.

[3] C. Yanover, O. Schueler-Furman, Y. Weiss, Minimizing and learning energy functions for side-chain prediction, Journal of Computational Biology 15 (7) (2008) 899–911.

[4] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT Press, 2009.

[5] M. Wainwright, M. Jordan, Graphical models, exponential families, and variational inference, Foundations and Trends in Machine Learning 1 (1–2) (2008) 1–305.

[6] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.

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

[8] S. Aji, R. McEliece, The generalized distributive law and free energy minimization, in: Conference on Communication Control and Computing, vol. 39, 2001, pp. 672–681.

[9] J. Bondy, U. Murty, Graph Theory, Springer, Berlin, 2008.

[10] D. Batra, A. Gallagher, D. Parikh, T. Chen, Beyond trees: MRF inference via outer-planar decomposition, in: CVPR, 2010, pp. 2496–2503.

[11] J. Blair, B. Peyton, An introduction to chordal graphs and clique trees, in: The IMA Volumes in Mathematics and Its Applications, vol. 56, 1993, pp. 1–30.

[12] J. Yarkony, C. Fowlkes, A. Ihler, Covering trees and lower-bounds on quadratic assignment, in: CVPR, 2010.

[13] M. Wainwright, T. Jaakkola, A. Willsky, A new class of upper bounds on the log partition function, IEEE Transactions on Information Theory 51 (7) (2005) 2313–2335.

[14] O. Meshi, A. Jaimovich, A. Globerson, N. Friedman, Convexifying the Bethe free energy, in: UAI, 2009.

[15] D. Bertsekas, W. Hager, O. Mangasarian, Nonlinear Programming, Athena Scientific, Belmont, MA, 1999.

[16] A. Globerson, T. Jaakkola, Approximate inference using conditional entropy decompositions, in: AISTATS, 2008.