Journal of Computational Science &Engineering 37 (2018) 990-993

Available at http://www.asocse.org ISSN 1710-4068

A local transition for a class of stable graph

Rui Wang*, Cheng Xu, Qianwen Zang

Department of Mathematics and Statistics, Qingdao University, Qingdao 266071, China

Abstract: **In this paper, two classes of stable graphs with three types of edges are studied. These include the class of MC graphs and the class of ribbonless graphs. Their stability ensures the equivalence between graphs before and after the marginalizing and conditioning process. We propose a transition method of them so that simpler graph structures could be obtained.**

Keywords: MC graph; Bayesian networks; Ribbonless graph

Received 2 September 2018, Revised 22 September 2018, Accepted 25 September 2018

*Corresponding Author: Rui Wang, wangrui9469@163.com

1. Introduction

Graphical model combines the family of probability distribution and its corresponding graph structure. Based on graph expression, the probabilistic graphical model could encode complex distribution of high dimension[1]. In a graph structure, vertices represent random variables and edges represent the direct probability between variables. Rather than disposing of the possible probability of all variables we meet in practice, we decompose the distribution into smaller factors which have certain definitions in various smaller probability spaces. The joint distribution is represented by product of those factors.

Pearl[2] have studied the causality relationship and the response to intervention between random variables in Bayesian network. The class of DAG (Directed Acyclic Graph) represents the conditional independence of variables in Bayesian Network. Aragam[3] regard DAG as recursive linear structural equation model, which present an effective solution for estimating regression value. For incomplete data set, DAG model acquires a good effect in simplifying conditional probability.

Marginalizing over unobserved variables (hidden variables) and conditioning over selected variables are two main processes in simplifying graph models. Wermuth[4] have testified that DAG is unstable in both processes. Its instability shows as a result of lacking two kinds of edges. For example, as shown in Figure 1, the vertice h is in the marginalizing set and after the process of marginalization, we obtain a new graph with a bidirected edge which is not a part of DAG. The DAG in Figure 2 produces a line edge after conditioning over h and there is no way to represent the same relationship between i and j using a DAG.

In order to eliminate the limitation, we introduce the class of MC graphs. Sadeghi[5] have shown that MC graph is stable after marginalizing and conditioning. MC graphs have three types of edges and describe all possible relationships between random variables, including neighbor (—), spouse (↔) and parent (→). In some cases, we do not have to acquire all relationships, but a part of them. Thus some vertices and edges adjacent to them could be removed. If the obtained graph expresses same independence as before, they are Markov equivalent[6,7].

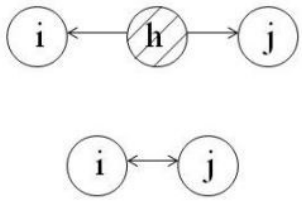

Figure 1. Marginalization over h in a DAG.

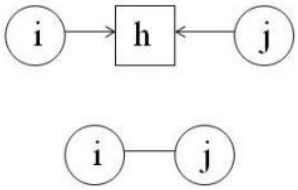

Figure 2. Conditioning over h in a DAG.

In the branches of MC graph, RG (Ribbonless Graph), SG (Summary Graph) and AG (Ancestral Graph) are three kinds of stable graph and own simpler structures. Li[8] have described their inclusion relations and transition algorithms. Among the three classes of graphs, AG has a simplest structure. Accordingly, there is more lost information in the generating process. Thus, we focus on the class of RG, which owns a simpler structure than MC graph and removes less vertices than AG.

In this paper, we propose a transition method from an MC graph to a Markov equivalent RG. The rest of the paper is organized as follows. In Section 2, we introduce the marginalizing and conditioning processes which were originally proposed by Koster[9]. In Section 3, we introduce the class of RGs and show how an MC graph generate a RG. In Section 4, a few concluding remarks are mentioned.

2. Preliminary

Jan T.A. Koster[9] proposed the method of marginalizing and conditioning over MC graphs based on d-separation in Bayesian network[2]. For an MC graph G = (V, E) , where V is a collection of vertices

2018@ The American Computational Science Society. All rights reserved.

and E is a collection of edges, a marginalizing process and a conditioning process can be applied to its vertices. Before introducing the processes, it is necessary to know some basic concepts.

There are there types of edges for an MC graph, including arrows (→), lines (—) and arcs (↔). A loop is an edge like i → i, i — i or i ↔ i. We denote the line loop(i — i) io for convenience.

A walk is a list ⟨ h0, e1, h1,⋯, ek, hk ⟩ of vertices and edges such that for $1 \leq i \leq k$, the edge ei has endpoints hi-1 and hi. A path is a walk with no repeated vertice or edge. A V - structure or simply VS is a path with three vertices and two edges. In MC graphs, we call VS (i, h, j) like i → h ←j, i ↔ h ← j and i ↔ h ↔ j collider VS and otherwise non-collider VS.

Two paths (including VS) are called endpoint-identical if the presence or lack of arrowheads pointing to endpoints on the path are same in both. For example, i → j, i → h ↔ j and i → h1 ↔ h2 → j are all endpoint-identical.

Now back to the marginalizing and conditioning processes. They were shown originally with a few tables[9] and now we simplify them for sententious editions.

The marginalizing process is as follows.

Step 1. Decide vertices to implement the algorithm and put them into the set M.

Step 2. For vertice h (∈ M) at which there exists a line-loop, generate an endpoint-identical edge over VS (i, h, j). Go next step when there is no more edge can be added.

Step 3. For vertice h (∈ M), generate an endpoint-identical edge over non-collider VS (i, h, j). Go next step when there is no more edge can be added.

Step 4. Remove all vertices in M and their adjacent edges.

Step 5. Remove all loops except line-loops.

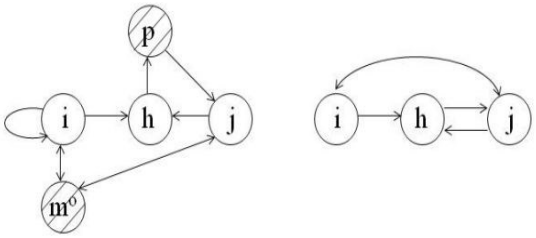

Figure 3. An MC graph and the obtained graph after marginalizing.

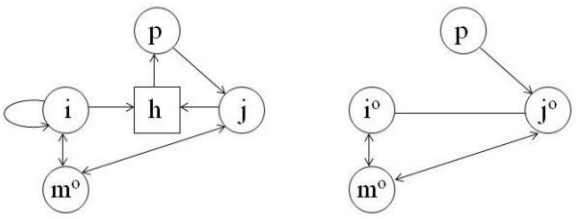

Figure 4. An MC graph and the obtained graph after conditioning.

For example, the graph in Figure 3 shows the result of this process supposing the marginalizing set M = { p, m }.

The conditioning process is as follows.

Step 1. Decide vertices to implement the algorithm and put them into the set C.

Step 2. For vertice h (∈ C), generate an endpoint-identical edge over collider VS (i, h, j). Go next step when there is no more edge can be added.

Step 3. Remove all vertices in C and their adjacent edges.

Step 4. Remove all loops except line-loops.

For example, the graph in Figure 3 shows the result of this process supposing the conditioning set C = { h }.

From the examples in Figure 3 and Figure 4, we can see that the class of MC graphs is stable.

3. Transition Method

In this section, we will introduce a method to generate a RG from a given MC graph. We first present some basic concepts and give an introduction to ribbon and RG.

We call a graph directed graph if there is no arc or line in the graph. The class of DAG is a special case of directed graph. A path (or a cycle) in a directed graph is direction-preserving if all its arrows point to one direction. For example, ρ = ⟨ h1 → h2 → ⋯ → hn ⟩ is a direction-preserving path. We call h1 the ancestor of hn and hn the descendant of h1.

A ribbon is a collider VS (i, h, j) such that both the following conditions hold:

There is no endpoint-identical edge between i and j.

h or a descendant of h is an endpoint of a line or on a direction-preserving cycle.

If h or a descendant of h is an endpoint of a line, then we call the ribbon straight-ribbon and if they are on a direction-preserving cycle we call the ribbon cyclic-ribbon. If h or a descendant of h is an endpoint of a line loop, then we call the ribbon loop-ribbon and otherwise non-loop-ribbon.

A RG is an MC graph with no loop or ribbon.

Now we present the transition method.

Step 1. Marginalizing and conditioning over an MC graph.

Step 2. Generate an endpoint-identical edge repeatedly between i and j over loop-ribbon VS (i, h, j).

Step 3. Remove all loops.

Consider A is the collection of added edges by this method. A non-loop-ribbon VS (i, h, j) is durable if the corresponding endpoint-identical edge is not in A. We denote the graph after step 1 GMC and the graph we finally get R.

Next we will see after the above steps, the graph we obtain is indeed a RG.

Obviously after step 2 there is no loop-ribbon in the graph and after step 3 the graph is loopless. Thus we show there is no durable non-loop ribbon in R. As we

2018@ The American Computational Science Society. All rights reserved.

know there is no non-loop-ribbon in GMC from previous section. Thus it is enough to show that there is no non-loop-ribbon in the generated graph.

Consider a durable non-loop-ribbon is created after an iteration and we denote it by VS (g, q, q1). Then we need to consider the following cases.

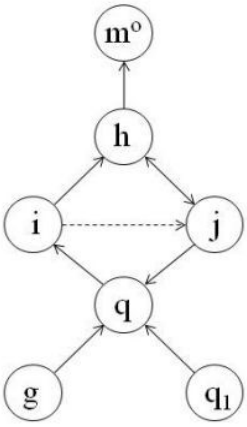

Figure 5. An i → j edge is added from a loop-ribbon and is on a direction-preserving cycle.

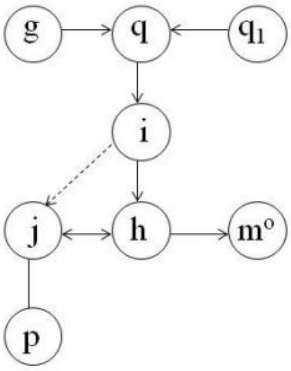

Figure 6. An i → j edge is added from a loop-ribbon and is on a direction-preserving path.

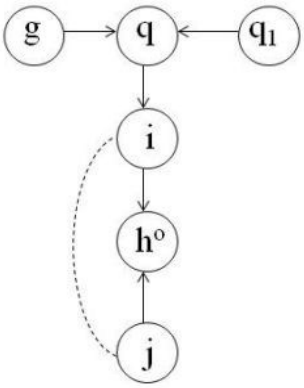

Figure 7. An i — j edge is added from a loop-ribbon VS (i, h, j).

(a).Suppose an i → j edge is added from a loop-ribbon VS (i, h, j). The edge is not on the ribbon VS (g, q, q1) but on a direction-preserving cycle containing q. Based on the definition of loop-ribbon, h or a descendant of h is an endpoint of a line-loop and there exists the i → k edge. This implies that a descendant of i is an endpoint of a line-loop. Thus a descendant of q is an endpoint of a line-loop, which indicates that VS (g, q, q1) is a loop-ribbon. This is against our assumption. (Figure 5)

(b).Suppose an i → j edge is added from a loop-ribbon VS (i, h, j). The edge is not on the ribbon VS (g, q, q1) but on a direction-preserving path from q to a node that is an endpoint of a line. Same as (a), a descendant of q is an endpoint of a line-loop, which states that VS (g, q, q1) is a loop-ribbon. Thus the case is not possible. (Figure 6)

(c).Suppose an i — j edge is added from a loop-ribbon VS (i, h, j). The edge is not on the ribbon VS (g, q, q1) and there is a direction-preserving path from q to i. Same as the cases present before, the VS (g, q, q1) is a not a non-loop-ribbon and the case is not possible. (Figure 7)

(d).Suppose the g •→ q ("•"indicate that whether there is an arrowhead at g or not is uncertain) edge is generated by a loop-ribbon VS (g, t, q). Based on the structure of ribbons, there exists a non-loop-ribbon VS (t, q, q1). According to our assumption, the ribbon is not durable, which indicates that its endpoint-identical edge must be in A. Once the t ←• q1 edge is added, the VS (g, t, q1) becomes a loop-ribbon and its endpoint-identical edge is added. As the edge is also an endpoint-identical edge of the non-loop-ribbon VS (g, q, q1), the ribbon is not durable, which is against our assumption. (Figure 8).

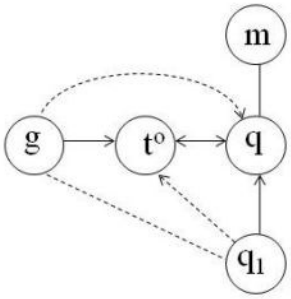

Figure 8. An g → q edge is added from a loop-ribbon VS (g, t, q).

4. Conclusion

In this paper, we propose a transition method for generating a class of stable graph: Ribbonless graph. For a given MC graph, we first apply marginalizing and conditioning process to it, then we find all loop-ribbons and add endpoint-identical edges for each of them. After deleting all loops in the graph, we get the needed RG, which owns a simpler structure than MC graph.

This method is a local algorithm, which means that we need to check all VS in the graph. In the future, global algorithm for this transition is necessary for us

2018@ The American Computational Science Society. All rights reserved.

to develop.

References

[1] Y. F. Wang, Q. S. Han, Tsinghua University Press, 2015.
[2] J. Pearl, Cambridge University Press, 2000.
[3] B. Aragam, A. A. Amini, Q. Zhou, ArXiv e-prints, 1511 (2015) 8963.
[4] N. Wermuth, D. R. Cox, J. Pearl, Technical Report, 94 (1994).
[5] K. Sadeghi, Bernoulli, 19 (2011) 2330-2358.
A. S. Andersson, D. Madigan, D. M. Perlman, Journal of Statistics, 24 (1997) 81-102.
[6] Hauser, P. Bühlmann, M. Chickering, Journal of Machine Learning Research, 13 (2012) 2409-2464.
[7] Y. Y. Li, L. Y. Yang, M. W. Yang, Journal of Jilin University (Science Edition), 53 (2015) 897-903.
[8] J. T. A. Koster, Bernoulli, 8 (2002) 817-840.

2018@ The American Computational Science Society. All rights reserved.