Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2012, Article ID 579543, 9 pages
doi:10.1155/2012/579543

Research Article

Structural Learning about Directed Acyclic Graphs from Multiple Databases

Qiang Zhao

School of Mathematical Sciences, Shandong Normal University, Jinan 250014, China

Correspondence should be addressed to Qiang Zhao, zhaoqmath@gmail.com

Received 5 October 2012; Accepted 19 November 2012

Academic Editor: Xiaodi Li

We propose an approach for structural learning of directed acyclic graphs from multiple databases. We first learn a local structure from each database separately, and then we combine these local structures together to construct a global graph over all variables. In our approach, we do not require conditional independence, which is a basic assumption in most methods.

1. Introduction

Graphical models including independence graphs, directed acyclic graphs (DAGs), and Bayesian networks have been applied widely to many fields, such as data mining, pattern recognition, artificial intelligence, complex systems, and causal discovery [1–4]. Graphical models can be used to cope with uncertainty for a large system with a great number of variables. Structural learning of graphical models from data is an important and difficult problem and has been discussed by many authors [1–5]. There are two main kinds of structural learning methods. One is constraint-based learning and the other is score-based learning. Most of the structural learning approaches deal with only one database with completely observed data. With the development and popularity of computers, various databases have been built, which may contain different sets of variables and overlap with each other. For example, in medical research, a researcher collects data of these variables, another researcher may collect data of other variables, and they have some common variables.

In this paper, we discuss how to learn the structures of DAGs from multiple databases with different and overlapped variables. In our approach, we first learn a local structure from each database separately, and then we combine these structures together to construct a global graph over all variables. Several theoretical results are shown for the validity of our algorithm. Our approach can validly discover DAGs from multiple databases. In our

approach, we only need a weaker condition than conditional independence, which is a basic assumption in most methods [1–5]. This approach can also utilize the prior knowledge of conditional independencies to reduce the number of variables in each conditional set.

Section 2 gives notation and definitions. In Section 3, we show how to construct the DAG with multiple databases. We give an example in Section 4 to illustrate our approach for recovering a DAG. Finally a discussion is given in Section 5.

2. Notation and Definitions

Let $\vec{G}_V = (V, \vec{E}_V)$ denote a DAG where V is a set of n vertices $\{a, b, \ldots\}$ and $\vec{E}_V \subseteq V \times V$ is a set of directed edges. A directed edge from a vertex a to another vertex b is denoted by $\langle a, b\rangle$, and we say that a is a parent of b and b is a child of a. We denote the set of all parents of a vertex b by $pa(b)$. A path l between the two distinct vertices a and b is a sequence of distinct vertices that starts with a, ends with b, and two consecutive vertices are connected by an edge; that is, $l = (c_0 = a, c_1, \ldots, c_{n-1}, c_n = b)$, where $\langle c_{i-1}, c_i\rangle$ or $\langle c_i, c_{i-1}\rangle$ is contained in $\vec{E}_V$, for $i = 1, \ldots, n$ $(n \geq 1)$, and $c_i \neq c_j$, for all $i \neq j$. We say that a is an ancestor of b and b is a descendant of a if there is a path from a to b in $\vec{G}_V$ and all edges on this path point at the direction toward b. Denote the set of ancestors of b as $an(b)$. A path l is said to be separated by a set of vertices Z if and only if there exists at least one $i \in \{1, 2, \ldots, k-1\}$, such that $c_i \in Z$. And l is said to be d-separated by Z if and only if

(1) l contains a "chain" $i \to m \to j$ or a "fork" $i \leftarrow m \to j$ such that the middle vertex m is in Z, or

(2) l contains an "inverted fork" $i \to m \leftarrow j$ such that the collision vertex m is not in Z and no descendant of m is in Z.

Two sets A and B of vertices are separated (d-separated) by a set C if C separates (d-separates) every path from any vertex in A to any vertex in B, and we say C is a separator (d-separator) of A and B. If $a \to c \leftarrow b$, $\langle a, b\rangle \notin \vec{E}_V$ and $\langle b, a\rangle \notin \vec{E}_V$ in a DAG $\vec{G}_V$, then the triple (a, c, b) is called an immorality and a and b are called two parents of the immorality.

Note that the two vertexes a and b of the DAG are nonadjacent if and only if they are d-separated by some subset $S \subset V\{a, b\}$. Though this is obvious the case by taking S to be either $pa(a)$ or $pa(b)$ for DAGs, there are certain types of graphs in which nonadjacency is not sufficient for separability, for example, the ancestral graph in [6].

Example 2.1. Consider a DAG $\vec{G}_V = (V, \vec{E}_V)$ in Figure 1, where $V = \{a, \ldots, j\}$ and $\vec{E}_V = \{\langle a, b\rangle, \langle a, c\rangle, \langle a, d\rangle, \langle b, e\rangle, \langle c, e\rangle, \langle c, f\rangle, \langle d, f\rangle, \langle f, h\rangle, \langle g, e\rangle, \langle g, h\rangle, \langle g, i\rangle, \langle h, j\rangle\}$. In this DAG, we have $pa(e) = \{b, c, g\}$ and that (c, e, b), (b, e, g), (c, e, g), (c, f, d), and (f, h, g) are immoralities. The path $l = (a, c, f, h)$ between a and h is d-separated by $\{c\}$ or $\{f\}$, while the path $l' = (a, b, e, g)$ between a and g is d-separated by the empty set. Vertex a is an ancestor of h and h is a descendant of a. The sets $A = \{a\}$ and $B = \{e, f, g, h, i, j\}$ are d-separated by the set $C = \{b, c, d\}$.

Let n vertices $\{a, b, \ldots\}$ in a DAG $\vec{G}_V$ denote n variables $\{X_1, \ldots, X_n\}$. If a joint distribution or density of variables $X_1, \ldots, X_n$ satisfies

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid pa_i), \tag{2.1}$$

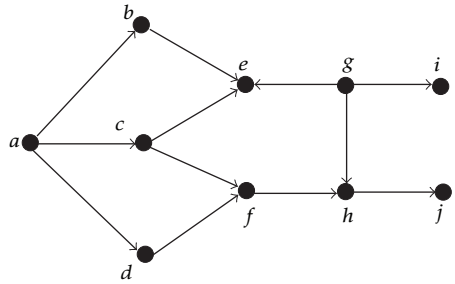


Figure 1: A directed graph $\vec{G}_V$.

where $P(x_i \mid pa_i)$ is the conditional probability or density of X_i given $pa(X_i) = pa_i$, then DAG $\vec{G}_V$ and the distribution P are said to be compatible [3] and P obeys the global directed Markov property of $\vec{G}_V$ [2]. We use the notation in [7] to denote independence. Let $X \perp\!\!\!\perp Y$ denote the independence of X and Y and $X \perp\!\!\!\perp Y|Z$ the conditional independence of X and Y given Z for any variables or sets of variables X, Y, and Z. Since in our discussion we always think the joint distribution corresponds to an underlying DAG, we do not differentiate the usage of letters as X, Y, Z, A, B, C to denote variables or vertexes; however, we will make an obvious reference if the context is not clear.

As pointed out by [3], if sets X and Y are d-separated by Z, then X is independent of Y conditionally on Z in every distribution that is compatible with $\vec{G}_V$. In this paper, we assume that all the distributions are compatible with $\vec{G}_V$. We also assume that all independencies of a probability distribution of variables in V can be checked by d-separation of $\vec{G}_V$, called the faithfulness assumption in [4]. The faithfulness assumption means that all independencies and conditional independencies among variables can be represented by $\vec{G}_V$. For a distribution which obeys the faithfulness assumption, we can learn the underlying DAG by checking the pairwise conditional independence $X \perp\!\!\!\perp Y \mid Z$, where X and Y are two random variables and Z is a subset of variables.

A hypergraph is a collection of vertex sets [8, 9]. Multiple databases $\mathcal{C} = \{C_1, \ldots, C_H\}$ are depicted as a hypergraph, where a hyperedge C_h is an observed variable set in a database and $\cup_{h=1}^{H} C_h = V$ [5, 10]. A database with an observed variable set C_h is treated as a sample from a marginal distribution of the variable set C_h. Let $D_h = C_h \cap (\cup_{k \neq h} C_k)$, which is the intersection of C_h and the other sets. Given a collection of databases $\mathcal{C}$, there is no information on higher interactions over different databases.

Example 2.2. Let $\mathcal{C} = \{C_1 = \{1,2,3,4\}, C_2 = \{1,3,5,6\}, C_3 = \{4,6,7\}\}$ be a hypergraph, as shown in Figure 2. We can get that $D_1 = \{1,3,4\}$, $D_2 = \{1,3,6\}$, and $D_3 = \{4,6\}$.

3. Structural Learning of DAGs

In this section, we propose an approach for structural learning of DAGs. In our approach, we first learn a local structure from each database, and then we combine these local structures together to construct a global graph over all variables.

Note that for a distribution obeying the faithfulness assumption with respect to a certain DAG $\vec{G}_V$, there may not exist a DAG which can fully represent all the conditional

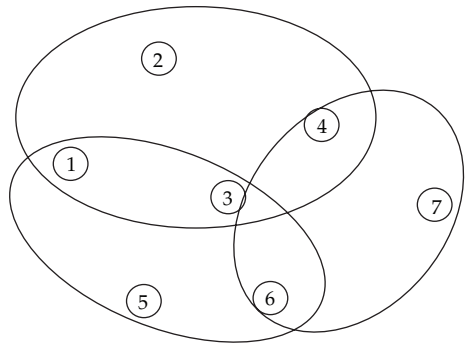


Figure 2: A hypergraph.

independencies in the marginal distribution; see [6] for more discussion on this issue. Though this fact implies that we may not expect to learn a DAG for each database, it will be shown that under a certain condition a marginal structure could be learned which partially reflects the original true structure.

We consider a joint distribution on a set of variables V which satisfies the faithfulness assumption and denote by $\vec{G}_V = (V, \vec{E}_V)$ the DAG which can fully represent the conditional independencies in this joint distribution. We consider the problem of structure recovery from multiple databases. To facilitate our discussion, we first give the definition of the local structure.

Definition 3.1. For a DAG $\vec{G}_V = (V, \vec{E}_V)$ and a subset $V' \subset V$, we define the local structure $\overline{G}_{V'}$ to be an undirected graph $\overline{G}_{V'} = (V', E')$ where the edge set $E' = \{(u, v) : u, v \in E' \ \ and \ \ u \not\perp\!\!\!\perp v \mid S \text{ for all } S \subset V' \setminus \{u, v\}\}$.

From the definition, it is known that to judge whether u and v are adjacent in the local structure $\overline{G}_{V'}$, we need only to search for a d-separator S from all possible variable subsets $S \subset V'$ such that two variables u and v are independent conditionally on S. With the faithfulness assumption, this is equivalent to test whether u and v are independent conditionally on S and this can be done by using data observed on V' only. Note that the edges in the local structure may be spurious in the sense that its two vertexes are not adjacent in the original DAG; we call these edges spurious edges. However, in the section below, we show that such learned local structure could be used to identify part of the true structure of the original DAG under one additional condition. We give a lemma to be used in proofs of theorems.

Lemma 3.2. *If u is not an ancestor of v, then u and v are d-separated by a subset of V if and only if they are d-separated by $pa(u)$.*

Proof. See [2]. □

The following two theorems show the relationship between the local structure and the true structure of the original DAG.

Theorem 3.3. *Let A, B, and C be a partition of V. If A and B are separated by C, then two vertices in A are d-separated by a subset of V if and only if they are d-separated by a subset of $A \cup C$.*

Proof. The necessity is obvious since $V \supseteq A \cup C$. For the sufficiency, let u and v be two vertices in A that are d-separated by S $(\subseteq V)$. Thus there is no edge connecting u and v in $\vec{G}_V$. Since u and v are contained in A and A and C are separated by B, $pa(u)$ and $pa(v)$ are contained in $A \cup C$. Without loss of generality, suppose that u is not an ancestor of v. From Lemma 3.2 we have that $pa(u)$ $(\subseteq A \cup C)$ d-separates u and v. □

From Theorem 3.3, we can see that two vertexes in A are adjacent in the DAG $\vec{G}_V$ if and only if they are adjacent in the local structure $\overline{G}_{A\cup C}$. This means that with the faithfulness assumption, the existence of edges falling into A can be determined validly from the marginal distribution of $A \cup C$.

Theorem 3.4. *Suppose A, B, and C is a partition of V and A and B are separated by C. Let $u \in C$ and $v \in A \cup C$. If u and v are d-separated by a subset of $A \cup C$, then they are d-separated by a subset of V.*

Proof. The result is obvious since $A \cup C \subseteq V$. □

According to Theorem 3.4, we can see that u $(\in C)$ and v $(\in A \cup C)$ are not adjacent in the DAG $\vec{G}_V$ if they are not adjacent in the local structure $\overline{G}_{A\cup C}$. This means that we may get spurious edges in the local structure $\overline{G}_{A\cup C}$.

According to the two Theorems above, we can get that an edge whose two vertices are contained only by one database C_h can be determined by using the marginal distribution of C_h without requirement of the other databases. Those edges crossing $C_h \setminus D_h$ and D_h or falling into D_h may be spurious. Their existence must be determined according to multiple databases.

Now we give the algorithm for structural learning of directed acyclic graphical models.

Algorithm 3.5. Construct a DAG from Multiple Databases

(1) Input: Multiple databases $\mathcal{C} = \{C_1, \ldots, C_H\}$.

(2) Construct a local structure $\overline{G}_h$ from database C_h separately for each h:

  (a) Initialize $\overline{G}_h$ as a complete undirected graph;

  (b) Delete edge (x, y) from $\overline{G}_h$ if there exists a subset Z of C_h such that x and y are conditional independent given Z.

(3) Construct the global undirected graph $\overline{G}_V$:

  (a) Initialize the edge set E of $\overline{G}_V$ as the union of all edge sets of $\overline{G}_h$ for $h = 1, \ldots, H$;

  (b) For an edge (x, y), which falls into some D_h, delete it from $\overline{G}_V$ if it is absent in some other $\overline{G}_{h'}$.

(4) Delete spurious edges to construct the global skeleton:

  (a) For an edge (x, y) with x in some $C_h \setminus D_h$ and y in D_h, delete it from $\overline{G}_V$ if there exists a subset Z of $ne(y)$ such that x and y are conditional independent given Z;

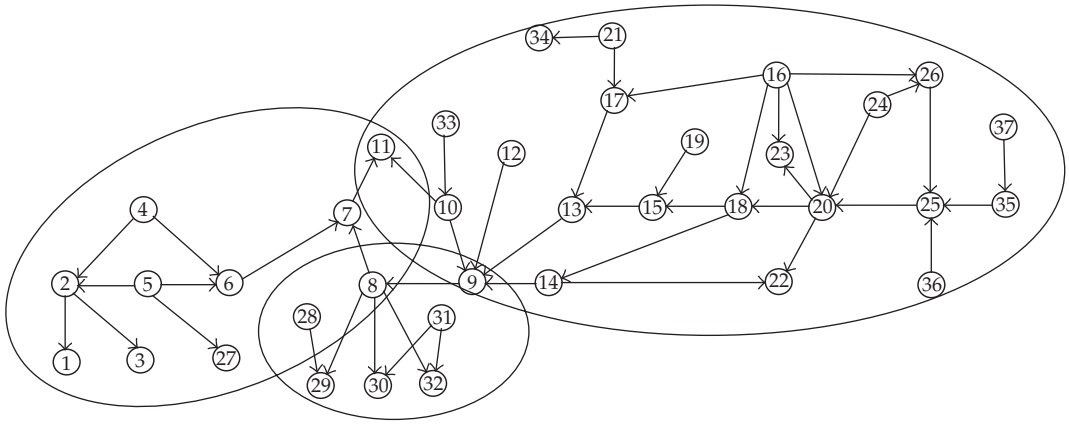


Figure 3: The ALARM network and multiple databases.

(b) For an edge (x, y), which fall in some D_h, delete it from $\overline{G}_V$ if there exists a subset Z of $ne(x) \setminus \{y\}$ or $ne(y) \setminus \{x\}$ such that x and y are conditional independent given Z.

(5) Construct the equivalence class:

(a) Orient the local skeleton $x - z - y$ as $x \rightarrow z \leftarrow y$ if x and y are not adjacent in $\overline{G}_V$ and $z \notin Z$;

(b) Orient other edges if each opposite of them creates either a directed cycle or a new immorality.

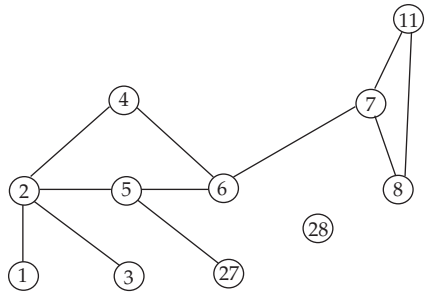
(6) Output: The equivalence class of DAGs.

Note that $ne(y)$ at step 4 denotes all the vertices that are adjacent with vertex y. According to Theorems 3.3 and 3.4, the equivalence class constructed by the above algorithm is valid.

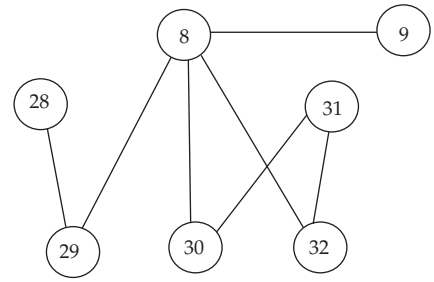
4. Illustration of Structural Learning

In this section, we illustrate our algorithm using the ALARM network in Figure 3 that is often used to evaluate structural learning algorithms [4, 11, 12]. The ALARM network in Figure 3 describes causal relations among 37 variables in a medical diagnostic system for patient monitoring. Using the network, some researchers generate continuous data from normal distributions and others generate discrete data from multinomial distributions [4, 12]. Our approach is applicable for both continuous and discrete data. Since the validity of our algorithm can be ensured by Theorems 3.3 and 3.4, the algorithm is illustrated by using conditional independencies from the underlying directed acyclic graph in Figure 3 rather than conditional independence tests from simulated data.

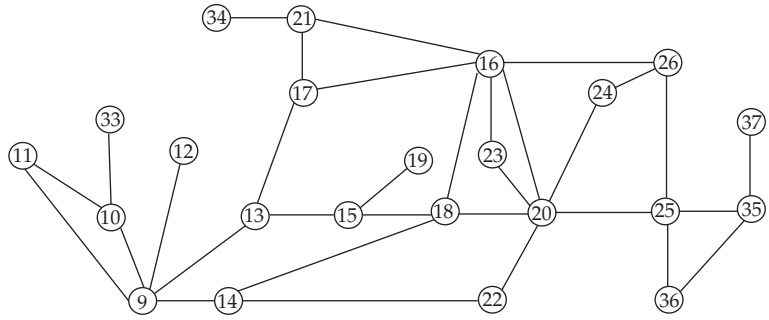
Suppose that we have three databases as depicted by the hypergraph in Figure 3. Database C_1 contains variables $\{1, 2, 3, 4, 5, 6, 7, 8, 11, 27, 28\}$, database C_2 contains variables $\{8, 9, 28, 29, 30, 31, 32\}$, and database C_3 contains variables $\{9, 10, 11, 12, 13, 14, 15, 16, 18, 19, 20, 22, 23, 24, 25, 26, 33, 34, 35, 36, 37\}$. Thus $D_1 = \{8, 11, 28\}$, $D_2 = \{8, 9, 28\}$, and $D_3 = \{9, 11\}$.



(a) The local structure $\overline{G}_1$ for database C_1



(b) The local structure $\overline{G}_2$ for database C_2



(c) The local structure $\overline{G}_3$ for database C_3

Figure 4: Local structures for all databases.

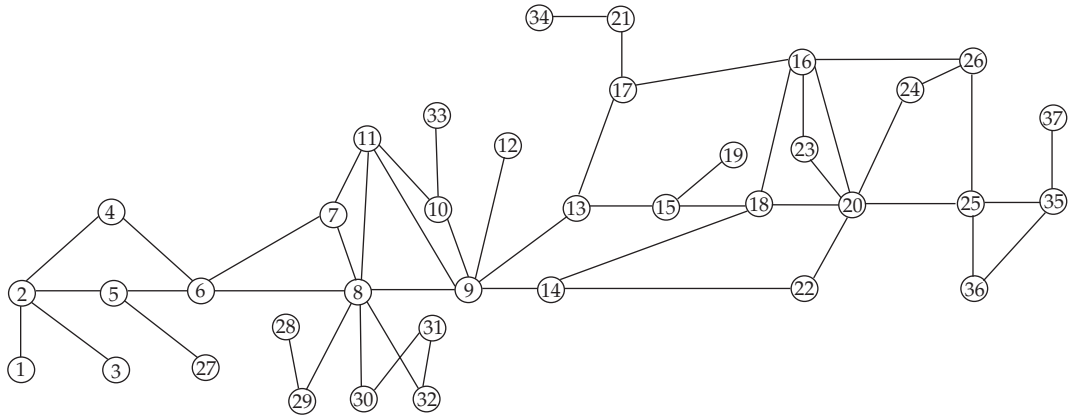


Figure 5: The global undirected graph.

Note that $C_1 \setminus D_1$ and $V \setminus C_1$ are not conditional independent given D_1. At Step 2, the local structures are obtained separately from the three databases, as shown in Figures 4(a), 4(b), and 4(c), respectively, for example, the undirected edge $(6, 27)$ because 6 is independent of 27 conditional on $\{5\}$.

At step 3, we combine three local structures together to construct the global undirected graph, as shown in Figure 5. At step 4, we delete the spurious edges to get the global skeleton, which is the undirected version of Figure 6. For example, the spurious edge $(8, 11)$ can be deleted since variables 8 and 11 are conditional independent given $\{7, 10\}$.

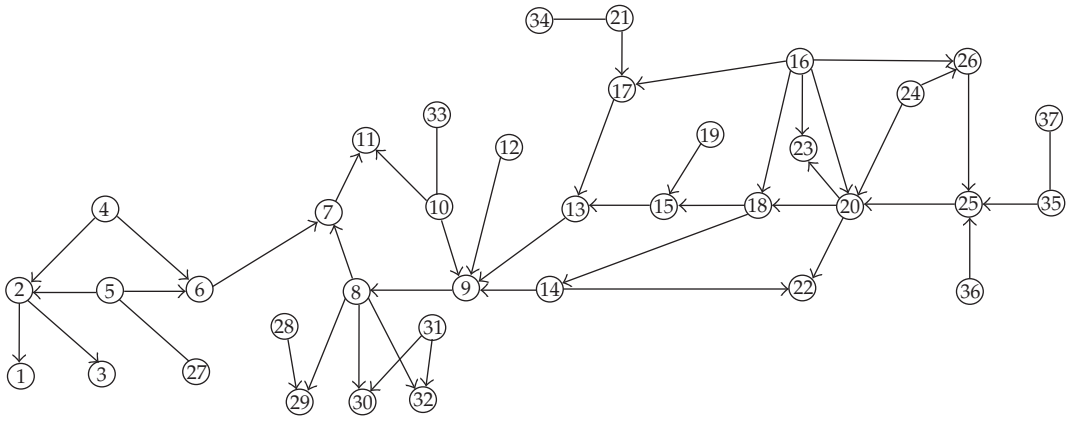


Figure 6: The partially directed acyclic graph.

At step 5, we determine immoralities and orient edges as much as possible. For example, the direction of the undirected edge $(1,2)$ is determined as $\langle 2,1\rangle$ by $\langle 4,2\rangle$ so as not to create a new v-structure, and the direction of the undirected edge $(16,23)$ is determined as $\langle 16,23\rangle$ by $\langle 16,20\rangle$ and $\langle 20,23\rangle$ so as not to create a cycle. At last we obtain the equivalence class in Figure 6, in which all directed edges are oriented correctly, except that four undirected edges $(5,27)$, $(10,33)$, $(21,34)$, and $(35,37)$ cannot be oriented because any of their orientation leads to a Markov equivalent DAG.

5. Discussion

In this paper, we presented an approach for structural learning of directed acyclic graphs from multiple databases. In our approach, we require that $C_h \setminus D_h$ and $V \setminus C_h$ are separated by D_h, which is a weaker condition than the condition that $C_h \setminus D_h$ and $V \setminus C_h$ are d-separated by D_h. This condition can be judged with experts' prior knowledge of associations among variables, such as Markov chains, chain graphical models, and time series.

There are several obvious advantages of our approach for structural learning. First we do not require conditional independence, which is a basic assumption in most methods. Second we search d-separators in C_h or $ne(x)$, which is much smaller than V. At last, the theoretical results proposed in this paper can be applied to scheme design of multiple databases. Without loss of information on structural learning of DAGs, a joint data set can be replaced by a group of incomplete data sets based on the prior knowledge.

Acknowledgments

I would like to thank the referees for their valuable comments and suggestions. This research was supported by NSFC (11001155), Doctoral Fund of Shandong Province (BS2010SW030), and A Project of Shandong Province Higher Educational Science and Technology Program (J11LA08).

References

[1] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter, *Probabilistic Networks and Expert Systems*, Springer-Verlag, New York, NY, USA, 1999.
[2] S. L. Lauritzen, *Graphical Models*, Oxford University Press, Oxford, UK, 1996.
[3] J. Pearl, *Causality: Models, Reasoning, and Inference*, Cambridge University Press, Cambridge, UK, 2000.
[4] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction and Search*, MIT Press, Cambridge, Mass, USA, 2nd edition, 2000.
[5] X. Xie, Z. Geng, and Q. Zhao, "Decomposition of structural learning about directed acyclic graphs," *Artificial Intelligence*, vol. 170, no. 4-5, pp. 422–439, 2006.
[6] T. Richardson and P. Spirtes, "Ancestral graph Markov models," *The Annals of Statistics*, vol. 30, no. 4, pp. 962–1030, 2002.
[7] A. P. Dawid, "Conditional independence in statistical theory," *Journal of the Royal Statistical Society B*, vol. 41, no. 1, pp. 1–31, 1979.
[8] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis, "On the desirability of acyclic database schemes," *Journal of the Association for Computing Machinery*, vol. 30, no. 3, pp. 479–513, 1983.
[9] C. Berge, *Graphs and Hypergraphs*, North-Holland Publishing, Amsterdam, The Netherlands, 2nd edition, 1976.
[10] Z. Geng, K. Wan, and F. Tao, "Mixed graphical models with missing data and the partial imputation EM algorithm," *Scandinavian Journal of Statistics*, vol. 27, no. 3, pp. 433–444, 2000.
[11] I. Beinlich, H. Suermondt, R. Chavez, and G. Cooper, "The ALARM moni- toring system: a case study with two probabilistic inference techniques for belief networks," in *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*, pp. 247–256, Springer-Verlag, Berlin, Germany, 1989.
[12] D. Heckerman, "A tutorial on learning with Bayesian networks," in *Learning in Graphical Models*, M. I. Jordan, Ed., pp. 301–354, Kluwer Academic, Dodrecht, The Netherlands, 1998.