Structural Learning of Multivariate Regression Chain Graphs via Decomposition

Mohammad Ali Javidian JAVIDIAN@EMAIL.SC.EDU

Marco Valtorta MGV@CSE.SC.EDU

Department of Computer Science & Engineering, University of South Carolina, Columbia, SC, 29201, USA.

Abstract

We extend the decomposition approach for learning Bayesian networks (BN) proposed by (Xie et al., 2006) to learning multivariate regression chain graphs (MVR CGs), which include BNs as a special case. The same advantages of this decomposition approach hold in the more general setting: reduces complexity and increased power of computational independence tests. Moreover, latent (hidden) variables can be represented in MVR CGs by using bidirected edges, and our algorithm correctly recovers any independence structure that is faithful to a MVR CG, thus greatly extending the range of applications of decomposition-based model selection techniques. While our new algorithm has the same complexity as the one in (Xie et al., 2006) for BNs, it requires larger components for general MVR CGs, to insure that sufficient data is present to estimate parameters.

Keywords: MVR chain graph; conditional independence; decomposition; m-separator; junction tree; augmented graph; triangulation; graphical model; Markov equivalent; structural learning.

1. Introduction

Probabilistic graphical models (PGM) use graphs, either undirected, directed, or mixed, to represent possible dependences among the variables of a multivariate probability distribution. Two types of graphical representations of distributions are commonly used, namely, Bayesian networks and Markov random fields (Markov networks). Both families encompass the properties of factorization and independences, but they differ in the set of independences they can encode and the factorization of the distribution that they induce.

Currently systems containing both causal and non-causal relationships are mostly modelled with directed acyclic graphs (DAGs). An alternative approach is using chain graphs (CGs). CGs contain two types of edges, the directed edge that corresponds to the causal relationship in DAGs and a second type of edge representing a symmetric relationship (Sonntag, 2016). While the interpretation of the directed edge in a CG is quite clear, the second type of edge can represent different types of relations and, depending on how we interpret it in the graph, we say that we have different CG interpretations. There are several possible interpretations of CGs with different separation criteria, i.e. different ways of reading conditional independences from the graph, and different intuitive meaning behind their edges. The first interpretation (LWF) was introduced by Lauritzen, Wermuth and Frydenberg (Lauritzen and Wermuth, 1989; Frydenberg, 1990) to combine DAGs and UGs. The second interpretation (AMP), was introduced by Andersson, Madigan and Perlman, and also combines DAGs and undirected graphs (UGs) but with a separation criterion that more closely resembles the one of DAGs (Andersson et al., 1996). The third interpretation, the multivariate regression interpretation (MVR), was introduced by Cox and Wermuth (Cox and Wermuth, 1993, 1996) to combine DAGs and bidirected (covariance) graphs.

Sonntag lists four constraint-based learning algorithms for CGs. All are based on testing if variables are (conditionally) independent in the data using an independence test, and using this information to deduce the structure of the optimal graph. These algorithms are the PC-like algorithms (Studený, 1997; Peña, 2014; Sonntag and Peña, 2012), the answer set programming (ASP) algorithms (Peña, 2016; Sonntag et al., 2015), the LCD algorithm (Ma et al., 2008) and the CKES algorithm (Peña et al., 2014). The former two have implementations for all three CG interpretations, while the latter two are only applicable for LWF CGs (Sonntag, 2016).

In this paper, we propose a decomposition approach for recovering structures of MVR CGs. Our algorithms are natural extensions of algorithms in (Xie et al., 2006). In particular, the rule in (Xie et al., 2006) for combining local structures into a global skeleton is still applicable and no more careful work (unlike, for example, algorithms in (Ma et al., 2008)) must be done to ensure a valid combination. Moreover, the method for extending a global skeleton to a Markov equivalence class is exactly the same as that for Bayesian networks.

Section 2 gives notation and definitions. In Section 3, we show a condition for decomposing structural learning of MVR CGs. Construction of m-separation trees to be used for decomposition is discussed in Section 4. We propose the main algorithm and then give an example in Section 5 to illustrate our approach for recovering the global structure of a MVR CG. In Section 6, we propose a new algorithm for structural learning of MVR CGs via decomposition by constructing a JV-junction tree from domain knowledge or from observed data patterns. Section 7 discusses the complexity and advantages of the proposed algorithms. The proofs of our main results and algorithms are given in Appendix A and B.

2. Definitions and Concepts

In this paper we consider graphs containing both directed ($\to$) and bidirected ($\leftrightarrow$) edges and largely use the terminology of (Xie et al., 2006; Richardson, 2003), where the reader can also find further details. Below we briefly list some of the most central concepts used in this paper.

If there is an arrow from a pointing towards b, a is said to be a parent of b. The set of parents of b is denoted as $pa(b)$. If there is a bidirected edge between a and b, a and b are said to be neighbors. The set of neighbors of a vertex a is denoted as $ne(a)$. The expressions $pa(A)$ and $ne(A)$ denote the collection of parents and neighbors of vertices in A that are not themselves elements of A. The boundary $bd(A)$ of a subset A of vertices is the set of vertices in $V \setminus A$ that are parents or neighbors to vertices in A.

A path of length n from a to b is a sequence $a = a_0, \dots, a_n = b$ of distinct vertices such that $(a_i \to a_{i+1}) \in E$, for all $i = 1, \dots, n$. A chain of length n from a to b is a sequence $a = a_0, \dots, a_n = b$ of distinct vertices such that $(a_i \to a_{i+1}) \in E$, or $(a_{i+1} \to a_i) \in E$, or $(a_{i+1} \leftrightarrow a_i) \in E$, for all $i = 1, \dots, n$. We say that u is an ancestor of v and v is a descendant of u if there is a path between u and v in G. The set of ancestors of v is denoted as $an(v)$, and we define $An(v) = an(v) \cup v$. We apply this definition to sets: $an(X) = \{\alpha | \alpha \text{ is an ancestor of } \beta \text{ for some } \beta \in X\}$. A partially directed cycle in a graph G is a sequence of n distinct vertices $v_1, \dots, v_n (n \geq 3)$, and $v_{n+1} \equiv v_1$, such that

- $\forall i (1 \leq i \leq n)$ either $v_i \leftrightarrow v_{i+1}$ or $v_i \to v_{i+1}$, and
- $\exists j (1 \leq j \leq n)$ such that $v_i \to v_{i+1}$.

A graph with only undirected edges is called an undirected graph (UG). A graph with only directed edges and without directed cycles is called a directed acyclic graph (DAG). Acyclic directed mixed graphs, also known as semi-Markov(ian) (Pearl, 2009) models contain directed ($\to$) and bi-directed ($\leftrightarrow$) edges subject to the restriction that there are no directed cycles (Richardson, 2003, 2009). A graph that has no partially directed cycles is called a MVR chain graph.

A nonendpoint vertex ζ on a chain is a *collider* on the chain if the edges preceding and succeeding ζ on the chain have an arrowhead at ζ, that is, $\to \zeta \leftarrow, or \leftrightarrow \zeta \leftrightarrow, or \leftrightarrow \zeta \leftarrow, or \to \zeta \leftrightarrow$. A nonendpoint vertex ζ on a chain which is not a collider is a noncollider on the chain. A chain between vertices α and β in a MVR chain graph G is said to be m-connecting given a set Z (possibly empty), with $\alpha, \beta \notin Z$, if:

(i) every noncollider on the path is not in Z, and

(ii) every collider on the path is in $An_G(Z)$.

A chain that is not m-connecting given Z is said to be blocked given (or by) Z. If there is no chain m-connecting α and β given Z, then α and β are said to be *m-separated* given Z. Sets X and Y are m-separated given Z, if for every pair α, β, with $\alpha \in X$ and $\beta \in Y$, α and β are m-separated given Z (X, Y, and Z are disjoint sets; X, Y are nonempty). We denote the independence model resulting from applying the m-separation criterion to G, by $\Im_m$(G). This is an extension of Pearl's d-separation criterion (Pearl, 1988) to MVR chain graphs in that in a DAG D, a chain is d-connecting if and only if it is m-connecting.

Two vertices x and y in a MVR chain graph G are said to be collider connected if there is a chain from x to y in G on which every non-endpoint vertex is a collider; such a chain is called a collider chain. Note that a single edge trivially forms a collider path, so if x and y are adjacent in a MVR chain graph then they are collider connected. The augmented graph derived from G, denoted $(G)^a$, is an undirected graph with the same vertex set as G such that

$$c - d \text{ in } (G)^a \Leftrightarrow c \text{ and } d \text{ are collider connected in } G.$$

Disjoint sets $X, Y \neq \emptyset$, and Z (Z may be empty) are said to be m^*-separated if X and Y are separated by Z in $(G_{an(X \cup Y \cup Z)})^a$. Otherwise X and Y are said to be m^*-connected given Z. The resulting independence model is denoted by $\Im_{m^*}(G)$.

According to (Richardson and Spirtes, 2002, Theorem 3.18.) and (Javidian and Valtorta, 2018a, Theorem 15), for a MVR chain graph G we have: $\Im_m(G) = \Im_{m^*}(G)$.

Let $\bar{G}_V = (V, \bar{E}_V)$ denote an undirected graph where $\bar{E}_V$ is a set of undirected edges. An undirected edge between two vertices u and v is denoted by (u, v). For a subset A of V, let $\bar{G}_A = (A, \bar{E}_A)$ be the subgraph induced by A and $\bar{E}_A = \{e \in \bar{E}_V | e \in A \times A\} = \bar{E}_V \cap (A \times A)$. An undirected graph is called complete if any pair of vertices is connected by an edge. For an undirected graph, we say that vertices u and v are separated by a set of vertices Z if each path between u and v passes through Z. We say that two distinct vertex sets X and Y are separated by Z if and only if Z separates every pair of vertices u and v for any $u \in X$ and $v \in Y$. We say that an undirected graph $\bar{G}_V$ is an undirected independence graph for a MVR CG G if the fact that a set Z separates X and Y in $\bar{G}_V$ implies that Z m-separates X and Y in G. Note that the augmented graph derived from a MVR CG G, $(G)^a$, is an undirected independence graph for G. We say that $\bar{G}_V$ can be decomposed into subgraphs $\bar{G}_A$ and $\bar{G}_B$ if

(1) $A \cup B = V$, and

(2) $C = A \cap B$ separates $V \setminus A$ and $V \setminus B$ in $\bar{G}_V$.

The above decomposition does not require that the separator C be complete, which is required for weak decomposition defined in (Lauritzen, 1996). In the next section, we show that a problem of structural learning of a MVR CG can also be decomposed into problems for its decomposed subgraphs even if the separator is not complete.

A triangulated (chordal) graph is an undirected graph in which all cycles of four or more vertices have a chord, which is an edge that is not part of the cycle but connects two vertices of the cycle (see, for example, Figure 1). For an undirected graph $\bar{G}_V$ which is not triangulated, we can add extra edges to it such that it becomes to be a triangulated graph, denoted by $\bar{G}_V^t$.

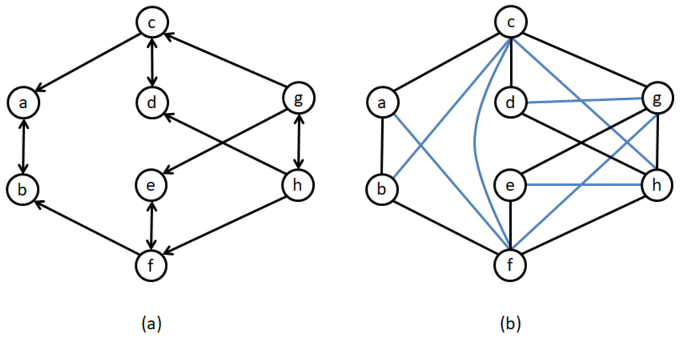

Figure 1: (a) A MVR CG G. (b) The augmented graph G^a, which is also a triangulated graph G^t.

Let $X \perp\!\!\!\perp Y$ denote the independence of X and Y, and $X \perp\!\!\!\perp Y|Z$ (or $\langle X, Y|Z\rangle$) the conditional independence of X and Y given Z. In this paper, we assume that all independencies of a probability distribution of variables in V can be checked by m-separations of G, called the faithfulness assumption (Spirtes et al., 2000). The faithfulness assumption means that all independencies and conditional independencies among variables can be represented by G.

The global skeleton is an undirected graph obtained by dropping direction of a MVR CG. Thus the absence of an edge (u, v) implies that there is a variable subset S of V such that u and v are independent conditional on S, that is, $u \perp\!\!\!\perp v|S$ for some $S \subseteq V \setminus \{u, v\}$. Two MVR CGs over the same variable set are called Markov equivalent if they induce the same conditional independence restrictions. Two MVR CGs are Markov equivalent if and only if they have the same global skeleton and the same set of v-structures (unshielded colliders) (Wermuth and Sadeghi, 2012). An equivalence class of MVR CGs consists of all MVR CGs which are Markov equivalent, and it is represented as a partially directed graph (i.e., a graph containing directed, undirected, and bidirected edges and no directed cycles) where the directed/bidirected edges represent edges that are common to every MVR CG in it, while the undirected edges represent that any appropriate orientation of them leads to a Markov equivalent MVR CG. Therefore the goal of structural learning is to construct a partially directed graph to represent the equivalence class. A local skeleton for a subset A of variables is an undirected subgraph for A in which the absence of an edge (u, v) implies that there is a subset S of A such that $u \perp\!\!\!\perp v|S$.

Now, we introduce the notion of m-separation trees, which is used to facilitate the representation of the decomposition. The concept is similar to the junction tree of cliques and the independence tree introduced for DAGs as d-separation trees in (Xie et al., 2006). Let $C = \{C_1, \ldots, C_H\}$ be a collection of distinct variable sets such that for $h = 1, \ldots, H, C_h \subseteq V$. Let T be a tree where each node corresponds to a distinct variable set in C, to be displayed as an oval (see, for example, Figure 2). The term 'node' is used for a m-separation tree to distinguish from the term 'vertex' for a graph in general. An undirected edge $e = (C_i, C_j)$ connecting nodes C_i and C_j in T is labeled with a separator $S = C_i \cap C_j$, which is displayed as a rectangle. Removing an edge e or equivalently, removing a separator S from T splits T into two subtrees T_1 and T_2 with node sets C_1 and C_2 respectively. We use $V_i = \cup_{C \in C_i} C$ to denote the union of the vertices contained in the nodes of the subtree T_i for $i = 1, 2$.

Definition 1 *A tree T with node set C is said to be a m-separation tree for a MVR chain graph $G = (V, E)$ if*

- $\cup_{C_i \in C} C_i = V$, *and*
- *for any separator S in T with V_1 and V_2 defined as above by removing S, we have $\langle V_1 \setminus S, V_2 \setminus S | S \rangle_G$.*

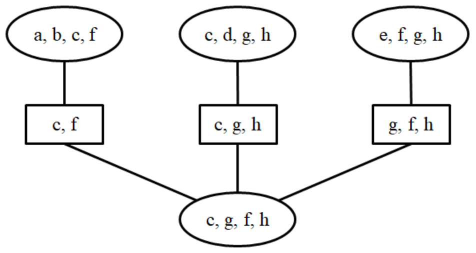

Figure 2: A m-separation tree.

Notice that a separator is defined in terms of a tree whose nodes consist of variable sets, while the m-separator is defined based on a MVR chain graph. In general, these two concepts are not related, though for a m-separation tree its separator must be some corresponding m-separator in the underlying MVR chain graph. The definition of m-separation trees for MVR chain graphs is similar to that of junction trees of cliques, see (Cowell et al., 1999; Lauritzen, 1996). Actually, it is not difficult to see that a junction tree of a MVR chain graph G is also a m-separation tree. However, as in (Ma et al., 2008), we point out two differences here: (a) a m-separation tree is defined with m-separation and it does not require that every node is a clique or that every separator is complete on the augmented graph; (b) junction trees are mostly used as inference engines, while our interest in m-separation trees is mainly derived from its power in facilitating the decomposition of structural learning.

A collection of variable sets $C = \{C_1, \ldots, C_H\}$ is said to be a hypergraph on V where each hyperedge C_h is a nonempty subset of variables, and $\cup_{h=1}^{H} C_h = V$. A hypergraph is a reduced hypergraph if $C_i \not\subseteq C_j$ for $i \neq j$. In this paper, only reduced hypergraphs are used, and thus simply called hypergraphs. In Section 4, a hyperedge will be used to represent the domain knowledge of association among variables (see, for example, Figure 4, (b)).

3. Decomposition of Structural Learning

Applying the following theorem to structural learning, we can split a problem of searching for m–separators and building the skeleton of a MVR CG into small problems for every node of m-separation tree T.

Theorem 2 *Let T be a m-separation tree for a MVR CG G. Vertices u and v are m-separated by $S \subseteq V$ in G if and only if (i) u and v are not contained together in any node C of T or (ii) there exists a node C that contains both u and v such that a subset S' of C, m-separates u and v.*

According to Theorem 2, a problem of searching for a m-separator S of u and v in all possible subsets of V is localized to all possible subsets of nodes in a m-separation tree that contain u and v. For a given m-separation tree T with the node set $C = \{C_1, \ldots, C_H\}$, we can recover the skeleton and all v-structures for a MVR CG as follows. First we construct a local skeleton for every node C_h of T, which is constructed by starting with a complete undirected subgraph and removing an undirected edge (u, v) if there is a subset S of C_h such that u and v are independent conditional on S. Then, in order to construct the global skeleton, we combine all these local skeletons together and remove edges that are present in some local skeletons but absent in other local skeletons. Then we determine every v-structure if two non-adjacent vertices u and v have a common neighbor in the global skeleton but the neighbor is not contained in the m-separator of u and v. Finally we can orient more undirected edges if none of them creates either a partially directed cycle or a new v-structure (see, for example, Figure 3). This process is formally described in the following algorithm:

Algorithm 1 (Construct the equivalence class of MVR CGs from a m-separation tree).

1. **Input**: a m-separation tree T with a node set $C = \{C_1, \ldots, C_H\}$.
2. Construct a local skeleton $\bar{G}_h$ for each tree node C_h:
    - Initialize $\bar{G}_h$ as a complete undirected graph;
    - If there exists a subset S_{uv} of $C_h \setminus \{u, v\}$ such that $u \perp\!\!\!\perp v | S_{uv}$, then delete edge (u, v) from $\bar{G}_h$ and save S_{uv} to the m-separator list S.
3. Construct the global skeleton $\bar{G}_V$:
    - Initialize the edge set $\bar{E}_V$ of $\bar{G}_V$ as the union of all edge sets of $\bar{G}_h, h = 1, \ldots, H$;
    - For a pair of vertices u and v contained in several local skeletons, delete edge (u, v) from $\bar{E}_V$ if it is absent in some skeleton.
4. For each m-separator S_{uv} in the list S, determine a v-structure $(u \circ\!\!\rightarrow w \leftarrow\!\!\circ v)$ if $(u \circ\!\!- w -\!\!\circ v)$ appears in the global skeleton and w is not in S_{uv}.
5. **Output**: the equivalence class of MVR CGs.

In order to obtain MVR CGs from an equivalence class, the following step may be added between steps 4 and 5:

- Orient other edges in a way that none of them creates either a partially directed cycle or a new v-structure.

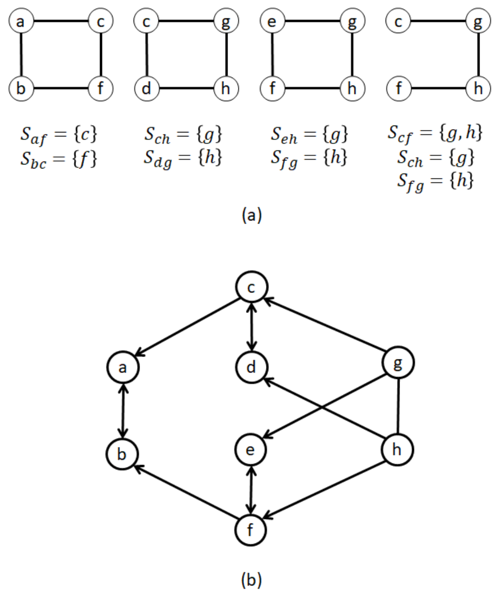

Figure 3: (a) Local skeletons for every node of T. (b) The global skeleton and all v-structures.

According to Theorem 2, we can prove that the global skeleton and all v-structures obtained by applying the above decomposition algorithm are correct, that is, they are the same as those obtained from the joint distribution of V, see Appendix A for the details of proof. Note that separators in a m-separation tree may not be complete in the augmented graph. Thus the decomposition is weaker than the decomposition usually defined for parameter estimation (Cowell et al., 1999; Lauritzen, 1996).

4. Construction of m-Separation Trees

As proposed in (Xie et al., 2006), one can construct a d-separation tree from observed data, from domain or prior knowledge of conditional independence relations or from a collection of databases. However, their arguments are not valid for constructing a m-separation tree from domain knowledge or from observed data patterns.

4.1 Constructing a m-Separation Tree from Observed Data

In several algorithms for structural learning of PGMs, the first step is to construct an undirected independence graph in which the absence of an edge (u, v) implies $u \perp\!\!\!\perp v | V \setminus \{u, v\}$. To construct such an undirected graph, we can start with a complete undirected graph, and then for each pair of variables u and v, an undirected edge (u, v) is removed if u and v are independent conditional on the set of all other variables (Xie et al., 2006). A m-separation tree can be built by constructing a junction tree from an undirected independence graph.

Theorem 3 *A junction tree constructed from an undirected independence graph for MVR CG G is a m-separation tree for G.*

4.2 Constructing a m-Separation Tree from Domain Knowledge or from Observed Data Patterns

In this subsection, we propose an approach for constructing a m-separation tree from domain knowledge or from observed data patterns without conditional independence tests. Domain knowledge of variable dependencies can be represented as a collection of variable sets $C = \{C_1, \dots, C_H\}$, in which variables contained in the same set may associate with each other directly but variables contained in different sets associate with each other through other variables. This means that two variables that are not contained in the same set are independent conditionally on all other variables. On the other hand, in an application study, observed data may have a collection of different observed patterns, $C = \{C_1, \dots, C_H\}$, where C_h is the set of observed variables for the hth group of individuals. In both cases, the condition to make our algorithms correct for structural learning from a collection C is that C must contain sufficient data such that parameters of the underlying MVR CG are estimable.

For a DAG, parameters are estimable if, for each variable u, there is an observed data pattern C_h in C that contains both u and its parent set. Thus a collection C of observed patterns has sufficient data for correct structural learning if there is a pattern C_h in C for each u such that C_h contains both u and its parent set in the underlying DAG. Also, domain knowledge is legitimate if, for each variable u, there is a hyperedge C_h in C that contains both u and its parent set (Xie et al., 2006). However, these conditions are not valid in the case of MVR chain graphs. In fact, for MVR CGs domain knowledge is legitimate if for each connected component τ, there is a hyperedge C_h in C that contains both τ and its parent set $pa_G(\tau)$. Also, a collection C of observed patterns has sufficient data for correct structural learning if there is a pattern C_h in C for each connected component τ such that C_h contains both τ and its parent set $pa_G(\tau)$ in the underlying MVR CG.

Algorithm 2 (Construct a m-separation tree from a hypergraph).

1. **Input**: a hypergraph $C = \{C_1, \dots, C_H\}$, where each hyperedge C_h is a variable set such that for each connected component τ, there is a hyperedge C_h in C that contains both τ and its parent set $pa_G(\tau)$.

2. For each hyperedge C_h, construct a complete undirected graph $\bar{G}_h$ with the edge set $\bar{E}_h = \{(u, v) | \forall u, v \in C_h\} = C_h \times C_h$.

3. Construct the entire undirected graph $\bar{G}_V = (V, \bar{E})$, where $\bar{E} = \bar{E}_1 \cup \ldots \cup \bar{E}_H$.

4. Construct a junction tree T by triangulating $\bar{G}_V$.

5. **Output**: T, which is a m-separation tree for the hypergraph C.

The correctness of Algorithm 2 is proven in Appendix B. Note that we do not need any conditional independence test in Algorithm 2 to construct a m-separation tree. In this algorithm, we can use the proposed algorithm in (Berry et al., 2004) to construct a minimal triangulated graph. In order to illustrate Algorithm 2, see Figure 4.

The example illustrated in Figure 5 shows that, if for each variable u there is a hyperedge C_h in C that contains both u and its parent set, we cannot guarantee the correctness of our algorithm. Note that vertices a and d are separated in the tree T of Figure 5 part (d) by removing vertex b, but a and d are not m-separated given b as can be verified using 5 part (a).

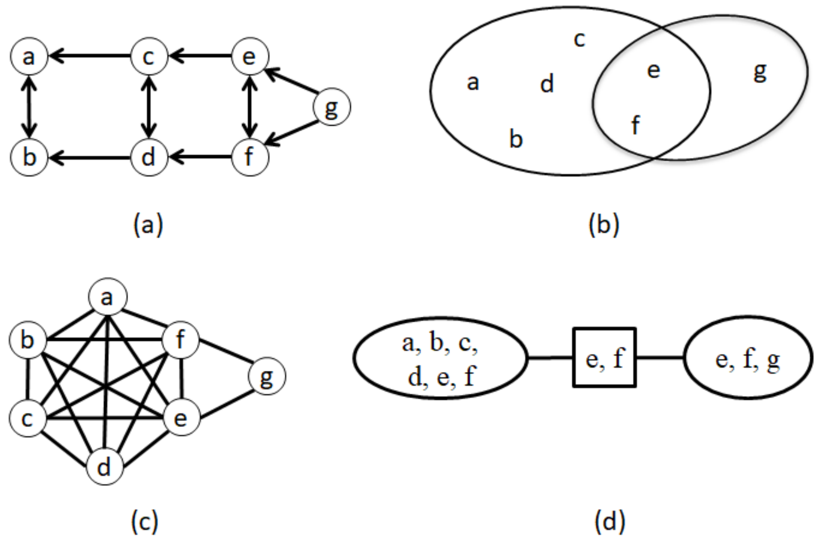

Figure 4: Construction the m-separation tree. (a) A MVR CG. (b) Domain knowledge of associations. (c) The undirected graph and triangulation. (d) The m-separation tree T.

The example illustrated in Figure 6 shows that, if for each variable u there is a hyperedge C_h in C that contains both u and its boundary set, Algorithm 2 does not necessarily give a m-separation tree because, for example, $S = \{a, b\}$ separates c and d in tree T of Figure 6 part (d), but S does not m-separate c and d in the MVR CG G in Figure 6 part (a).

The two previous examples show that, in order to build a m-separation tree using Algorithm 2, it is necessary that, for each connected component τ, there is a $C_h \in C$ that contains both τ and its parent set $pa_G(\tau)$ in the underlying MVR CG. However, surprisingly, for the example of Figure 6, we can still recover the MVR CG G if we use the hypergraph in Figure 6 part (b) as the input of Algorithm 2. Building on this observation, we propose a new structural learning method for MVR CGs in section 6 and conjecture that it is correct; a proof is forthcoming.

One of the most important differences between our algorithms and algorithms proposed in (Xie et al., 2006) is that (Xie et al., 2006) assumes that there are no latent (hidden) variables, while in this paper we deal with situations where there are latent variables.

5. Structural Learning via Decomposition

In this section, we formally describe the complete algorithm for structural learning of MVR CGs via decomposition.

Main Algorithm (The decomposition approach for structural learning of MVR CGs with domain knowledge).

1. **Input**: a hypergraph $C = \{C_1, \ldots, C_H\}$ (as the domain knowledge, or databases with observed data patterns C).

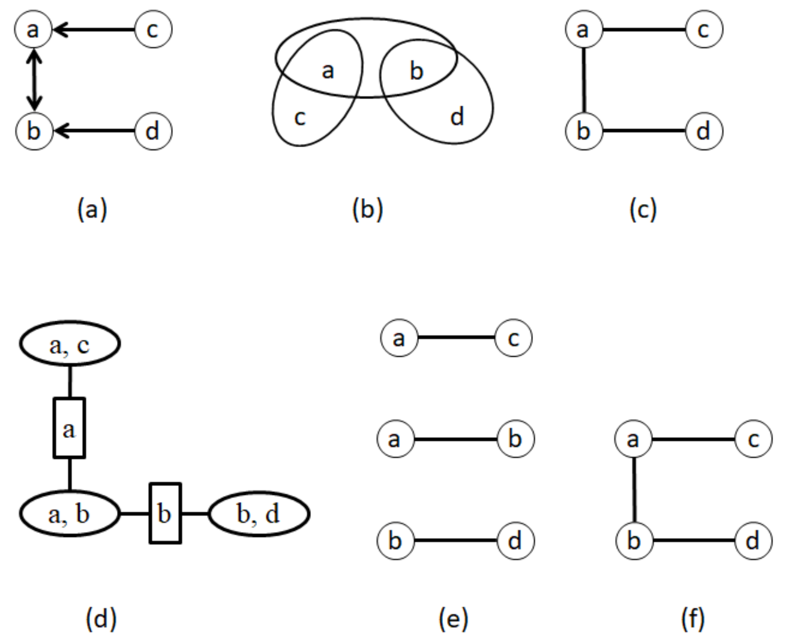

Figure 5: Insufficiency of having a hypergraph that contains both u and its parent set for every $u \in V$. (a) A MVR CG. (b) Domain knowledge of associations. (c) The undirected graph constructed by union of complete graphs corresponding to each hyperedge, which is also a triangulated graph. (d) The junction tree T. (f) Local skeleton for every node of T. (g) The global skeleton and all v-structures.

2. Call Algorithm 2 to construct a m-separation tree T from the hypergraph C.

3. If the sizes of nodes in T are too large, then refine T with observed data:

   - Construct an undirected independence subgraph over each node of T;
   - Construct the entire undirected independence graph $\bar{G}_V = (V, \bar{E})$, where $\bar{E}$ is the union of all edge sets of subgraphs;
   - Construct a junction tree T' by triangulating $\bar{G}_V$, which is a m-separation tree.

4. Call Algorithm 1 to construct the equivalence class of MVR CGs from T' (or T if the sizes of nodes are not very large).

5. **Output**: the equivalence class of MVR CGs.

The correctness of the main algorithm is proven in Appendix B. In order to illustrate the Main Algorithm, see Figure 7.

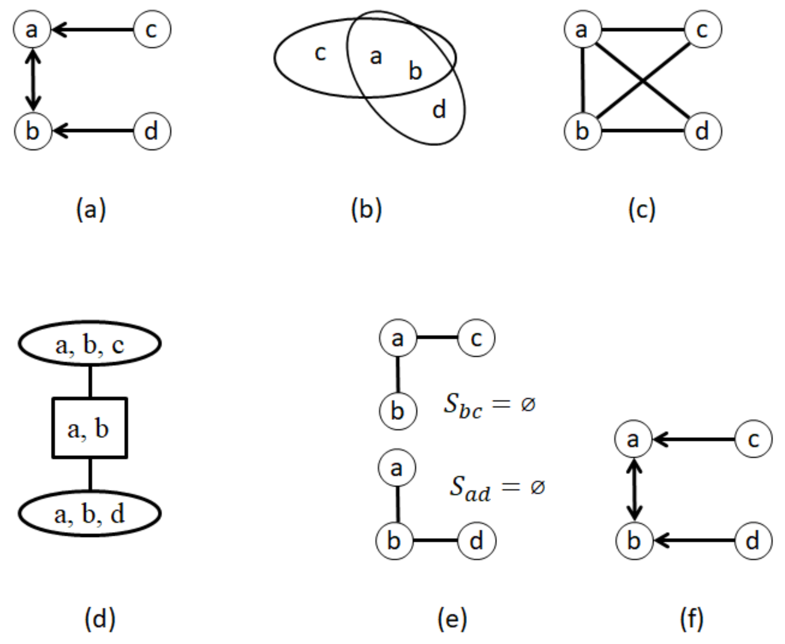

Figure 6: Sufficiency of having a hypergraph that contains both u and its boundary set for every $u \in V$. (a) A MVR CG. (b) Domain knowledge of associations. (c) The undirected graph constructed by union of complete graphs corresponding to each hyperedge, which is also a triangulated graph. (d) The junction tree T. (f) Local skeleton for every node of T. (g) The global skeleton and all v-structures.

6. Structural Learning via Decomposition by Constructing a JV-Junction Tree from Domain Knowledge or from Observed Data Patterns

In this section, we formally describe a new algorithm for structural learning of MVR CGs via decomposition by constructing a JV-junction tree from domain knowledge or from observed data patterns. We observed in subsection 4.2 that if we modify Algorithm 2 in a such a way that, for each variable u, there is a hyperedge C_h in C that contains both u and its boundary set, Algorithm 2 does not necessarily give a m-separation tree. However, we can still recover the MVR CG G if we use the junction tree of step 4, which we call JV-junction tree, as input to Algorithm 1. This process is formally described in the following algorithm:

***JV*-Junction Tree Algorithm** (The decomposition approach for structural learning of MVR CGs with domain knowledge by constructing a JV-junction tree from domain knowledge or from observed data patterns).

1. **Input**: a hypergraph $C = \{C_1, \ldots, C_H\}$ (as the domain knowledge, or databases with observed data patterns C) such that for each vertex u, there is a hyperedge C_h in C that contains both u and its boundary set.

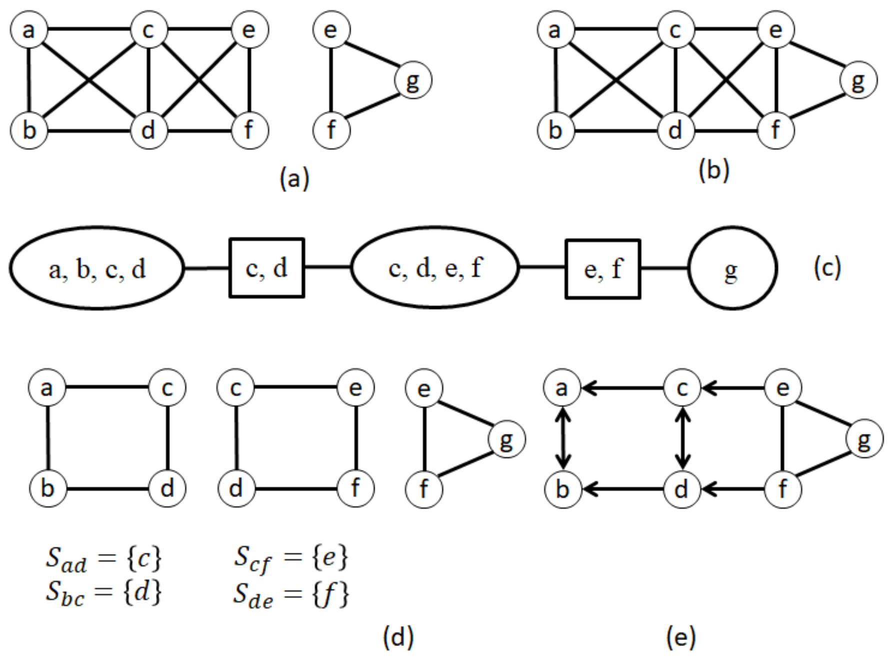

Figure 7: Main algorithm. (a) Undirected independence subgraphs for each node. (b) Combining subgraphs in (a). (c) The m-separation tree T'. (d) Local skeletons for every node of T'. (e) The global skeleton and all v-structures.

2. Call Algorithm 2, but with the input of step 1, to construct a JV-junction tree T from the hypergraph C. (T is not necessarily a m-separation tree.)
3. If the sizes of nodes in T are too large, then refine T with observed data:
   - Construct an undirected independence subgraph over each node of T;
   - Construct the entire undirected graph $\bar{G}_H = (V, \bar{E})$, where $\bar{E}$ is the union of all edge sets of subgraphs;
   - Construct a JV-junction tree T' by triangulating $\bar{G}_H$.
4. Call Algorithm 1, but with T' as input, to construct the equivalence class of MVR CGs from T' (or T if the sizes of nodes are not very large).
5. **Output**: the equivalence class of MVR CGs.

A proof of correctness of the JV-Junction Tree Algorithm is forthcoming. For an example of use of the JV-Junction Tree Algorithm, see Figure 6.

7. Complexity Analysis and Advantages

In this section, we start by comparing our algorithm with the main algorithm in (Xie et al., 2006) that is designed specifically for DAG structural learning when the underlying graph structure is a

DAG. We make this choice of the DAG specific algorithm so that both algorithms can have the same separation tree as input and hence are directly comparable.

In a DAG, all chain components are singletons. Therefore, sufficiency of having a hypergraph that contains both τ and its parent set for every chain component is equivalent with having a hypergraph that contains both u and its parent set for every $u \in V$, when the underlying graph structure is a DAG. Therefore, it is obvious that our algorithm has the same effect and the same complexity as the main algorithm in (Xie et al., 2006).

The same advantages mentioned by (Xie et al., 2006) for their BN structural learning algorithm hold for our algorithm when applied to MVR CGs. For the reader convenience, we list them here. First a m-separation tree can be constructed based on the prior or domain knowledge rather than conditional independence tests. By using the m-separation tree, independence tests are performed only conditionally on smaller sets contained in a node of the m-separation tree rather than on the full set of all other variables. Thus our algorithm has higher power for statistical tests. Second, the theoretical results proposed in this paper can be applied to scheme design of multiple databases. Without loss of information on structural learning of MVR CGs, a joint data set can be replaced by a group of incomplete data sets based on the domain or prior knowledge of conditional independencies among variables (Xie et al., 2006).

Finally, the computational complexity can be reduced. This complexity analysis focuses only on the number of conditional independence tests for constructing the equivalence class. Decomposition of graphs is a computationally simple task compared to the task of testing conditional independence for a large number of triples of sets of variables. The triangulation of an undirected graph is used in our algorithms to construct a m-separation from an undirected independence graph. Although the problem for optimally triangulating an undirected graph is NP-hard, sub-optimal triangulation methods (Berry et al., 2004) may be used provided that the obtained tree does not contain too large nodes to test conditional independencies. Two of the best known algorithms are lexicographic search and maximum cardinality search, and their complexities are $O(|V||E|)$ and $O(|V|+|E|)$, respectively (Berry et al., 2004). Thus in our algorithms, the conditional independence tests dominate the algorithmic complexity.

The complexity of the Main Algorithm and JV-Junction Tree Algorithm is $O(Hm^2 2^m)$ as claimed in (Xie et al., 2006, Section 6), where H is the number of hyperedges (usually $H \ll |V|$) and $m = \max_h |C_h|$ where $|C_h|$ denotes the number of variables in C_h (m usually is much less than $|V|$).

Appendix A. Proofs of Theoretical Results

Lemma 4 *Let ρ be a chain from u to v, and W be the set of all vertices on ρ (W may or may not contain u and v). Suppose that (the endpoints of) a chain ρ is by S. If $W \subseteq S$, then the chain ρ is blocked by W and by any set containing W.*

Proof Since the blocking of the chain ρ depends on those vertices between u and v that are contained in the m-separator, and since W contains all vertices on ρ, ρ is also blocked by $S \cap W = W$ if ρ is blocked by S. Since all colliders on ρ have already been activated conditionally on W, adding other vertices into the conditional set does not make any new collider active on ρ. This implies that ρ is blocked by any set containing W. ■

Lemma 5 *Let T be a m-separation tree for a MVR CG G, and K be a separator of T that separates T into two subtrees T_1 and T_2 with variable sets V_1 and V_2 respectively. Suppose that ρ is a chain from u to v in G where $u \in V_1 \setminus K$ and $v \in V_2 \setminus K$. Let W denote the set of all vertices on ρ (W may or may not contain u and v). Then the chain ρ is blocked by $W \cap K$ and by any set containing $W \cap K$.*

Proof Since $u \in V_1 \setminus K$ and $v \in V_2 \setminus K$, there is a sequence from s (may be u) to y (may be v) in $\rho = (u, \ldots, s, t, \ldots, x, y, \ldots, v)$ such that $s \in V_1 \setminus K$ and $y \in V_2 \setminus K$ and all vertices from t to x are contained in K. Let ρ' be the sub-chain of ρ from s to y and W' the vertex set from t to x, so $W' \subseteq K$. Since $s \in V_1 \setminus K$ and $y \in V_2 \setminus K$, we have from definition of m-separation tree that K m-separates s and y in G, i.e., K blocks ρ'. By lemma 4, we obtain that ρ' is blocked by $W'(\subseteq K)$ and any set containing W'. Since $W' \subseteq (K \cap W)$, ρ' is blocked by $K \cap W$ and by any set containing $K \cap W$. Thus $\rho(\supseteq \rho')$ is also blocked by them. ■

Remark 6 *Javidian and Valtorta showed that if we find a separator over S in $(G_{An(u \cup v)})^a$ then it is a m-separator in G. On the other hand, if there exists a m-separator over S in G then there must exist a separator over S in $(G_{An(u \cup v)})^a$ by removing all nodes which are not in $An(u \cup v)$ from it (Javidian and Valtorta, 2018b).*

Observations in Remark 6 yield the following results.

Lemma 7 *Let u and v be two non-adjacent vertices in MVR CG G, and let ρ be a chain from u to v. If ρ is not contained in $An(u \cup v)$, then ρ is blocked by any subset S of $an(u \cup v)$.*

Proof Since $\rho \not\subseteq An(u \cup v)$, there is a sequence from s (may be u) to y (may be v) in $\rho = (u, \ldots, s, t, \ldots, x, y, \ldots, v)$ such that s and y are contained in $An(u \cup v)$ and all vertices from t to x are out of $An(u \cup v)$.Then the edges $s - t$ and $x - y$ must be oriented as $s \circ\!\!\rightarrow t$ and $x \leftarrow\!\!\circ\, y$, otherwise t or x belongs to $an(u \cup v)$. Thus there exist at least one collider between s and y on ρ. The middle vertex w of the collider closest to s between s and y is not contained in $an(u \cup v)$, and any descendant of w is not in $an(u \cup v)$, otherwise there is a (partially) directed cycle. So ρ is blocked by the collider, and it cannot be activated conditionally on any vertex in S where $S \subseteq an(u \cup v)$. ■

Lemma 8 *Let T be a m-separation tree for a MVR CG G. For any vertex u there exists at least one node of T that contains u and $bd(u)$.*

Proof If $bd(u)$ is empty, it is trivial. Otherwise let C denote the node of T which contains u and the most number of elements of u's boundary. Since no set can separate u from a parent (or neighbor), there must be a node of T that contains u and the parent (or neighbor). If u has only one parent (or neighbor), then we obtain the lemma. If u has two or more elements in its boundary, we choose two arbitrary elements v and w of u's boundary that are not contained in a single node but are contained in two different nodes of T, say $\{u, v\} \subseteq C$ and $\{u, w\} \subseteq C'$ respectively, since all vertices in V appear in T. On the chain from C to C' in T, all separators must contain u, otherwise they cannot separate C from C'. However, any separator containing u cannot separate v and w because $v \circ\!\!\rightarrow u \leftarrow\!\!\circ\, w$ is an active chain between v and w in G. Thus we got a contradiction. ■

Lemma 9 *Let T be a m-separation tree for a MVR CG G and C a node of T. If u and v are two vertices in C that are non-adjacent in G, then there exists a node C' of T containing u, v and a set S such that S m-separates u and v in G.*

Proof Without loss of generality, we can suppose that v is not a descendant of the vertex u in G, i.e., $v \in nd(u)$. According to the local Markov property for MVR chain graphs proposed by Javidian and Valtorta in (Javidian and Valtorta, 2018a), we know that $u \perp\!\!\!\perp [nd(u) \setminus bd(u)] | pa_G(u)$. By Lemma 8, there is a node C_1 of T that contains u and $bd(u)$. If $v \in C_1$, then S defined as the parents of u m-separates u from v.

If $v \notin C_1$, choose the node C_2 that is the closest node in T to the node C_1 and that contains u and v. Consider that there is at least one parent (or neighbor) p of u that is not contained in C_2. Thus there is a separator K connecting C_2 toward C_1 in T such that K m-separates p from all vertices in $C_2 \setminus K$. Note that on the chain from C_1 to C_2 in T, all separators must contain u, otherwise they cannot separate C_1 from C_2. So, we have $u \in K$ but $v \notin K$ (if $v \in K$, then C_2 is not the closest node of T to the node C_1). In fact, for every parent (or neighbor) p' of u that is contained in C_1 but not in C_2, K separates p' from all vertices in $C_2 \setminus K$, especially the vertex v.

Define $S = (an(u \cup v) \cap C_2)$, which is a subset of C_2. We need to show that u and v are m-separated by S, that is, every chain between u and v in G is blocked by S.

If ρ is not contained in $An(u \cup v)$, then we obtain from Lemma 7 that ρ is blocked by S.

When ρ is contained in $An(u \cup v)$, let x be adjacent to u on ρ, that is, $\rho = (u, x, y, \dots, v)$. We consider the three possible orientations of the edge between u and x. We now show that ρ is blocked in all three cases.

i: $u \leftarrow x$, so we know that x is not a collider and we have two possible sub-cases:

1. $x \in C_2$. In this case the chain ρ is blocked at x.
2. $x \notin C_2$. In this case K m-separates x from v. By Lemma 5, we can obtain that the sub-chain ρ' from x to v can be blocked by $W \cap K$ where W denotes the set of all vertices between x and v (not containing x and v) on ρ'. Since $S \supseteq (W \cap K)$, we obtain from Lemma 5 that S also blocks ρ'. Hence the chain ρ is blocked by S.

ii: $u \rightarrow x$. We have the following sub-cases:

1. $x \in an(u)$. This case is impossible because a directed cycle would occur.
2. $x \in an(v)$. This case is impossible because v cannot be a descendant of u.

iii: $u \leftrightarrow x$. We have the following sub-cases:

1. $x \in an(u)$. This case is impossible because a partially directed cycle would occur.
2. $x \in an(v)$ and v is in the same chain component τ that contains u, x. This is impossible, because in this case we have a partially directed cycle.
3. $x \in an(v)$ and v is not in the same chain component τ that contains u, x. We have the following sub-cases:
   - $x \notin C_2$. In this case K m-separates x from v. By Lemma 5, we can obtain that the sub-chain ρ' from x to v can be blocked by $W \cap K$ where W denotes the set of all vertices between x and v (not containing x and v) on ρ'. Since $S \supseteq (W \cap K)$, we obtain from Lemma 5 that S also blocks ρ'. Hence the chain ρ is blocked by S.

- $x \in C_2$. We have the three following sub-cases:
  - $u \leftrightarrow x \rightarrow y$. In this case $x \in S$ blocks the chain. Note that in this case it is possible that $y = v$.
  - $u \leftrightarrow x \leftarrow y$. So, y ($\neq v$ o.w., a directed cycle would occur) is not a collider. If $y \in C_2$ then the chain ρ is blocked at y. Otherwise, we have the two following sub-cases:
    - There is a node C' between C_1 and C_2 that contains y (note that it is possible that $C' = C_1$), so K m-separates y from v and the same argument used for case i.2 holds.
    - In this case K m-separates y from p ($p \in bd(u) \cap C_1$ and $p \notin C_2$), which is impossible because the chain $p \circ\!\!\rightarrow u \leftrightarrow x \leftarrow y$ is active (note that $u, x \in K$).
  - $u \leftrightarrow x \leftrightarrow y$. If there is an outgoing ($\rightarrow$) edge from y ($\neq v$ o.w., a partially directed cycle would occur) then the same argument in the previous sub-case ($u \leftrightarrow x \leftarrow y$) holds. Otherwise, y is a collider. If $y \notin C_2$ then the same argument in the previous sub-case ($u \leftrightarrow x \leftarrow y$) holds. If $y \in C_2$, there must be a non-collider vertex on the chain ρ between y and v to prevent a (partially) directed cycle. The same argument as in the previous sub-case ($u \leftrightarrow x \leftarrow y$) holds.

■

Proof [Proof of Theorem 2] ($\Rightarrow$) If condition (i) is the case, nothing remains to prove. Otherwise, Lemma 9 implies condition (ii).
($\Leftarrow$) Assume that u and v are not contained together in any node C of T. Also, assume that C_1 and C_2 are two nodes of T that contain u and v, respectively. Consider that C_1' is the most distant node from C_1, between C_1 and C_2, that contains u and C_2' is the most distant node from C_2, between C_1 and C_2, that contains v. Note that it is possible that $C_1' = C_1$ or $C_2' = C_2$. By the condition (i) we know that $C_1' \neq C_2'$. Any separator between C_1' and C_2' satisfies the assumptions of Lemma 5. The sufficiency of condition (i) is given by Lemma 5.

The sufficiency of conditions (ii) is trivial by the definition of m-separation. ■

Proof [Proof of Theorem 3] From (Cowell et al., 1999), we know that any separator S in junction tree T separates $V_1 \setminus S$ and $V_2 \setminus S$ in the triangulated graph $\bar{G}_V^t$, where V_i denotes the variable set of the subtree T_i induced by removing the edge with a separator S attached, for $i = 1, 2$. Since the edge set of $\bar{G}_V^t$ contains that of undirected independence graph $\bar{G}_V$ for G, $V_1 \setminus S$ and $V_2 \setminus S$ are also separated in $\bar{G}_V$. Since $\bar{G}_V$ is an undirected independence graph for G, using Definition 1 we obtain that T is a m-separation tree for G. ■

Appendix B. Proofs for Correctness of the Algorithms

Proof [Correctness of Algorithm 1] By the sufficiency of Theorem 2, the initializations at steps 2 and 3 for creating edges guarantee that no edge is created between any two variables which are not in the same node of the m-separation tree. Also, by the sufficiency of Theorem 2, deleting edges at steps 2 and 3 guarantees that any other edge between two m-separated variables can be deleted in some local skeleton. Thus the global skeleton obtained at step 3 is correct. In a maximal ancestral graph, every missing edge corresponds to at least one independence in the corresponding independence model (Richardson and Spirtes, 2002), and MVR CGs are a subclass of maximal ancestral graphs (Javidian and Valtorta, 2018a). Therefore, according to the necessity of Theorem 2, each augmented edge (u, v) in the undirected independence graph must be deleted at some subgraph over a node of the m-separation tree. Furthermore, according to Lemma 4, for every v-structure $(u \circ\!\!\rightarrow w \leftarrow\!\!\circ v)$ there is a node in m-separation tree T that contains u, v and w, and obviously $w \notin S_{uv}$. Therefore, we can determine all v-structures at step 4, which completes our proof. ■

Proof [Correctness of Algorithm 2] Since an augmented graph for a MVR CG G is an undirected independence graph, by definition of an undirected independence graph, we only need to show that $\bar{G}_V$ defined in step 3 contains all edges of $(G_V)^a$. It is obvious that $\bar{E}$ contains all edges obtained by dropping directions of directed edges in G since any set cannot m-separate two vertices that are adjacent in G.

Now we show that $\bar{E}$ also contains any augmented edge that connects vertices u and v having a collider chain between them, that is, $(u, v) \in \bar{E}$. Any chain graph yields a directed acyclic graph D of its chain components having $\mathcal{T}$ as a node set and an edge $T_1 \rightarrow T_2$ whenever there exists in the chain graph G at least one edge $u \rightarrow v$ connecting a node u in T_1 with a node v in T_2 (Marchetti and Lupparelli, 2011). So, there is a collider chain between two nodes u and v if and only if there is a chain component $\tau \in \mathcal{T}$ such that

- $u, v \in \tau$, or
- $u \in \tau$ and $v \in pa_G(\tau)$ or vice versa, or
- $u, v \in pa_G(\tau)$

Since for each connected component τ there is a $C_h \in C$ containing both τ and its parent set $pa_G(\tau)$, in all of above mentioned cases we have an (u, v) edge in step 2. Therefore, $\bar{G}_V$ defined in step 3 contains all edges of $(G_V)^a$. ■

Proof [Correctness of Main Algorithm] The correctness of Algorithm 1 and Algorithm 2 has been proved above. Thus we only need to prove that step 3 is correct for obtaining a m-separation tree. With an argument similar to that used in the proof of correctness of Algorithm 2, we have that the entire undirected independence graph is constructed correctly at the step 3. Then the m-separation tree can be obtained correctly by using an algorithm for constructing a junction tree. ■

Conclusion and Summary

We proposed a decomposition approach for structural learning of MVR CGs, which extend BNs by allowing the implicit representation of latent (hidden) variables, using bidirected edges (each bidirected edge represents a hidden common parent of its endpoints i.e., $u \leftrightarrow v$ represents $u \leftarrow h \rightarrow v$, where h is a latent variable). In this approach, which extends the pioneering work of

(Xie et al., 2006), a problem of learning a large MVR CG is split into problems of learning small subgraphs. Domain or prior knowledge of conditional independencies can be utilized to facilitate the decomposition of structural learning. We theoretically proved the correctness of the proposed algorithms. Both the complexity of the algorithms and the power of conditional independence tests can be improved by decomposing a large graph into small subgraphs.

We believe that our approach is extendable to the structural learning of AMP chain graphs (Andersson et al., 1996). So, the natural continuation of the work presented here would be to develop a learning algorithm via decomposition for AMP chain graphs. Finally, we have a plan to design an R language package that implements our algorithms and compares our approach with the PC-like algorithm (Sonntag and Peña, 2012) in practice.

Acknowledgements

This work has been partially supported by Office of Naval Research grant ONR N00014-17-1-2842. This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), award/contract number 2017-16112300009. The views and conclusions contained therein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes, notwithstanding annotation therein.

References

S. A. Andersson, D. Madigan, and M. D. Perlman. Alternative Markov properties for chain graphs. *Uncertainty in artificial intelligence*, pages 40–48, 1996.

A. Berry, J. Blair, P. Heggernes, and B. Peyton. Maximum cardinality search for computing minimal triangulations of graphs. *Algorithmica*, 39:287–298, 2004.

R. Cowell, A. P. Dawid, S. Lauritzen, and D. J. Spiegelhalter. *Probabilistic networks and expert systems. Statistics for Engineering and Information Science*. Springer-Verlag, 1999.

D. R. Cox and N. Wermuth. Linear dependencies represented by chain graphs. *Statistical Science*, 8(3):204–218, 1993.

D. R. Cox and N. Wermuth. *Multivariate Dependencies-Models, Analysis and Interpretation*. Chapman and Hall, 1996.

M. Frydenberg. The chain graph Markov property. *Scandinavian Journal of Statistics*, 17(4):333–353, 1990.

M. A. Javidian and M. Valtorta. On the properties of MVR chain graphs. *The 9th International Conference on Probabilistic Graphical Models, Prague (under review)*, 2018a.

M. A. Javidian and M. Valtorta. Finding minimal separators in ancestral graphs. *(under review)*, 2018b.

M. A. Javidian and M. Valtorta. Structural learning of multivariate regression chain graphs via decomposition. `https://cse.sc.edu/~mgv/reports/index.html`, 2018c.

S. Lauritzen. *Graphical Models*. Oxford Science Publications, 1996.

S. Lauritzen and N. Wermuth. Graphical models for associations between variables, some of which are qualitative and some quantitative. *The Annals of Statistics*, 17(1):31–57, 1989.

Z. Ma, X. Xie, and Z. Geng. Structural learning of chain graphs via decomposition. *Journal of Machine Learning Research*, 9:2847–2880, 2008.

G. Marchetti and M. Lupparelli. Chain graph models of multivariate regression type for categorical data. *Bernoulli*, 17(3):827–844, 2011.

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

J. Pearl. *Causality. Models, reasoning, and inference*. Cambridge University Press, 2009.

J. M. Peña. Learning marginal AMP chain graphs under faithfulness. *European Workshop on Probabilistic Graphical Models PGM: Probabilistic Graphical Models*, pages 382–395, 2014.

J. M. Peña. Learning acyclic directed mixed graphs from observations and interventions. *JMLR: Workshop and Conference Proceedings*, 52:392–402, 2016.

J. M. Peña, D. Sonntag, and J. Nielsen. An inclusion optimal algorithm for chain graph structure learning. *In Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, pages 778–786, 2014.

T. S. Richardson. Markov properties for acyclic directed mixed graphs. *Scandinavian Journal of Statistics*, 30(1):145–157, 2003.

T. S. Richardson. A factorization criterion for acyclic directed mixed graphs. *In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 462–470, 2009.

T. S. Richardson and P. Spirtes. Ancestral graph Markov models. *The Annals of Statistics*, 30(4): 962–1030, 2002.

D. Sonntag. *Chain Graphs: Interpretations, Expressiveness and Learning Algorithms) [http://liu.diva-portal.org/smash/get/diva2:910177/FULLTEXT01.pdf]*. Linköping University, 2016.

D. Sonntag and J. M. Peña. Learning multivariate regression chain graphs under faithfulness. *In: Proceedings of the 6th European Workshop on Probabilistic Graphical Models*, pages 299–306, 2012.

D. Sonntag, M. Jãrvisalo, J. M. Peña, and A. Hyttinen. Learning optimal chain graphs with answer set programming. *In Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, pages 822–831, 2015.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search, second ed.* MIT Press, Cambridge, MA., 2000.

M. Studený. A recovery algorithm for chain graphs. *International Journal of Approximate Reasoning*, 17:265–293, 1997.

N. Wermuth and K. Sadeghi. Sequences of regressions and their independences. *Test*, 21:215–252, 2012.

X. Xie, Z. Zheng, and Q. Zhao. Decomposition of structural learning about directed acyclic graphs. *Artificial Intelligence*, 170(4-5):422–439, 2006.