Journal of Machine Learning Research 9 (2008) 2523-2547 Submitted 6/07; Revised 2/08; Published 11/08

Active Learning of Causal Networks with Intervention Experiments and Optimal Designs

Yang-Bo He HEYB@MATH.PKU.EDU.CN
Zhi Geng ZGENG@MATH.PKU.EDU.CN
School of Mathematical Sciences, LMAM
Peking University
Beijing 100871, China

Editor: Andre Elisseeff

Abstract

The causal discovery from data is important for various scientific investigations. Because we cannot distinguish the different directed acyclic graphs (DAGs) in a Markov equivalence class learned from observational data, we have to collect further information on causal structures from experiments with external interventions. In this paper, we propose an active learning approach for discovering causal structures in which we first find a Markov equivalence class from observational data, and then we orient undirected edges in every chain component via intervention experiments separately. In the experiments, some variables are manipulated through external interventions. We discuss two kinds of intervention experiments, randomized experiment and quasi-experiment. Furthermore, we give two optimal designs of experiments, a batch-intervention design and a sequential-intervention design, to minimize the number of manipulated variables and the set of candidate structures based on the minimax and the maximum entropy criteria. We show theoretically that structural learning can be done locally in subgraphs of chain components without need of checking illegal v-structures and cycles in the whole network and that a Markov equivalence subclass obtained after each intervention can still be depicted as a chain graph.

Keywords: active learning, causal networks, directed acyclic graphs, intervention, Markov equivalence class, optimal design, structural learning

1. Introduction

A directed acyclic graph (DAG) (also called a Bayesian network) is a powerful tool to describe a large complex system in various scientific investigations, such as bioinformatics, epidemiology, sociology and business (Pearl, 1988; Lauritzen, 1996; Whittaker, 1990; Aliferis et al., 2003; Jansen et al., 2003; Friedman, 2004). A DAG is also used to describe causal relationships among variables. It is crucial to discover the structure of a DAG for understanding a large complex system or for doing uncertainty inference on it (Cooper and Yoo, 1999; Pearl, 2000). There are many methods of structural learning, and the main methods are Bayesian methods (Cooper and Yoo, 1999; Heckerman, 1997) and constraint-based methods (Spirtes et al., 2000). From data obtained in observational studies, we may not have enough information to discover causal structures completely, but we can obtain only a Markov equivalence class. Thus we have to collect further information of causal structures via experiments with external interventions. Heckerman et al. (1995) discussed structural learning of Bayesian networks from a combination of prior knowledge and statistical data. Cooper and Yoo (1999) presented a method of causal discovery from a mixture of experimental and obser-

vational data. Tian and Pearl (2001a,b) proposed a method of discovering causal structures based on dynamic environment. Tong and Koller (2001) and Murphy (2001) discussed active learning of Bayesian network structures with posterior distributions of structures based on decision theory. In these methods, causal structures are discovered by using additional information from domain experts or experimental data.

Chain graphs were introduced as a natural generalization of DAGs to admit more flexible causal interpretation (Lauritzen and Richardson, 2002). A chain graph contains both directed and undirected edges. A chain component of a chain graph is a connected undirected graph obtained by removing all directed edges from the chain graph. Andersson et al. (1997) showed that DAGs in a Markov equivalence class can be represented by a chain graph. He et al. (2005) presented an approach of structural learning in which a Markov equivalence class of DAGs is sequentially refined into some smaller subclasses via domain knowledge and randomized experiments.

In this paper, we discuss randomized experiments and quasi-experiments of external interventions. We propose a method of local orientations in every chain component, and we show theoretically that the method of local orientations does not create any new v-structure or cycle in the whole DAG provided that neither v-structure nor cycle is created in any chain component. Thus structural learning can be done locally in every chain component without need of checking illegal v-structures and cycles in the whole network. Then we propose the optimal designs of interventional experiments based on the minimax and maximum entropy criteria. These results greatly extend the approach proposed by He et al. (2005). In active learning, we first find a Markov equivalence class from observational data, which can be represented by a chain graph, and then we orient undirected edges via intervention experiments. Two kinds of intervention experiments can be used for orientations. One is randomized experiment, in which an individual is randomly assigned to some level combination of the manipulated variables at a given probability. Randomization can disconnect the manipulated variables from their parent variables in the DAG. Although randomized experiments are most powerful for learning causality, they may be inhibitive in practice. The other is quasi-experiment, in which the pre-intervention distributions of some variables are changed via external interventions, but we cannot ensure that the manipulated variables can be disconnected from their parent variables in the DAG, and thus the post-intervention distributions of manipulated variables may still depend on their parent variables. For example, the pre-intervention distribution of whether patients take a vaccine or not may depend on some variables, and the distribution may be changed by encouraging patients with some benefit in the quasi-experiment, but it may still depend on these variables. Furthermore, we discuss the optimal designs by which the number of manipulated variables is minimized or the uncertainty of candidate structures is minimized at each experiment step based on the minimax and the maximum entropy criteria. We propose two kinds of optimal designs: a batch-intervention experiment and a sequential intervention experiment. For the former, we try to find the minimum set of variables to be manipulated in a batch such that undirected edges are all oriented after the interventions. For the latter, we first choose a variable to be manipulated such that the Markov equivalence class can be reduced by manipulating the variable into a subclass as small as possible, and then according to the current subclass, we repeatedly choose a next variable to be manipulated until all undirected edges are oriented.

In Section 2, we introduce notation and definitions and then show some theoretical results on Markov equivalence classes. In Section 3, we present active learning of causal structures via external interventions and discuss randomized experiments and quasi-experiments. In Section 4, we propose two optimal designs of intervention experiments, a batch-intervention design and a sequen-

tial intervention design. In Section 5, we show simulation results to evaluate the performances of intervention designs proposed in this paper. Conclusions are given in Section 6. Proofs of theorems are given in Appendix.

2. Causal DAGs and Markov Equivalence Class

A graph G can be defined to be a pair $G = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V} = \{V_1, \cdots, V_n\}$ denotes the node set and $\mathbb{E}$ denotes the edge set which is a subset of the set $\mathbb{V} \times \mathbb{V}$ of ordered pairs of nodes. If both ordered pairs (V_i, V_j) and (V_j, V_i) are in $\mathbb{E}$, we say that there is an undirected edge between V_i and V_j, denoted as $V_i - V_j$. If $(V_i, V_j) \in \mathbb{E}$ and $(V_j, V_i) \notin \mathbb{E}$, we call it a directed edge, denoted as $V_i \to V_j$. We say that V_i is a neighbor of V_j if there is an undirected or directed edge between V_i and V_j. A graph is directed if all edges of the graph are directed. A graph is undirected if all edges of the graph are undirected.

A sequence $(V_1, V_2, \cdots, V_k)$ is called a *partially directed path* from V_1 to V_k if either $V_i \to V_{i+1}$ or $V_i - V_{i+1}$ is in G for all $i = 1, \ldots, k-1$. A partially directed path is a directed path if there is not any undirected edge in the path. A node V_i is an *ancestor* of V_j and V_j is a *descendant* of V_i if there is a directed path from V_i to V_j. A *directed cycle* is a directed path from a node to itself, and a *partially directed cycle* is a partially directed path from a node to itself.

A graph with both directed and undirected edges is a chain graph if there is not any partially directed cycle. Figure 1 shows a chain graph with five nodes. A chain component is a node set whose nodes are connected in an undirected graph obtained by removing all directed edges from the chain graph. An undirected graph is chordal if every cycle of length larger than or equal to 4 possesses a chord.

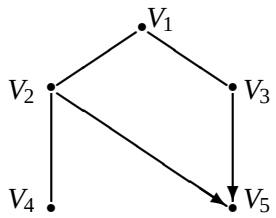

Figure 1: A chain graph G^* depicts the essential graph of G, G_1, G_2 and G_3.

A directed acyclic graph (DAG) is a directed graph which does not contain any directed cycle. A causal DAG is a DAG which is used to describe the causal relationships among variables $V_1, \cdots, V_n$. In the causal DAG, a directed edge $V_i \to V_j$ is interpreted as that the *parent* node V_i is a cause of the *child* node V_j, and that V_j is an effect of V_i. Let $pa(V_i)$ denote the set of all parents of V_i and $ch(V_i)$ denote the set of all *children* of V_i. Let τ be a node subset of $\mathbb{V}$. The *subgraph* $G_\tau = (\tau, \mathbb{E}_\tau)$ induced by the subset τ has the node set τ and the edge set $\mathbb{E}_\tau = \mathbb{E} \cap (\tau \times \tau)$ which contains all edges falling into τ. Two graphs have *the same skeleton* if they have the same set of nodes and the same set of edges regardless of their directions. A head-to-head structure is called a *v-structure* if the parents are not adjacent, such as $V_1 \to V_2 \leftarrow V_3$.

Figure 2 shows four different causal structures of five nodes. The causal graph G in Figure 2 depicts that V_1 is a cause of V_3, which in turn is a cause of V_5.

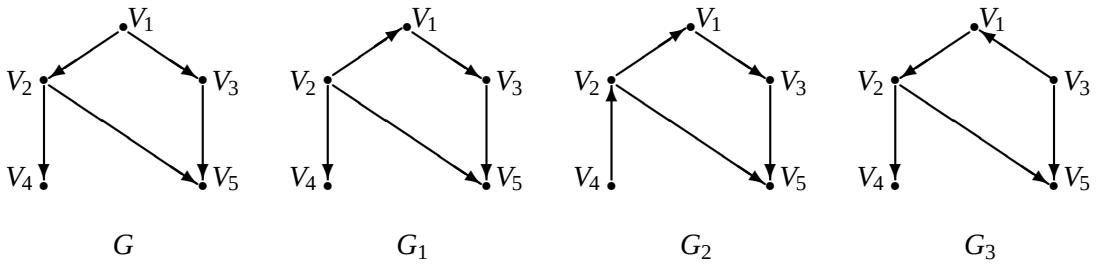

Figure 2: The equivalence class $[G]$.

A joint distribution P satisfies Markov property with respect to a graph G if any variable of G is independent of all its non-descendants in G given its parents with respect to the joint distribution P. Furthermore, the distribution P can be factored as follows

$$P(v_1, v_2, \cdots, v_n) = \prod_{i=1}^{n} P(v_i | pa(v_i)),$$

where v_i denotes a value of variable V_i, and $pa(v_i)$ denotes a value of the parent set $pa(V_i)$ (Pearl, 1988; Lauritzen, 1996; Spirtes et al., 2000). In this paper, we assume that any conditional independence relations in P are entailed by the Markov property, which is called the faithfulness assumption (Spirtes et al., 2000). We also assume that there are no latent variables (that is, no unmeasured variables) in causal DAGs. Different DAGs may encode the same Markov properties. A Markov equivalence class is a set of DAGs that have the same Markov properties. Let $G_1 \sim G_2$ denote that two DAGs G_1 and G_2 are Markov equivalent, and let $[G]$ denote the equivalence class of a DAG G, that is, $[G] = \{G' : G' \sim G\}$. The four DAGs G, G_1, G_2 and G_3 in Figure 2 form a Markov equivalence class $[G]$. Below we review two results about Markov equivalence of DAGs given by Verma and Pearl (1990) and Andersson et al. (1997).

Lemma 1 *(Verma and Pearl, 1990) Two DAGs are Markov equivalent if and only if they have the same skeleton and the same v-structures.*

Andersson et al. (1997) used an essential graph G^* to represent the equivalence class $[G]$.

Definition 2 *The essential graph $G^* = (\mathbb{V}, \mathbb{E}^*)$ of G has the same node set and the same skeleton as G, whose one edge is directed if and only if it has the same orientation in every DAG in $[G]$ and whose other edges are undirected.*

For example, G^* in Figure 1 is the essential graph of G in Figure 2. The edges $V_2 \rightarrow V_5$ and $V_3 \rightarrow V_5$ in G^* are directed since they have the same orientation for all DAGs of $[G]$ in Figure 2, and other edges are undirected.

Lemma 3 *(Andersson et al., 1997) Let G^* be the essential graph of $G = (\mathbb{V}, \mathbb{E})$. Then G^* has the following properties:*

(i) *G^* is a chain graph,*

(ii) *G^*_τ is chordal for every chain component τ, and*

(iii) *$V_i \to V_j - V_k$ does not occur as an induced subgraph of G^*.*

Suppose that G is an unknown underlying causal graph and that its essential graph $G^* = (\mathbb{V}, \mathbb{E})$ has been obtained from observational data, and has k chain components $\{\tau_1, \cdots, \tau_k\}$. Its edge set $\mathbb{E}$ can be partitioned into the set $\mathbb{E}_1$ of directed edges and the set $\mathbb{E}_2$ of undirected edges. Let G^*_τ denote a subgraph of the essential G^* induced by a chain component τ of G^*. Any subgraph of the essential graph induced by a chain component is undirected. Since all v-structures can be discovered from observational data, any subgraph G'_τ of G' should not have any v-structure for $G' \in [G]$. For example, the essential graph G^* in Figure 1 has one chain component $\tau = \{V_1, V_2, V_3, V_4\}$. It can been seen that G'_τ has no v-structure for $G' \in \{G, G_1, G_2, G_3\}$.

Given an essential graph G^*, we need to orient all undirected edges in each chain component to discover the whole causal graph G. Below we show that the orientation can be done separately in every chain component. We also show that there are neither new v-structures nor cycles in the whole graph as long as there are neither v-structures nor cycles in any chain component. Thus in the orientation process, we only need to ensure neither v-structures nor cycles in any component, and we need not check new v-structures and cycles for the whole graph.

Theorem 4 *Let τ be a chain component of an essential graph G^*. For each undirected edge $V - U$ in G^*_τ, neither orientation $V \to U$ nor $V \leftarrow U$ can create a v-structure with any node W outside τ, that is, neither $V \to U \leftarrow W$ nor $W \to V \leftarrow U$ can occur for any $W \notin \tau$.*

Theorem 4 means that there is not any node W outside the component τ which can build a v-structure with two nodes in τ.

Theorem 5 *Let τ be a chain component of G^*. If orientation of undirected edges in the subgraph G^*_τ does not create any directed cycle in the subgraph, then the orientation does not create any directed cycle in the whole DAG.*

According to Theorems 4 and 5, we find that the undirected edges can be oriented separately in each chain component regardless of directed and undirected edges in other part of the essential graph as long as neither cycles nor v-structures are constructed in any chain component. Thus the orientation for one chain component does not affect the orientations for other components. The orientation approach and its correctness will be discussed in Section 3.

3. Active Learning of Causal Structures via External Interventions

To discover causal structures further from a Markov equivalence class obtained from observational data, we have to perform external interventions on some variables. In this section, we consider two kinds of external interventions. One is the randomized experiment, in which the post-intervention distribution of the manipulated variable V_i is independent of its parent variables. The other is the quasi-experiment, in which the distribution of the manipulated variable V_i conditional on its parents $pa(V_i)$ is changed by manipulating V_i. For example, the distribution of whether patients take a vaccine or not is changed by randomly encouraging patients at a discount.

3.1 Interventions by Randomized Experiments

In this subsection, we conduct interventions as randomized experiments, in which some variables are manipulated from external interventions by assigning individuals to some levels of these variables in a probabilistic way. For example, in a clinical trial, every patient is randomly assigned to a treatment group of $V_i = v_i$ at a probability $P'(v_i)$. The randomized manipulation disconnects the node V_i from its parents $pa(V_i)$ in the DAG. Thus the pre-intervention conditional probability $P(v_i|pa(v_i))$ of $V_i = v_i$ given $pa(V_i) = pa(v_i)$ is replaced by the post-intervention probability $P'(v_i)$ while all other conditional probabilities $P(v_j|pa(v_j))$ for $j \neq i$ are kept unchanged in the randomized experiment. Then the post-intervention joint distribution is

$$P_{V_i}(v_1, v_2, \cdots, v_n) = P'(v_i) \prod_{j \neq i} P(v_j|pa(v_j)),$$

(Pearl, 1993). From this post-intervention distribution, we have $P_{V_i}(v_i|pa(v_i)) = P_{V_i}(v_i)$, that is, the manipulated variable V_i is independent of its parents $pa(V_i)$ in the post-intervention distribution. Under the faithfulness assumption, it is obvious that an undirected edge between V_i and its neighbor V_j can be oriented as $V_i \leftarrow V_j$ if the post-intervention distribution has $V_i \perp\!\!\!\perp V_j$, otherwise it is oriented as $V_i \rightarrow V_j$, where $V_i \perp\!\!\!\perp V_j$ denotes that V_i is independent of V_j. The orientation only needs an independence test for the marginal distribution of variables V_i and V_j. Notice that the independence is tested by using only the experimental data without use of the previous observational data.

Let $e(V_i)$ denote the orientation of edges which is determined by manipulating node V_i. If V_i belongs to a chain component τ (that is, it connects at least one undirected edge), then the Markov equivalence class $[G]$ can be reduced by manipulating V_i to the post-intervention Markov equivalence class $[G]_{e(V_i)}$

$$[G]_{e(V_i)} = \{G' \in [G] | G' \text{ has the same orientation as } e(V_i)\}.$$

A Markov equivalence class is split into several subclasses by manipulating V_i, each of which has different orientations $e(V_i)$. Let $G^*_{e(V_i)}$ denote the post-intervention essential graph which depicts the post-intervention Markov equivalence class $[G]_{e(V_i)}$. We show below that $G^*_{e(V_i)}$ also has the properties of essential graphs.

Theorem 6 *Let τ be a chain component of the pre-intervention essential graph G^* and V_i be a node in the component τ. The post-intervention graph $G^*_{e(V_i)}$ is also a chain graph, that is, $G^*_{e(V_i)}$ has the following properties:*

(i) *$G^*_{e(V_i)}$ is a chain graph,*

(ii) *$G^*_{e(V_i)}$ is chordal, and*

(iii) *$V_j \rightarrow V_k - V_l$ does not occur as an induced subgraph of $G^*_{e(V_i)}$.*

By Theorem 6, the pre-intervention chain graph is changed by manipulating a variable to another chain graph which has less undirected edges. Thus variables in chain components can be manipulated repeatedly until the Markov equivalence subclass is reduced to a subclass with a single DAG, and properties of chain graphs are not lost in this intervention process.

According to the above results, we first learn an essential graph from observational data, which is a chain graph (Andersson et al., 1997) and depicts a Markov equivalence class (Heckerman et

al., 1995; Verma and Pearl, 1990; Castelo and Perlman, 2002). Next we choose a variable V_i to be manipulated from a chain component, and we can orient the undirected edges connecting V_i and some other undirected edges whose reverse orientations create v-structures or cycles. Repeating this process, we choose a next variable to be manipulated until all undirected edges are oriented. Below we give an example to illustrate the intervention process.

Example 1. Consider an essential graph in Figure 3, which depicts a Markov equivalence class with 12 DAGs in Figure 4. After obtaining the essential graph from observational data, we manipulate some variables in randomized experiments to identify a causal structure in the 12 DAGs. For example, Table 1 gives four possible orientations and Markov equivalence subclasses obtained by manipulating V_1. A class with 12 DAGs is split into four subclasses by manipulating V_1. The post-intervention subclasses (ii) and (iv) have only a single DAG separately. Notice that undirected edges not connecting V_1 can also be oriented by manipulating V_1. The subclasses (i) and (iii) are depicted by post-intervention essential graphs (a) and (b) in Table 1 respectively, both of which are chain graphs. In Table 2, the first column gives four possible independence sets obtained by manipulating V_1. For the set with $V_1 \perp\!\!\!\perp V_2$ and $V_1 \not\perp\!\!\!\perp V_3$, the causal structure is the DAG (3) in Figure 4, and thus we need not further manipulate other variables. For the third set with $V_1 \not\perp\!\!\!\perp V_2$ and $V_1 \not\perp\!\!\!\perp V_3$, we manipulate the next variable V_2. If $V_2 \perp\!\!\!\perp V_3$, then the causal structure is the DAG (1), otherwise it is the DAG (2). For the fourth set with $V_1 \perp\!\!\!\perp V_2$ and $V_1 \perp\!\!\!\perp V_3$, we may need further to manipulate variables V_2, V_3 and V_4 to identify a causal DAG.

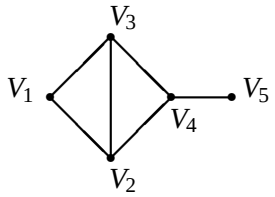

Figure 3: An essential graph of DAGs

3.2 Interventions by Quasi-experiments

In the previous subsection we discussed interventions by randomized experiments. Although randomized experiments are powerful tools to discover causal structures, it may be inhibitive or impractical. In this subsection we consider quasi-experiments. In a quasi-experiment, individuals may choose treatments non-randomly, but their behaviors of treatment choices are influenced by experimenters. For example, some patients may not comply with the treatment assignment from a doctor, but some of them may comply, which is also called an indirect experiment in Pearl (1995).

If we perform an external intervention on V_i such that V_i has a conditional distribution $P'(v_i|pa(v_i))$ different from the pre-intervention distribution $P(v_i|pa(v_i))$ in (1) and other distributions are kept unchanged, then we have the post-intervention joint distribution

$$P_{V_i}(v_1, v_2, \cdots, v_n) = P'(v_i|pa(v_i)) \prod_{j \neq i} P(v_j|pa(v_j)).$$

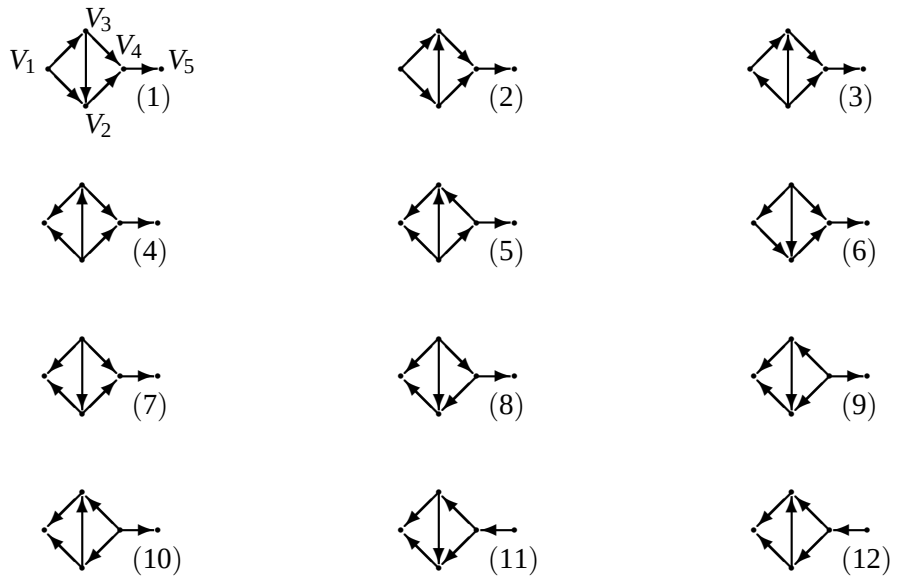

Figure 4: All DAGs in the equivalence class given in Figure 3.

No of subclass	$e(V_1)$	DAGs in a subclass	post-intervention essential graphs
(i)	$V_2 \leftarrow V_1 \rightarrow V_3$	$(1,2)$	V_1 (a)
(ii)	$V_2 \rightarrow V_1 \rightarrow V_3$	(3)	
(iii)	$V_2 \rightarrow V_1 \leftarrow V_3$	$(4,5,$ $7-12)$	V_1 (b)
(iv)	$V_2 \leftarrow V_1 \leftarrow V_3$	(6)	

Table 1: The post-intervention subclasses and essential graphs obtained by manipulating V_1.

In the external intervention, we may not be able to manipulate V_i, but we only need to change its conditional distribution, which may still depend on its parent variables. We call such an experiment a quasi-experiment. Below we discuss how to orient undirected edges via such quasi-experiments. Let τ be a chain component of the essential graph G^*, $ne(V_k)$ be the neighbor set of V_k, C be the children of V_k outside τ (that is, $C = ch(V_k) \setminus \tau$), and B be the set of all potential parents of V_k, that is, $B = ne(V_k) \setminus C$ is the neighbor set of V_k minus the children of V_k which have been identified in the chain graph. Let $V_i - V_k$ be an undirected edge in a chain component τ, and we want to orient the undirected edge by manipulating V_i. Since B is the neighbor set of V_k, we have $V_i \in B$ and thus

V_1	V_2	V_3	V_4	*DAG* in Fig. 4
$V_1 \perp\!\!\!\perp V_2$ *and* $V_1 \not\perp\!\!\!\perp V_3$	*	*	*	(3)
$V_1 \not\perp\!\!\!\perp V_2$ *and* $V_1 \perp\!\!\!\perp V_3$	*	*	*	(6)
$V_1 \not\perp\!\!\!\perp V_2$ *and* $V_1 \not\perp\!\!\!\perp V_3$	$V_2 \perp\!\!\!\perp V_3$	*	*	(1)
	$V_2 \not\perp\!\!\!\perp V_3$	*	*	(2)
$V_1 \perp\!\!\!\perp V_2$ *and* $V_1 \perp\!\!\!\perp V_3$	$V_2 \perp\!\!\!\perp V_3$ *and* $V_2 \not\perp\!\!\!\perp V_4$	*	*	(7)
	$V_2 \not\perp\!\!\!\perp V_3$ *and* $V_2 \not\perp\!\!\!\perp V_4$	$V_3 \not\perp\!\!\!\perp V_4$	*	(4)
		$V_3 \perp\!\!\!\perp V_4$	*	(5)
	$V_2 \perp\!\!\!\perp V_3$ *and* $V_2 \perp\!\!\!\perp V_4$	$V_3 \not\perp\!\!\!\perp V_4$	*	(8)
		$V_3 \perp\!\!\!\perp V_4$	$V_4 \not\perp\!\!\!\perp V_5$	(9)
			$V_4 \perp\!\!\!\perp V_5$	(11)
	$V_2 \not\perp\!\!\!\perp V_3$ *and* $V_2 \perp\!\!\!\perp V_4$	*	$V_4 \not\perp\!\!\!\perp V_5$	(10)
			$V_4 \perp\!\!\!\perp V_5$	(12)

Table 2: The intervention process to identify a causal structure from the essential graph in Figure 3, where $*$ means that the intervention is unnecessary.

$B \neq \emptyset$. Below we show a result which can be used to identify the direction of the undirected edge $V_i - V_k$ via a quasi-experiment of intervention on V_i.

Theorem 7 *For a quasi-experiment of intervention on V_i, we have the following properties*

1. $P_{V_i}(v_k|B) = P(v_k|B)$ *for all v_k and B if V_i is a parent of V_k, and*
2. $P_{V_i}(v_k) = P(v_k)$ *for all v_k if V_i is a child of V_k.*

According to Theorem 7, we can orient the undirected edge $V_i - V_k$ as

1. $V_i \leftarrow V_k$ if $P_{V_i}(v_k|B) \neq P(v_k|B)$ for some v_k and B, or
2. $V_i \rightarrow V_k$ if $P_{V_i}(v_k) \neq P(v_k)$ for some v_k.

The nonequivalence of pre- and post-intervention distributions is tested by using both experimental data and observational data, which is different from that of randomized experiments.

Example 1 (continued). Consider again the essential graph in Figure 3. We use a quasi-experiment of manipulating V_1 in order to orient the undirected edges connecting V_1 ($V_3 - V_1 - V_2$). We may test separately four null hypotheses $P_{V_1}(v_2) = P(v_2)$, $P_{V_1}(v_3) = P(v_3)$, $P_{V_1}(v_2|v_1, v_3, v_4) = P(v_2|v_1, v_3, v_4)$ and $P_{V_1}(v_3|v_1, v_2, v_4) = P(v_3|v_1, v_2, v_4)$ with both observational and experimental data. We orient $V_1 - V_2$ as $V_1 \rightarrow V_2$ if $P_{V_1}(v_2) \neq P(v_2)$, otherwise as $V_1 \leftarrow V_2$ (or further check whether there is a stronger evidence of $P_{V_1}(v_2|v_1, v_3, v_4) \neq P(v_2|v_1, v_3, v_4)$). Similarly we can orient $V_1 - V_3$. Finally we obtain four possible orientations as shown in Table 1.

If both $P_{V_i}(v_k) = P(v_k)$ and $P_{V_i}(v_k|B) = P(v_k|B)$ for all v_k and B hold for a quasi-experiment, then we cannot identify the direction of edge $V_i - V_k$ from the intervention. For example, suppose that there are only two variables V_1 and V_2, V_1 has three levels and V_1 is the parent of V_2. If the true conditional distribution of V_2 given V_1 is: $p(v_2|V_1 = 1) = p(v_2|V_1 = 2) \neq p(v_2|V_1 = 3)$, then the

undirected edge $V_1 - V_2$ cannot be oriented with the intervention on V_1 with $p_{V_1}(V_1 = v) \neq p(V_1 = v)$ for $v = 1$ and 2 but $p_{V_1}(V_1 = 3) = p(V_1 = 3)$ because we have that $p_{V_1}(v_2) = p(v_2)$ for all v_2 and that $p_{V_1}(v_2|B) = p(v_2|B)$ where $B = \{V_1\}$. In a quasi-experiment, an experimenter may not be able to manipulate V_1, and thus this phenomenon can occur. If V_1 can be manipulated, then the experimenter can choose the distribution of V_2 to avoid this phenomenon.

4. Optimal Designs of Intervention Experiments

In this section, we discuss the optimal designs of intervention experiments which are used to minimize the number of manipulated variables or to minimize the uncertainty of candidate structures after an intervention experiment based on some criteria. Since the orientation for one chain component is unrelated to the orientations for other components, we can design an intervention experiment for each chain component separately. As shown in Section 2, given a chain component τ, we orient the subgraph over τ into a DAG G_τ without any v-structure or cycle via experiments of interventions in variables in τ. For simplicity, we omit the subscript τ in this section. In the following subsections, we discuss intervention designs for only one chain component. We first introduce the concept of sufficient interventions and discuss their properties of sufficient interventions, then we present the optimal design of batch interventions, and finally we give the optimal design of sequential interventions. For optimizing quasi-experiments of interventions, we assume that intervention on a variable V_i will change the marginal distribution of its child V_j, that is, there is a level v_j such that $P_{V_i}(v_j) \neq P(v_j)$ for $V_i \to V_j$. Under this assumption, all undirected edges connecting a node V_i can be oriented via a quasi-experiment of intervention on variable V_i. Without the assumption, there may be some undirected edge which cannot be oriented even if we perform interventions in both of its two nodes.

4.1 Sufficient Interventions

It is obvious that we can identify a DAG in a Markov equivalence class if we can manipulate all variables which connect undirected edges. However, it may be unnecessary to manipulate all of these variables. Let $\mathcal{S} = (V_1, V_2, \cdots, V_k)$ denote a sequence of manipulated variables. We say that a sequence of manipulated variables is sufficient for a Markov equivalence class $[G]$ if we can identify one DAG from all possible DAGs in $[G]$ after these variables in the sequence are manipulated. That is, we can orient all undirected edges of the essential graph G^* no matter which G in $[G]$ is the true DAG. There may be several sufficient sequences for a Markov equivalence class $[G]$.

Let g denote the number of nodes in the chain component, and h the number of undirected edges within the component. Then there are at most 2^h possible orientation of these undirected edges, and thus there are at most 2^h DAGs over the component in the Markov equivalence class. Given a permutation of nodes in the component, a DAG can be obtained by orienting all undirected edges backwards in the direction of the permutation, and thus there are at most $\min\{2^h, g!\}$ DAGs in the class.

Theorem 8 *If a sequence $\mathcal{S} = (V_1, V_2, \cdots, V_k)$ of manipulated variables is sufficient, then any permutation of $\mathcal{S}$ is also sufficient.*

According to Theorem 8, we can ignore the order of variables in an intervention sequence and treat the sequence as a variable set. Thus, if $\mathcal{S}$ is a sufficient set, then $\mathcal{S}'$ which contains $\mathcal{S}$

is also sufficient. Manipulating V_i, we obtain a class $E(V_i) = \{e(V_i)\}$ of orientations (see Table 1 as an example). Given an orientation $e(V_i)$, we can obtain the class $[G]_{e(V_i)}$ by (3). We say that $e(V_1,\ldots,V_k) = \{e(V_1),\ldots,e(V_k)\}$ is a legal combination of orientations if there is not any v-structure or cycle formed and there is not any undirected edge oriented in two different directions by these orientations. For a set $\mathcal{S} = (V_1,\ldots,V_k)$ of manipulated variables, the Markov equivalence class is reduced into a class

$$[G]_{e(V_1,\ldots,V_k)} = [G]_{e(V_1)} \cap \ldots \cap [G]_{e(V_k)}$$

for a legal combination $e(V_1,\ldots,V_k)$ of orientations. If $[G]_{e(V_1,\ldots,V_k)}$ has only one DAG for all possible legal combinations $e(V_1,\ldots,V_k) \in E(V_1) \times \ldots \times E(V_k)$, then the set $\mathcal{S}$ is a sufficient set for identifying any DAG in $[G]$. Let $\mathbb{S}$ denote the class of all sufficient sets, that is, $\mathbb{S} = \{\mathcal{S} : \mathcal{S} \text{ is sufficient}\}$. We say that a sequence $\mathcal{S}$ is minimum if any subset of $\mathcal{S}$ is not sufficient.

Theorem 9 *The intersection of all sufficient sets is an empty set, that is, $\bigcap_{\mathcal{S}\in\mathbb{S}} \mathcal{S} = \varnothing$. In addition, the intersection of all minimum sufficient sets is also an empty set.*

From Theorem 9, we can see that there is not any variable that must be manipulated to identify a causal structure. Especially, any undirected edge can be oriented by manipulating either of its two nodes.

4.2 Optimization for Batch Interventions

We say that an intervention experiment is a batch-intervention experiment if all variables in a sufficient set $\mathcal{S}$ are manipulated in a batch to orient all undirected edges of an essential graph. Let $|\mathcal{S}|$ denote the number of variables in $\mathcal{S}$. We say that a batch intervention design is optimal if its sufficient set $\mathcal{S}_o$ has the smallest number of manipulated variables, that is, $|\mathcal{S}_o| = \min\{|\mathcal{S}| : \mathcal{S} \in \mathbb{S}\}$. Given a Markov equivalence class $[G]$, we try to find a sufficient set S which has the smallest number of manipulated variables for identifying all possible DAGs in the class $[G]$. Below we give an algorithm to find the optimal design for batch interventions, in which we first try all sets with a single manipulated variable, then try all sets with two variables, and so on, until each post-intervention Markov equivalence class has a single DAG.

Given a Markov equivalence class $[G]$, we manipulate a node V and obtain an orientation of some edges, denoted by $e(V)$. The class $[G]$ is split into several subclasses, denoted by $[G]_{e(V)}$ for all possible orientations $e(V)$. Let $[G]_{e(V_1,V_2)}$ denote a subclass with an orientation obtained by manipulating V_1 and V_2. The following algorithm 1 performs exhaustive search for the optimal design of batch interventions. Before calling Algorithm 1, we need to enumerate all DAGs in the class $[G]$, and then we can easily find $[G]_{e(V_i)}$ according to (3). There are at most $\min\{g!, 2^h\}$ DAGs in the class $[G]$, and thus the upper bound of the complexity for enumerating all $\{[G]_{e(V_i)}\}$ is $\min\{g!, 2^h\}$. We may be able to have an efficient method to find all $\{[G]_{e(V_i)}\}$ using the structure of the chain component.

Algorithm 1 Algorithm for finding the optimal designs of batch interventions

Input: A chain graph G induced by a chain component $\tau = \{V_1, \ldots, V_g\}$, and $[G]_{e(V_i)}$ for all $e(V_i)$ and i.

Output: All optimal designs of batch interventions.

```
Initialize the size k of the minimum intervention set as k = 0.
repeat
  Set k = k + 1.
  for all possible variable subsets S = {V_{i_1}, ..., V_{i_k}} do
    if |[G]_{e(S)}| = 1 for all possible legal combination e(S) of orientations then
      return the minimum sufficient set S
    end if
  end for
until find some sufficient sets
```

Algorithm 1 exhaustively searches all combinations of manipulated variables to find the minimum sufficient sets, and its complexity is $O(g!)$, although Algorithm 1 may stop whenever it finds some minimum sets. The calculations in Algorithm 1 are only simple set operations

$$[G]_{e(\mathcal{S})} = [G]_{e(V_{i_1})} \cap \ldots \cap [G]_{e(V_{i_k})},$$

where all $[G]_{e(V_i)}$ have been found before calling Algorithm 1. Notice that a single chain component usually has a size g much less than the total number n of variables. Algorithm 1 is feasible for a mild size g. A more efficient algorithm or a greedy method is needed for a large g and h. In this case, there are too many DAGs to enumerate. We can first take a random sample of DAGs from the class $[G]$ with the simulation method proposed in the next subsection, and then we use the sample approximately to find an optimal design.

A possible greedy approach is to select a node to be first manipulated from the chain component which has the largest number of neighbors such that the largest number of undirected edges are oriented by manipulating it, and then delete these oriented edges. Repeat this process until there is not any undirected edge left. But there are cases where the sufficient set obtained from the greedy method is not minimum.

Example 1 (continued). Consider the essential graph in Figure 3, which depicts a Markov equivalence class with 12 DAGs in Figure 4. From Algorithm 1, we can find that $\{1,2,4\}$, $\{1,3,4\}$, $\{2,3,4\}$ and $\{2,3,5\}$ are all the minimum sufficient sets. The greedy method can obtain the same minimum sufficient sets for this example.

4.3 Optimization for Sequential Interventions

The optimal design of batch interventions presented in the previous subsection tries to find a minimum sufficient set $\mathcal{S}$ before any variable is manipulated, and thus it cannot use orientation results obtained by manipulating the previous variables during the intervention process. In this subsection, we propose an experiment of sequential interventions, in which variables are manipulated sequentially. Let $\mathcal{S}^{(t)}$ denote the set of variables that have been manipulated before step t and $\mathcal{S}^{(0)} = \emptyset$. At step t of the sequential experiment, according to the current Markov equivalence class $[G]_{e(\mathcal{S}^{(t-1)})}$ obtained by manipulating the previous variables in $\mathcal{S}^{(t-1)}$, we choose a variable V to be manipulated

based on some criterion. We consider two criteria for choosing a variable. One is the minimax criterion based on which we choose a variable V such that the maximum size of subclasses $[G]_{e(\mathcal{S}^{(t)})}$ for all possible orientations $e(\mathcal{S}^{(t)})$ is minimized. The other is the maximum entropy criterion based on which we choose a variable V such that the following entropy is maximized for any V in the chain component τ

$$H_V = -\sum_{i=1}^{M} \frac{l_i}{L} \log \frac{l_i}{L},$$

where l_i denotes the number of possible DAGs of the chain component with the ith orientation $e(V)_i$ obtained by manipulating V, $L = \sum_i l_i$ and M is the number of all possible orientations $e(V)_1, \ldots, e(V)_M$ obtained by manipulating V. Based on the maximum entropy criterion, the post-intervention subclasses have sizes as small as possible and they have sizes as equal as possible, which means uncertainty for identifying a causal DAG from the Markov equivalence class is minimized by manipulating V. Below we give two examples to illustrate how to choose variables to be manipulated in the optimal design of sequential interventions based on the two criteria.

Example 1 (continued). Consider again the essential graph in Figure 3, which depicts a Markov equivalence class with 12 DAGs in Figure 4. Tables 3 to 6 show the results for manipulating one of variables V_1, V_2 (symmetry to V_3), V_4 and V_5 respectively in order to distinguish the possible DAGs in Figure 4. The first row in these tables gives possible orientations obtained by manipulating the corresponding variable. The second row gives DAGs obtained by the orientation, where numbers are used to index DAGs in Figure 4. The third row gives the number l_i of DAGs of this chain component for the ith orientation. The entropies for manipulating $V_1, \ldots, V_5$ are 0.9831, 1.7046, 1.7046, 1.3480, 0.4506, respectively. Based on the maximum entropy criterion, we choose variable V_2 or V_3 to be manipulated first. The maximum numbers l_i of DAGs for manipulating one of $V_1, \ldots, V_5$ are 8, 3, 3, 6, 10, respectively. Based on the minimax criterion, we also choose variable V_2 or V_3 to be manipulated first.

Although the same variable V_2 or V_3 is chosen to be manipulated first in the above example, in general, the choice may be different based on the two criteria. The minimax criterion tends to be more conservative, and the entropy criterion tends to be more uniform. For example, consider two interventions for an equivalence class with 10 DAGs: one splits the class into 8 subclasses with the numbers $(l_1, \ldots, l_8) = (1,1,1,1,1,1,1,3)$ of DAGs, the other splits it into 5 subclasses with the numbers of DAGs equal to (2,2,2,2,2). Then the minimax criterion chooses the second intervention, while the maximum entropy criterion chooses the first intervention.

To find the number (l_i for $i = 1, \cdots, M$), we need to enumerate all DAGs in the class $[G]$ and then count the number l_i of DAGs with the same orientations as $e(V)_i$. As discussed in Section 4.2, the upper bound of the complexity for calculating all l_i is $O(\min\{g!, 2^h\})$. Generally the size g of a chain component is much less than the number n of the full variable set and the number h of undirected edges in a chain component is not very large. In the following example, we show a special case with a tree structure, where the calculation is easy.

Example 2. In this example, we consider a special case that a chain component has a tree structure. It does not mean that a DAG is a tree, and it is not uncommon in a chain component (see Figure 1). Since there are no v-structures in any chain component, all undirected edges in a subtree can be oriented as long as we find its root. Manipulating a node V in a tree, we can

Orientation	$V_2 \leftarrow V_1 \rightarrow V_3$	$V_2 \rightarrow V_1 \rightarrow V_3$	$V_2 \rightarrow V_1 \leftarrow V_3$	$V_2 \leftarrow V_1 \leftarrow V_3$
DAGs	$\{1,2\}$	$\{3\}$	$\{4,5,7,8,9,10,11,12\}$	$\{6\}$
l_i	2	1	8	1
Entropy is 0.9831 and maximum l_i is 8				

Table 3: Manipulating V_1

Orientation						
DAGs	$\{8,9,11\}$	$\{10,12\}$	$\{3,4,5\}$	$\{2\}$	$\{1,6\}$	$\{7\}$
l_i	3	2	3	1	2	1
Entropy is 1.7046 and maximum l_i is 3						

Table 4: Manipulating V_2

Orientation					
DAGs	$\{1,2,3,4,6,7\}$	$\{5\}$	$\{8\}$	$\{9,10\}$	$\{11,12\}$
l_i	6	1	1	2	2
Entropy is 1.3480 and maximum l_i is 6					

Table 5: Manipulating V_4

Orientation	$V_4 \rightarrow V_5$	$V_4 \leftarrow V_5$
DAGs	$\{1,2,3,4,5,6,7,8,9,10\}$	$\{11,12\}$
l_i	10	2
Entropy is 0.4506 and maximum l_i is 10		

Table 6: Manipulating V_5

determinate all orientations of edges connecting V, and thus all subtrees that are emitted from V can be oriented, but only one subtree with V as a terminal cannot be oriented. Suppose that node V connects M undirected edges, and let l_i denote the number of nodes in the ith subtree connecting V for $i = 1,\ldots,M$. Since each node in the ith subtree may be the root of this subtree, there are l_i possible orientations for the ith subtree. Thus we have the entropy for manipulating V

$$H_V = -\sum_{i=1}^{M} \frac{l_i}{L} log \frac{l_i}{L}.$$

Consider the chain component $\tau = \{V_1,\ldots,V_4\}$ of the chain graph G^* in Figure 1, which has a tree structure. In Table 7, the first column gives variables to be manipulated, the second column gives possible orientations via the intervention, the third column gives the equivalence subclasses (see Figure 2) for each orientation, the fourth column gives the number l_i of possible DAGs for the ith

orientation and the last column gives the entropy for each intervention. From Table 7, we can see that manipulating V_1 or V_2 has the maximum entropy and the minimax size.

Intervention	Orientation	Subclass of DAGs	l_i	H_V
V_1	$V_2 \leftarrow V_1 \rightarrow V_3$	G	1	1.0397
	$V_2 \rightarrow V_1 \rightarrow V_3$	G_1, G_2	2	
	$V_2 \leftarrow V_1 \leftarrow V_3$	G_3	1	
V_2	$V_4 \leftarrow V_2 \leftarrow V_1$	G, G_3	2	1.0397
	$V_4 \leftarrow V_2 \rightarrow V_1$	G_1	1	
	$V_4 \rightarrow V_2 \rightarrow V_1$	G_2	1	
V_3	$V_1 \rightarrow V_3$	G, G_1, G_2	3	0.5623
	$V_1 \leftarrow V_3$	G_3	1	
V_4	$V_4 \leftarrow V_2$	G, G_1, G_3	3	0.5623
	$V_4 \rightarrow V_2$	G_2	1	

Table 7: Manipulating variables in a chain component with a tree structure.

An efficient algorithm or an approximate algorithm is necessary when both g and h are very large. A simulation algorithm can be used to estimate l_i/L. In this simulation method, we randomly take a sample of DAGs without any v-structure from the class $[G]$. To draw such a DAG, we randomly generate a permutation of all nodes in the class, orient all edges backwards in the direction of the permutation, and keep only the DAG without any v-structure. There may be some DAGs in the sample which are the same, and we keep only one of them. Then we count the number l_i' of DAGs in the sample which have the same orientation as $e(V)_i$. We can use l_i'/L' to estimate l_i/L, where $L' = \sum_i l_i'$. When the sample size tends to infinite, all DAGs in the class can be drawn, and then the estimate l_i'/L' tends to l_i/L. Another way to draw a DAG is that we randomly orient each undirected edge of the essential graph, but we need to check whether there is any cycle besides v-structure.

5. Simulation

In this section, we use two experiments to evaluate the active learning approach and the optimal designs via simulations. In the first experiment, we evaluate a whole process of structural learning and orientation in which we first find an essential graph using the PC algorithm and then orient the undirected edges using the approaches proposed in this paper. In the second experiment, we compare various designs for orientations starting with the same underlying essential graph. For both experiments, the DAG (1) in Figure 4 is used as the underlying DAG and all variables are binary. Its essential graph is given in Figure 3 and there are other 11 DAGs which are Markov equivalent to the underlying DAG (1), as shown in Figure 4. This essential graph can also be seen as a chain component of a large essential graph. All conditional probabilities $P(v_j|pa(v_j))$ are generated from the uniform distribution $U(0,1)$. We repeat 1000 simulations with the sample size $n = 1000$.

In each simulation of the first experiment, we first use the PC algorithm to find an essential graph with the significance level $\alpha = 0.15$ with which the most number of true essential graphs were obtained among various significance levels in our simulations. Then we use the intervention approach proposed in Section 3 to orient undirected edges of the essential graph. To compare the performances of orientations for different significance levels and sample sizes used in

intervention experiments, we run simulations for various combinations of significance levels $\alpha_I = 0.01, 0.05, 0.10, 0.15, 0.20, 0.30$ and sample sizes $n_I = 50, 100, 200, 500$ in intervention experiments. To compare the performance of the experiment designs, we further give the numbers of manipulated variables that are necessary to orient all undirected edges of the same essential graphs in various intervention designs. We run the simulations using R 2.6.0 on an Intel(R) Pentium(R) M Processor with 2.0 GHz and 512MB RAM and MS XP. It takes averagely 0.4 second of the processor time for a simulation, and each simulation needs to finish the following works: (1) generate a joint distribution and then generate a random sample of size $n = 1000$, (2) find an essential graph using the PC algorithm, (3) find an optimal design, and (4) repeatedly generate experimental data of size n_I until identifying a DAG.

To make the post-intervention distribution $P'(v_i|pa(v_i))$ different from the pre-intervention $P(v_i|pa(v_i))$, we use the post-intervention distribution of the manipulated variable V_i as follows

$$P'(v_i|pa(v_i)) = P'(v_i) = \begin{cases} 1, & P(v_i) \leq 0.5; \\ 0, & \text{otherwise.} \end{cases}$$

To orient an undirected edge $V_i - V_j$, we implemented both the independence test of the manipulated V_i and its each neighbor variable V_j for randomized experiments and the equivalence test of pre- and post-intervention distributions (i.e., $P_{V_i}(v_j) = P(v_j)$ for all v_j) in our simulations. Both tests have the similar results and the independence test is little more efficient than the equivalence test. To save space, we only show the simulation results of orientations obtained by the equivalence test and the optimal design based on the maximum entropy criterion in Table 8, and other designs have the similar results of orientations.

To evaluate the performance of orientation, we define the percentage of correct orientations as the ratio of the number of correctly oriented edges to the number of edges that are obtained from the PC algorithm and belong to the DAG (1) in Figure 4. The third column λ in Table 8 shows the average percentages of correctly oriented edges of the DAG (1) in 1000 simulations. To separate the false orientations due to the PC algorithm from those due to intervention experiments, we further check the cases that the essential graph in Figure 3 is correctly obtained from the PC algorithm. The fourth column *m* shows the number of correct essential graphs obtained from the PC algorithm in 1000 simulations. In the fifth column, we show the percentage λ' of correct orientations for the correct essential graph. Both λ and λ' increase as n_I increases. Comparing λ and λ', it can be seen that there are more edges oriented correctly when the essential graph is correctly obtained from the PC algorithm. From the sixth to eleven columns, we give the cumulative distributions of the number of edges oriented correctly when the essential graph is correctly obtained. The column labeled '$\geq i$' means that we correctly oriented more than or equal to *i* of 6 edges of the essential graph in Figure 3, and the values in this column denote the percents of DAGs with more than or equal to *i* edges correctly oriented in those simulations. For example, the column '≥ 5' means that more than or equal to 5 edges are oriented correctly (i.e., the DAGs (1), (2) and (6) in Figure 4), and 0.511 in the first line means that 51.1% of $m = 409$ correct essential graphs were oriented with '≥ 5' correct edges. The column '6' means that the underlying DAG (1) is obtained correctly. From this column, it can be seen that more and more DAGs are identified correctly as the size n_I increases. The cumulative distribution for ≥ 0 is equal to one and is omitted. From these columns, it can be seen that more and more edges are correctly oriented as the size n_I increases. From λ and λ', we can see that a larger α_I is preferable for a smaller size n_I, and a smaller α_I is preferable for a larger

n_I. For example, $\alpha_I = 0.20$ is the best for $n_I = 50$, $\alpha_I = 0.10$ for $n_I = 100$, $\alpha_I = 0.05$ for $n_I = 200$, $\alpha_I = 0.01$ for $n_I = 500$.

					The number of edges oriented correctly					
n_I	α_I	λ	m	λ'	6	≥ 5	≥ 4	≥ 3	≥ 2	≥ 1
50	.01	.672	409	.758	0.401	0.511	0.868	0.870	0.927	0.973
	.05	.699	409	.782	0.496	0.616	0.829	0.839	0.934	0.976
	.10	.735	418	.808	0.538	0.646	0.833	0.868	0.969	0.993
	.15	.745	407	.821	0.516	0.690	0.855	0.909	0.966	0.990
	.20	.756	404	.826	0.564	0.723	0.832	0.899	0.963	0.978
	.30	.741	373	.819	0.501	0.729	0.823	0.920	0.965	0.979
100	.01	.761	401	.850	0.586	0.706	0.910	0.925	0.975	0.995
	.05	.774	408	.846	0.588	0.721	0.885	0.919	0.973	0.993
	.10	.806	425	.878	0.668	0.814	0.896	0.925	0.974	0.993
	.15	.794	410	.868	0.624	0.790	0.878	0.932	0.985	1.000
	.20	.788	382	.875	0.626	0.812	0.890	0.948	0.982	0.992
	.30	.798	417	.861	0.583	0.777	0.856	0.959	0.988	1.000
200	.01	.822	421	.901	0.724	0.808	0.945	0.948	0.988	0.995
	.05	.836	402	.911	0.701	0.853	0.950	0.973	0.995	0.995
	.10	.833	408	.900	0.686	0.863	0.917	0.949	0.993	0.995
	.15	.823	382	.901	0.696	0.851	0.911	0.955	0.995	1.000
	.20	.826	395	.886	0.658	0.820	0.889	0.962	0.990	0.997
	.30	.822	402	.887	0.614	0.828	0.905	0.975	0.998	1.000
500	.01	.870	369	.966	0.878	0.943	0.984	0.992	1.000	1.000
	.05	.869	388	.940	0.802	0.920	0.951	0.977	0.995	0.997
	.10	.863	399	.936	0.762	0.905	0.952	0.995	1.000	1.000
	.15	.859	433	.926	0.723	0.898	0.956	0.986	0.995	1.000
	.20	.846	390	.923	0.703	0.890	0.956	0.990	0.997	1.000
	.30	.834	389	.893	0.599	0.820	0.949	0.992	1.000	1.000

Table 8: The simulation results

In the second experiment, we compare the numbers of manipulated variables to orient the same underlying essential graph for different experimental designs. In the following simulations, we set $n_I = 100$ and $\alpha_I = 0.1$, and all orientations start with the true essential graph in Figure 3. As shown in Section 4.2, the optimal batch design and the design by the greedy method always need three variables to be manipulated for orientation of the essential graph. For the optimal sequential designs, the frequencies of the numbers of manipulated variables in 1000 simulations are given in Table 9. In the random design labeled 'Random', we randomly select a variable to be manipulated at each sequential step, only one variable is manipulated for orientations in 268 of 1000 simulations, and four variables are manipulated in 55 of 1000 simulations. In the middle of Table 9, we show the simulation results of the optimal sequential designs based on the minimax criterion and its approximate designs obtained by drawing a sample of DAGs. The minimax design needs only one or two variables to be manipulated in all 1000 simulations. We show three approximate designs which draw h, $h \times 5$ and $h \times 10$ DAGs from a chain component with h undirected edges respectively. For

example, the sample sizes of DAGs from the initial essential graph $[G]$ with $h = 6$ undirected edges are 6, 30 and 60, respectively. As the sample size increases, the distribution of the manipulated variable numbers tends to the distribution for the exact minimax design. The optimal sequential design based on the maximum entropy criterion has a very similar performance as that based on the minimax criterion, as shown in the bottom of Table 9. According to Table 9, all of the sequential intervention designs (Random, Minimax, Entropy and their approximations) are more efficient than the batch design, and the optimal designs based on the minimax and the maximum entropy criteria are more efficient than the random design.

	m^*			
Design	1	2	3	4
Random	268	475	202	55
Minimax	437	563	0	0
Approx. (h)	372	469	159	0
Approx. ($h \times 5$)	413	573	14	0
Approx. ($h \times 10$)	426	574	0	0
Entropy	441	559	0	0
Approx. (h)	375	454	171	0
Approx. ($h \times 5$)	435	547	18	0
Approx. ($h \times 10$)	425	574	1	0

m^* denotes the number of manipulated variables

Table 9: The frequencies of the numbers of interventions

6. Conclusions

In this paper, we proposed a framework for active learning of causal structures via intervention experiments, and further we proposed optimal designs of batch and sequential interventions based on the minimax and the maximum entropy criteria. A Markov equivalence class can be split into subclasses by manipulating a variable, and a causal structure can be identified by manipulating variables repeatedly. We discussed two kinds of external intervention experiments, the randomized experiment and the quasi-experiment. In a randomized experiment, the distribution of a manipulated variable does not depend on its parent variables, while in a quasi-experiment, it may depend on its parents. For a randomized experiment, the orientations of an undirected edge can be determined by testing the independence of the manipulated variable and its neighbor variable only with experimental data. For a quasi-experiment, the orientations can be determined by testing the equivalence of pre- and post-intervention distributions with both experimental and observational data. We discussed two optimal designs of batch and sequential interventions. For the optimal batch design, a smallest set of variables to be manipulated is found before interventions, which is sufficient to orient all undirected edges of an essential graph. But the optimal batch design does not use orientation results obtained by manipulating the previous variables during the intervention process, and thus it may be less efficient than the optimal sequential designs. For the optimal sequential design, we choose a variable to be manipulated sequentially such that the current Markov equivalence class can be reduced to a subclass with potential causal DAGs as little as possible. We discussed two

criteria for optimal sequential designs, the minimax and the maximum entropy criteria. The exact, approximate and greedy methods are presented for finding the optimal designs.

The scalability of the optimal designs proposed in this paper depends only on the sizes of chain components but does not depend on the size of a DAG since the optimal designs are performed separately within every chain component. As discussed in Section 4, the optimal designs need to find the number of possible DAGs in a chain component, which has a upper bound $\min\{2^h, g!\}$. When both the number h of undirected edges and the number g of nodes in a chain component are very large, instead of using the optimal designs, we may use the approximate designs via sampling DAGs. We checked several standard graphs found at the Bayesian Network Repository (http://compbio.cs.huji.ac.il/Repository/). We extracted their chain components and found that most of their chain components have tree structures and their sizes are not large. For example, ALARM with 37 nodes has 4 chain components with only two nodes in each component, HailFinder with 56 nodes has only one component with 18 nodes, Carpo with 60 nodes has 9 components with at most 7 nodes in each component, Diabets with 413 nodes has 25 components with at most 3 nodes, and Mumin 2 to Mumin 4 with over 1000 nodes have at most 21 components with at most 35 nodes. Moreover, all of those largest chain components have tree structures, and thus we can easily carry out optimal designs as discussed in Example 2.

In this paper, we assume that there are no latent variables. Though the algorithm can orient the edges of an essential graph and output a DAG based on a set of either batch or sequential interventions, the application of the method for learning causality in the real word is pretty limited because latent or hidden variables are typically present in real-world data sets.

Acknowledgments

We would like to thank the guest editors and the three referees for their helpful comments and suggestions that greatly improved the previous version of this paper. This research was supported by Doctoral Program of Higher Education of China (20070001039), NSFC (70571003, 10771007, 10431010), NBRP 2003CB715900, 863 Project of China 2007AA01Z43, 973 Project of China 2007CB814905 and MSRA.

Appendix A. Proofs of Theorems

Before proving Theorems 4 and 5, we first give a lemma which will be used in their proofs.

Lemma 10 *If a node $V \in \mathbb{V}$ is a parent of a node U in a chain component τ of G^* (i.e., $(V \to U) \in G^*$, $U \in \tau$, $V \in \mathbb{V}$ and $V \notin \tau$), then V is a parent of all nodes in τ (i.e., $(V \to W) \in G$ for any $W \in \tau$).*

Proof By (iii) of Lemma 3, $V \to U - W$ does not occur in any induced subgraph of G^*. Thus for any neighbor of U in the chain component τ, W and V must be adjacent in G^*. Because $V \notin \tau$, the edge between V and W is directed. There are two alternatives as shown in Figures 5 and 6 for the subgraph induced by $\{V, U, W\}$.

If it is the subgraph in Figure 6 (i.e., the $V \to W \in G'$ for any $G' \in [G]$), then $W \to U$ must be in G' for any $G' \in [G]$ in order to avoid a directed cycle, as shown in Figure 7. So $W \to U$ must be in G^*. It is contrary to the fact that $\{U, W\} \in \tau$ is in a chain component of G^*. So V must also be a parent of W. Because all variables in τ are connected by undirected edges in G^*_τ, V must be a parent

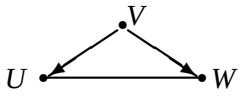

Figure 5: SG_1

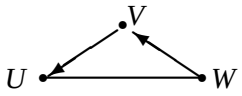

Figure 6: SG_2

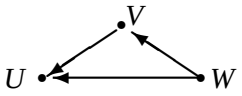

Figure 7: SG_3

of all other variables in τ. ■

Proof of Theorem 4. According to Lemma 10, if a node W outside a component τ points at a node V in τ, then W must point at each node U in τ. Thus W, V and U cannot form a v-structure. ■

Proof of Theorem 5. Suppose that Theorem 5 does not hold, that is, there is a directed path $V_1 \to \cdots \to V_k$ in G_τ which is not a directed cycle, but $W_1 \to \cdots \to W_i \to V_1 \to \cdots \to V_k \to W_{i+1} \to \cdots \to W_1$ is a directed cycle, where $W_i \notin \tau$. We denote this cycle as DC. From Lemma 10, W_i must also be a parent of V_k, and thus $W_1 \to \cdots \to W_i \to V_k \to W_{i+1} \to \cdots \to W_1$ is also a directed cycle, denoted as DC'. Now, every edge of DC' is out of G_τ. Similarly, we can remove all edges in other chain components from DC' and keep the path being a directed cycle. Finally, we can get a directed cycle in the directed subgraph of G^*. It contradicts the fact that G^* is an essential graph of a DAG. So we proved Theorem 5. ■

To prove Theorem 6, we first present an algorithm for finding the post-intervention essential graph $G^*_{e(V)}$ via the orientation $e(V)$, then we show the correctness of the algorithm using several lemmas, and finally we give the proof of Theorem 6 with $G^*_{e(V)}$ obtained by the algorithm. In order to prove that $G^*_{e(V)}$ is also a chain graph, we introduce an algorithm (similar to Step D of SGS and the PC algorithm in Spirtes et al., 2000) for constructing a graph, in which some undirected edges of the initial essential graph are oriented with the information of $e(V)$. Let τ be a chain graph of G^*, $V \in \tau$ and $e(V)$ be an orientation of undirected edges connecting V.

Algorithm 2 Find the post-intervention essential graph via orientation $e(V)$

Input: The essential graph G^* and $e(V)$

Output: The graph H

Orient the undirected edges connecting V in the essential graph G^* according to $e(V)$ and denote the graph as H.

Repeat the following two rules to orient some other undirected edges until no rules can be applied:

(i) if $V_1 \to V_2 - V_3 \in H$ and V_1 and V_3 are not adjacent in H, then orient $V_2 - V_3$ as $V_2 \to V_3$ and update H;

(ii) if $V_1 \to V_2 \to V_3 \in H$ and $V_1 - V_3 \in H$, then orient $V_1 - V_3$ as $V_1 \to V_3$ and update H.

return the graph H

It can be shown that H constructed by Algorithm 2 is a chain graph and H is equal to the post-intervention essential graph $G^*_{e(V)}$. We show those results with the following three Lemmas.

Lemma 11 *Let G^* be the essential graph of DAG G, τ be a chain component of G^* and I be a DAG over τ. Then there is a DAG $G' \in [G]$ such that $I = G'_\tau$ if and only if I is a DAG with the same skeleton as G^*_τ and without v-structures.*

Proof If there is a DAG $G' \in [G]$ such that $I = G'_\tau$, we have from Lemma 1 that I is a DAG with the same skeleton as G^*_τ and without v-structures.

Let I be a DAG with the same skeleton as G^*_τ and without v-structures, and G' be any DAG in the equivalence class $[G]$. We construct a new DAG I' from G' by substituting the subgraph G'_τ of G' with I. I' has the same skeleton as G'. From Theorems 4 and 5, I' has the same v-structures as G'. Thus I' is equivalent to G' and $I' \in [G]$. ■

Lemma 12 *Let H be a graph constructed by Algorithm 2. Then H is a chain graph.*

Proof If H is not a chain graph, there must be a directed cycle in subgraph H_τ for some chain component of G^*. Moreover, G^*_τ is chordal and $H \subset G^*$, and thus H_τ is chordal too. So we can get a three-edge directed cycle in H_τ as given in Figure 8 or 9.

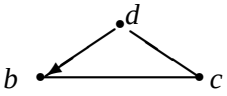

Figure 8: SG_6

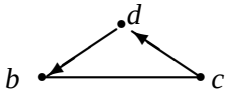

Figure 9: SG_{6_1}

If Figure 9 is a subgraph of H obtained at some step of Algorithm 2, then the undirected edge b—c is oriented as $b \leftarrow c$ according to Algorithm 2. Thus only Figure 8 can be a subgraph of H.

According to Lemma 10, we have that the directed edge $d \rightarrow b$ is not in G^*. Since all edges connecting a have been oriented in Step 1 of Algorithm 2, $d \rightarrow b$ is not an edge connecting a. So $d \rightarrow b$ must be identified at step 2 of Algorithm 2. There are two situations, one is to avoid a v-structure as shown in Figure 10, the other is to avoid a directed cycle as Figure 13.

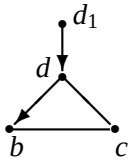

Figure 10: SG_7

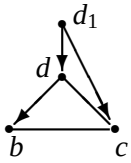

Figure 11: SG_8

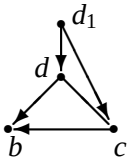

Figure 12: SG_9

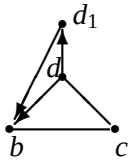

Figure 13: SG_{10}

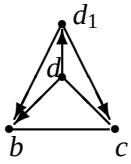

Figure 14: SG_{11}

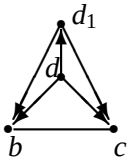

Figure 15: SG_{12}

We can arrange all directed edges in H_τ in order of orientations performed at Step 2 of Algorithm 2. First, we prove that the directed edge $d \rightarrow b$ in Figure 8 is not the first edge oriented at Step 2 of Algorithm 2.

In the first case as Figure 10, if $d \to b$ is the first edge oriented at Step 2 of Algorithm 2, we have $d_1 = a$. Because b and a are not adjacent, and $d—c$ is an undirected edge in H, we have that $d_1 \to c$ must be in H as Figure 11, where $d_1 = a$. Now we consider the subgraph $b—c \leftarrow d_1$. According to the rules (i) and (ii) in Algorithm 2, we have that $b \leftarrow c$ is in $G^*_{e(a)}$ as Figure 12, which contradicts the assumption that $b—c \in H$.

In the second case as Figure 13, if $d \to b$ is the first edge oriented at Step 2 of Algorithm 2, we have $d_1 = a$.

Considering the structure $d_1 \to b—c$ and that $d—c$ is an undirected edge in H, we have that $d_1 \to c$ must be in H as Figure 14. Now we consider the subgraph of $\{d, d_1, c\}$. By Algorithm 2, $d \to c$ is in H as Figure 15, which contradicts the assumption that $d—c \in H$. Thus we have that the first edge oriented at Step 2 of Algorithm 2 is not in any directed cycle. Suppose that the first k oriented edges at Step 2 of Algorithm 2 are not in any directed cycle. Then we want to prove that the $(k+1)th$ oriented edge is also not in a directed cycle.

Let $d \to b$ be the $(k+1)th$ oriented edge at Step 2 of Algorithm 2, and Figure 8 be a subgraph of H. There are also two cases as Figures 10 and 13 for orienting $d \to b$.

In the case of Figure 10, since $d_1 \to d$ is in the first k oriented edges and $d—c \in H$, we have that $d_1 \to c$ must be in H. We also get that $b \leftarrow c$ must be in H as Figure 12, which contradicts the assumption that $b—c \in H$.

In the case of Figure 10, since $d_1 \to b$ and $d \to d_1$ are in the first k oriented edges and $b—c \in H$, we have that $d_1 \to c$ must be in H. We also get that $d \leftarrow c$ must be in H as Figure 15, which contradicts the assumption that $d—c \in H$. So the $(k+1)$th oriented edge is also not in any directed cycle. Now we can get that every directed edge in H_τ is not in any directed cycle. It implies that there are no directed cycles in H_τ, and thus H is a chain graph. ■

Lemma 13 *Let $G^*_{e(V)}$ be the post intervention essential graph with the orientation $e(V)$ and H be the graph constructed by Algorithm 2. We have $G^*_{e(V)} = H$.*

Proof We first prove $G^*_{e(a)} \subseteq H$. We just need to prove that all directed edges in H must be in $G^*_{e(a)}$. We use induction to finish the proof.

After Step 1 of Algorithm 2, all directed edges in H are in $G^*_{e(a)}$. We now prove that the first directed edge oriented at Step 2 of Algorithm 2, such as $b \leftarrow c$, is in $G^*_{e(a)}$. Because $b \leftarrow c$ must be oriented by the rule (i) of Algorithm 2, there must be a node $d \notin \tau$ such that $b—c \leftarrow d$ is the subgraph of H. So $b \leftarrow c \leftarrow d$ must be a subgraph in each $G' \in G^*_{e(a)}$. Otherwise, $b \to c \leftarrow d$ forms a v-structure such that $G' \notin [G]$. Thus we have $b \leftarrow c \in G^*_{e(a)}$.

Suppose that the first k oriented edges at Step 2 of Algorithm 2 are in $G^*_{e(a)}$. We now prove that the $(k+1)$th oriented edge at Step 2 of Algorithm 2 is also in $G^*_{e(a)}$. Denoting the $(k+1)$th oriented edge as $l \leftarrow h$, according to the rules in Algorithm 2, there are two cases to orient $l \leftarrow h$ as shown in Figures 16 and 17.

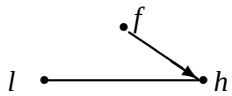

Figure 16: SG_4

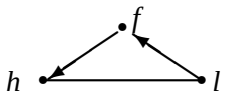

Figure 17: SG_5

In Figure 16, because $f \to h$ is in every DAG $G' \in G^*_{e(a)}$, in order to avoid a new v-structure, we have that $l \leftarrow h$ must be in every DAG $G' \in G^*_{e(a)}$. Thus we have $l \leftarrow h \in G^*_{e(a)}$. In Figure 17, because $l \to f$ and $f \to h$ are in every DAG $G' \in G^*_{e(a)}$, in order to avoid a directed cycle, we have that $h \leftarrow l$ must be in every DAG $G' \in G^*_{e(a)}$. Thus we have $h \leftarrow l \in G^*_{e(a)}$. Now we get that the $(k+1)$th oriented edge at Step 2 of Algorithm 2 is also in $G^*_{e(a)}$. Thus all directed edges in H are also in $G^*_{e(a)}$ and then we have $G^*_{e(a)} \subseteq H$.

Because H is a chain graph by Lemma 12, we also have $H \subseteq G^*$. By Lemma 11, for any undirect edge a—b of H_τ where τ is a chain component of H, there exist G_1 and $G_2 \in G^*_{e(a)}$ such that $a \to b$ occurs in G_1 and $a \leftarrow b$ occurs in G_2. It means that a—b also occurs in $G^*_{e(a)}$. So we have $H \subseteq G^*_{e(a)}$, and then $G^*_{e(a)} = H$. ■

Proof of Theorem 6. By definition of $G^*_{e(V)}$, we have that $G^*_{e(V)}$ has the same skeleton as the essential graph G^* and contains all directed edges of G^*. That is, all directed edges in G^* are also directed in $G^*_{e(V)}$. So property 2 of Theorem 6 holds. Property 3 of Theorem 6 also holds because all DAGs represented by $G^*_{e(V)}$ are Markov equivalent. From Lemmas 12 and 13, we can get that $G^*_{e(V)}$ is a chain graph. ■

Proof of Theorem 7. We first prove property 1. Let $C = ch(V_k) \setminus \tau$. Then $B = ne(V_k) \setminus C$ contains all parents of V_k and the children of V_k in τ. Let $A = An(\{B, V_k\})$ be the ancestor set of all nodes in $\{B, V_k\}$. Since V_i is a parent of V_k for property 1, we have $V_i \in A$. The post-intervention joint distribution of A is

$$P_{V_i}(A) = P'(v_i|pa(v_i)) \prod_{v_j \in A \setminus V_i} P(v_j|pa(v_j)). \tag{1}$$

Let $U = A \setminus \{B, V_k\}$. Then we have from the post-intervention joint distribution (1)

$$\begin{aligned} P_{V_i}(v_k|B) &= \frac{\sum_U P'(v_i|pa(v_i)) \prod_{v_j \in A \setminus V_i} P(v_j|pa(v_j)))}{\sum_{U,V_k} P'(v_i|pa(v_i)) \prod_{V_j \in A \setminus V_i} P(v_j|pa(v_j))} \\ &= \frac{\sum_U P'(v_i|pa(v_i)) \prod_{v_j \in A \setminus \{ch(V_k) \cap \tau, V_k\}} P(v_j|pa(v_j)) \prod_{v_j \in \{ch(V_k) \cap \tau, V_k\}} P(v_j|pa(v_j))}{\sum_{U,V_k} P'(v_i|pa(v_i)) \prod_{v_j \in A \setminus \{ch(V_k) \cap \tau, V_k\}} P(v_j|pa(v_j)) \prod_{v_j \in \{ch(V_k) \cap \tau, V_k\}} P(v_j|pa(v_j))}, \end{aligned}$$

where $\sum_U$ denotes a summation over all variables in the set U.

Below we want to factorize the denominator into a production of summation over U and summation over V_k. First we show that the factor $P'(v_i|pa(v_i)) \prod_{v_j \in A \setminus \{ch(V_k) \cap \tau, V_k\}} P(v_j|pa(v_j))$ does not contain V_k because V_k appears only in the conditional probabilities of $ch(V_k)$ and the conditional probability of V_k. Next we show that $\prod_{v_j \in \{ch(V_k) \cap \tau, V_k\}} P(v_j|pa(v_j))$ does not contain any variable in U. From definition of B, we have $B \supseteq (ch(V_k) \cap \tau)$. Then from definition of U, we have that V_j in $\{ch(V_k) \cap \tau, V_k\}$ is not in U. Now we just need to show that any parent of any node V_j in $\{ch(V_k) \cap \tau, V_k\}$ is also not in U:

1. By definitions of B and U, the parents of V_k is not in U.

2. Consider parents of nodes in $\{ch(V_k) \cap \tau\}$. Let W is such a parent, that is, $W \to V_j$ for $V_j \in \{ch(V_k) \cap \tau\}$. There is a head to head path ($W \to V_j \leftarrow V_k$). We show that W is not in U separately for two cases: $W \in \tau$ and $W \notin \tau$. For the first case of $W \in \tau$, there is an undirected

edge between W and V_k in G^*_τ since there is no v-structure in the subgraph G'_τ for any $G' \in [G]$. Then from definition of B, we have $W \in B$. For the second case of $W \notin \tau$, W must be a parent of V_k by Lemma 10, and then W is in B. Thus we obtain $W \notin U$.

We showed that the factor $\prod_{V_j \in \{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j))$ does not contain any variable in U. Thus the numerator and the summations over U and V_k in the denominator can be factorized as follows

$$\begin{aligned}
&P_{V_i}(v_k|B) \\
&= \frac{\prod_{v_j \in \{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j)) \sum_U P'(v_i|pa(v_i)) \prod_{v_j \in A\setminus\{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j))}{\sum_{V_k} \prod_{v_j \in \{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j)) \sum_U P'(v_i|pa(v_i)) \prod_{v_j \in A\setminus\{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j))} \\
&= \frac{\prod_{v_j \in \{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j))}{\sum_{V_k} \prod_{v_j \in \{ch(V_k)\cap\tau, V_k\}} P(v_j|pa(v_j))} = P(v_k|B).
\end{aligned}$$

Thus we proved property 1.

Property 2 is obvious since manipulating V_i does not change the distribution of its parent V_k. Formally, let $an(V_k)$ be the ancestor set of V_k. If $V_k \in pa(V_i)$, then we have $P_{V_i}(an(v_k), v_k) = P(an(v_k), v_k)$ and thus $P_{V_i}(V_k) = P(V_k)$. ■

Proof of Theorem 8. Manipulating a node V_i will orient all of undirected edges connecting V_i. Thus the orientations of undirected edges do not depend on the order in which the variables are manipulated. If a sequence $\mathcal{S}$ is sufficient, then its permutation is also sufficient. ■

Proof of Theorem 9. Suppose that $\mathcal{S} = (V_1, \ldots, V_K)$ is a sufficient set. We delete a node, say V_i, from $\mathcal{S}$, and define $\mathcal{S}'_{[i]} = \mathcal{S} \setminus \{V_i\}$. If the set $\mathcal{S}'_{[i]}$ is no longer sufficient, then we can add other variables to $\mathcal{S}'_{[i]}$ without adding V_i such that $\mathcal{S}'_{[i]}$ becomes to be sufficient. This is feasible since any undirected edge can be oriented by manipulating either of its two nodes. Thus we have $\bigcap_{i=1}^{K} \mathcal{S}'_{[i]} = \varnothing$. Since all $\mathcal{S}'_{[i]}$ belong to $\mathbb{S}$, we proved $\bigcap_{\mathcal{S}\in\mathbb{S}} \mathcal{S} = \varnothing$.

Similarly, for each minimum sequence $\mathcal{S}$, we can define $\mathcal{S}'_{[i]}$ such that it does not contain V_i and it is a minimum sufficient set. Thus the intersection of all minimum sufficient sets is empty. ■

References

C. Aliferis, I. Tsamardinos, A. Statnikov and L. Brown. Causal explorer: A probabilistic network learning toolkkit for biomedical discovery. In *International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences*, pages 371-376, 2003.

S. A. Andersson, D. Madigan and M. D. Perlman. A characterization of markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25:505-541, 1997.

R. Castelo and M. D. Perlman. Learning Essential graph Markov models from data. In *Proceedings 1st European Workshop on Probabilistic Graphical Models*, pages 17-24, 2002.

G. F. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 116-125, 1999.

N. Friedman. Inferring cellular networks using probabilistic graphical models. *Science*, 303(5659):799-805, 2004.

Y. He, Z. Geng and X. Liang. Learning causal structures based on Markov equivalence class. In *ALT, Lecture Notes in Artificial Intelligence* 3734, pages 92-106, 2005.

D. Heckerman, D. Geiger and D. M. Chickering. Learning Bayesian networks: The Combination of knowledge and statistical data. *Machine Learning*, 20:197-243, 1995.

D. Heckerman. A Bayesian approach to causal discovery. *Data Mining and Knowledge Discovery*, 1(1):79-119, 1997.

R. Jansen, H. Y. Yu and D. Greenbaum. A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science*, 302(5644):449-453, 2003.

S. L. Lauritzen. *Graphical Models*. Oxford Univ. Press. 1996.

S. L. Lauritzen, T. S. Richardson. Chain graph models and their casual interpretations. *Journal of the Royal Statistical society series B-statistical methodology*,64:321-348, Part 3, 2002.

M. Kalisch, P. Buhlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm.*Journal of Machine Learning Research* 8, 613-636, 2007.

K. P. Murphy. Active Learning of Causal Bayes Net Structure, *Technical Report*, Department of Computer Science, University of California Berkeley, 2001.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

J. Pearl. Graphical models, causality and intervention. *Statist. Sci.*, 8:266-269, 1993.

J. Pearl. Causal inference from indirect experiments. *Artifcal Intelligence in Medicine*, 7:561-582, 1995.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

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

J. Tian and J. Pearl. Causal Discovery from Changes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 512-521, 2001a.

J. Tian and J. Pearl. Causal Discovery from Changes: a Bayesian Approach, UCLA Cognitive Systems Laboratory, Technical Report (R-285), 2001b.

S. Tong and D. Koller. Active learning for structure in bayesian networks. In *International Joint Conference on Artificial Intelligence*, pages 863-869, 2001.

T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 220-227, 1990.

J. Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley, New York. 1990.