An Efficient Causal Discovery Algorithm for Linear Models

Zhenxing Wang
The Chinese University of Hong Kong
Room 115, Ho Sin Hang Engineering Building
Shatin, N.T. Hong Kong
zxwang@cse.cuhk.edu.hk

Laiwan Chan
The Chinese University of Hong Kong
Room 1016, Ho Sin Hang Engineering Building
Shatin, N.T. Hong Kong
lwchan@cse.cuhk.edu.hk

ABSTRACT

Bayesian network learning algorithms have been widely used for causal discovery since the pioneer work [13, 18]. Among all existing algorithms, three-phase dependency analysis algorithm (TPDA) [5] is the most efficient one in the sense that it has polynomial-time complexity. However, there are still some limitations to be improved. First, TPDA depends on mutual information-based conditional independence (CI) tests, and so is not easy to be applied to continuous data. In addition, TPDA uses two phases to get approximate skeletons of Bayesian networks, which is not efficient in practice. In this paper, we propose a two-phase algorithm with partial correlation-based CI tests: the first phase of the algorithm constructs a Markov random field from data, which provides a close approximation to the structure of the true Bayesian network; at the second phase, the algorithm removes redundant edges according to CI tests to get the true Bayesian network. We show that two-phase algorithm with partial correlation-based CI tests can deal with continuous data following arbitrary distributions rather than only Gaussian distribution.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms, Performance

Keywords

Bayesian networks, causal modeling, graphical models

1. INTRODUCTION

Causal relation discovery was formulated theoretically as learning the structures of Bayesian networks by the pioneer SGS algorithm [17], which is a representative algorithm learning Bayesian networks by CI tests without knowing vertex ordering in advance. A significant limitation is that SGS algorithm requires an exponential number of CI tests. The following PC algorithm [16] enhances the SGS algorithm so that it is efficient when underlying Bayesian networks are sparse but still has exponential-time complexity at the worst case. TPDA is the first one that requires a polynomial number of CI tests and guarantees to return correct networks. The key concept TPDA relying on is monotone faithfulness and all analysis about monotone faithfulness is based on mutual information. The problem is that mutual information is not easy to be applied to continuous data if we do not have enough knowledge of distribution data following. In addition, calculating conditional mutual information given a set of variables can be very difficult when the size of the set is large. This difficulty makes searching the initial approximate skeletons of Bayesian networks not efficient.

In this paper, we propose a two-phase algorithm as an improved version of TPDA. The two-phase algorithm has following advantages: (i) it has polynomial running time and a more efficient phase to estimate initial skeletons than TPDA; (ii) it can deal with continuous data with arbitrary distributions; (iii) the correctness of output networks is guaranteed. Here is the general idea of two-phase algorithm. In phase one, the algorithm constructs a Markov random field from data. This Markov random field is used as the initial skeleton of the Bayesian network in phase two. In the next step, for every edge in the Markov random field, phase two decides whether it is a true edge in the underlying Bayesian network by CI tests. If an edge is a redundant edge, it will be deleted in phase two. After these two phases, we get the correct Bayesian network. This two-phase algorithm can serve as a general framework in the sense that it can be applied to whatever data as long as there is a way to conduct CI tests.

The difficulty of learning Markov random fields from data is that we have to conduct CI tests given a large set of variables, which is also the reason why TPDA uses two complicated phases to get the initial approximations of the skeletons of Bayesian networks. Two-phase algorithm employs partial correlation-based CI tests to overcome this difficulty. Partial correlation-based CI tests are widely used in causation discovering algorithms [17, 16, 12, 11, 14]. However, all existing algorithms only use partial correlation-based CI tests to deal with Gaussian variables. For non-Gaussian data, a series of causation discovering algorithms [15, 6, 7] are purposed and those algorithms are combined with PC algorithm to deal with data following arbitrary distributions. A recent attempt to apply partial correlation-based CI tests

KDD'10, July 25–28, 2010, Washington, DC, USA.

to non-Gaussian data can be found in [19]. In this paper, we will show that: (i) merely partial correlation-based CI tests are able to deal with linear model with arbitrary distributions; (ii) Gaussian distribution is only a special case of linear model; (iii) partial correlation-based CI tests satisfy monotone faithfulness according to experiment results.

The remainder of the paper is structured as follows. In section 2, we introduce important concepts of Markov random fields and Bayesian networks. Based on these concepts, we analyze the relation between Markov random fields and Bayesian networks and give the framework of two-phase algorithm. In section 3, we explain the idea of using partial correlation to measure conditional independence, and give a heuristic two-phase algorithm. In section 4, we give complexity analysis and correctness proof of two-phase algorithm. Some empirical results are shown and discussed in section 5. Finally, we conclude our work in section 6.

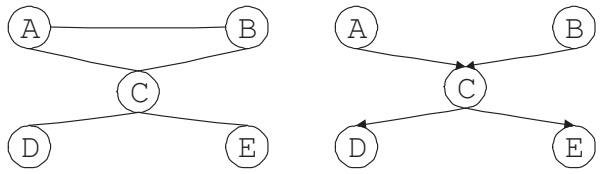
2. TWO-PHASE ALGORITHM

2.1 Preliminaries

Definition 2.1 Markov random field [9] is an undirected graph $G_M(V, E)$, where vertex set V denotes a set of random variables, satisfying the Markov property that an edge $\{u, v\} \notin E$ if and only if u and v are conditional independent given all other variables (denoted by $u \perp\!\!\!\perp v | V \backslash \{u, v\}$).

Definition 2.2 Bayesian network [18] is a directed acyclic graph $G_B(V, E)$, where vertex set V denotes a set of random variables, satisfying the Markov property that every variable is independent of any subset of its non-descendants variables conditioned on its parents. If we view all edges in G_B as undirected, we get an undirected graph which is the **skeleton** of G_B, denoted by $skeleton(G_B)$.

Definition 2.3 If a Markov random field $G_M(V, \tilde{E})$ and a Bayesian network $G_B(V, E)$ are defined on the same variable set V, we say that $G_M(V, \tilde{E})$ and $G_B(V, E)$ are a **Markov-Random-Field-Bayesian-Network pair**, abbreviated to **MRF-BN pair**.



(a) Markov random field (b) Bayesian network

Figure 1: An example of MRF-BN pair

Definition 2.4 $G_B(V, E)$ is a Bayesian network and P a distribution over V. If G_B represents all and only independence relations true in P, we say that G_B and P are **faithful** [18] to each other.

Remark 2.1 Under the condition that Bayesian network $G_B(V, E)$ is faithful to the underlying distribution P, there is an edge in G_B connecting vertices u and v if and only if there is no $C \subseteq V \backslash \{u, v\}$ making $u \perp\!\!\!\perp v | C$.

Definition 2.5 In a Bayesian network G_B, a vertex v is a **collider** [18] on a simple path U if there are two distinct adjacent vertices of v on U, and both of them are parents of v. Otherwise, we call v **noncollider**. Specially, if v is the parent of both of its adjacent vertices on U, v is called **diverging vertex**.

Definition 2.6 In a Bayesian network $G_B(V, E)$, two vertices u and v are **d-separated** [18] by $C \subseteq V \backslash \{u, v\}$ if and only if every simple path from u to v is blocked by C. A simple path U is **blocked** by C if and only if at least one noncollider on U is in C or at least one collider and all of its descendants are not in C. Specially, u and v can never be d-separated if there is an edge connecting u and v. In this paper, we call the set C **cut set**.

Remark 2.2 Given the condition that Bayesian network $G_B(V, E)$ and the underlying distribution P are faithful to each other, vertices u and v satisfy $u \perp\!\!\!\perp v | C$, where $C \subseteq V \backslash \{u, v\}$, if and only if u and v are d-separated by C. This fact links conditional independence with a graphical representation, which provides a way to convert causation discovery into a graph searching problem.

2.2 Framework of Two-Phase Algorithm

Two-Phase algorithm works in a similar way as TPDA. It first constructs an approximate skeleton of the underlying Bayesian network. And then it removes redundant edges by CI tests to get the true Bayesian network. Correctness of the algorithm requires that the approximate skeleton must contain all edges in the true Bayesian network while efficiency of the algorithm requires that the approximate skeleton contains as few redundant edges as possible. Two-phase algorithm is designed based on following facts which we will prove in section 4: (I) the Markov random field contains all edges in the skeleton of the corresponding Bayesian network in its MRF-BN pair as figure 1 illustrates. (II) for any two vertices u and v in a Bayesian network G_B, if there exits at least one set of vertices d-separating u and v, we are always capable to construct a cut set d-separating u and v by choosing vertices only from simple paths connecting u and v in G_B.

Figure 1 shows an example of MRF-BN pair. In figure 1 (a), there is a Markov random field containing all edges belonging to its counterpart Beyesian network in its MRF-BN pair as figure 1 (b) shows. However, there is also a redundant edge $\{A, B\}$ in figure 1 (a), which has to be removed before we get the correct Bayesian network.

The first phase of two-phase algorithm constructs a Markov random field from data according to the Markov property in definition 2.1. Namely, for each pair of variables u and v in data set V, two-phase algorithm does not add edge $\{u, v\}$ into Markov random field $G_M(V, E)$ if and only if $u \perp\!\!\!\perp v | V \backslash \{u, v\}$. The difficult part is how to conduct CI tests when the size of V is large. TPDA uses two complicated phases to estimate skeletons of Bayesian networks to avoid CI tests on large V. Two-phase algorithm employs partial correlation-based CI tests to overcome the difficulty, since partial correlation can be easily computed even when the size of V is large, which will be explained in section 3. The first fact guarantees that all edges in the skeleton of the Bayesian network we try to learn are included in this Markov random field. However, this Markov random field

Algorithm 1: Framework of two-phase algorithm

Data: Data set $\mathcal{D}$
Result: The underlying Bayesian network $G_B(V, E)$

```
begin
    // Phase I starts from here;
    Initialize a Markov random field G_M(Ṽ, Ẽ) by mapping each v ∈ Ṽ to a distinct variable in D and setting Ẽ = ∅;
    foreach pair {u, v} in Ṽ do
        if not u ⫫ v | Ṽ \ {u, v} then
            add {u, v} into Ẽ;
        end if
    end foreach
    // Phase II starts from here;
    Initialize V = Ṽ and E = Ẽ;
    foreach edge {u, v} ∈ E do
        Construct a set C by collecting all vertices on simple paths in G_B connecting u and v;
        foreach Z ⊆ C do
            if u ⫫ v | Z then
                remove {u, v} from E;
            end if
        end foreach
    end foreach
    Orient edges in E and output G_B(V, E)
end
```

may also contain some redundant edges which do not belong to the underlying Bayesian network. Thus, in phase two, the algorithm checks every edge in this Markov random field by CI tests to decide whether to remove it from the skeleton of the output Bayesian network. The second fact we listing above enables us to do CI tests in an efficient way. The idea is as follows. To make sure that an edge, say $\{u, v\}$, belongs to the skeleton of a Bayesian network, in principle we have to show that u and v are dependent conditioned on any set of the variables except u and v themselves. If this is the case, the number of situations we have to check grows exponentially with the number of variables in the Bayesian network, which is the reason why most of the Bayesian network learning algorithms are inefficient or even infeasible in practice. The second fact implies that when we try to find a set of vertices conditioned on which u and v are independent (d-separated), instead of searching the whole vertex set, we can restrict our search within the vertices on simple pathes connecting u and v. In another word, if we cannot find a subset of the set of all vertices on simple pathes connecting u and v to d-separate u and v, edge $\{u, v\}$ should be added to the skeleton of the Bayesian network. We describe above idea formally by the framework of two-phase algorithm.

There are some algorithms available for orienting edges in skeletons of Bayesian networks. The one we used in this paper is that in the SGS algorithm [17], which orients edges by identifying all V-structures in a Bayesian network. Without other information, edge orienting algorithm cannot distinguish Bayesian networks in a Markov equivalent class, which contains Baysian networks with same skeletons and V-structures. Therefore, two-phase algorithm just outputs an arbitrary Bayesian network from Markov equivalent class.

3. HEURISTIC TWO-PHASE ALGORITHM

In order to complete the framework two-phase algorithm, we need to figure out an efficient way to conduct CI tests. We notice that in phase two, if the size of the initial cut set C (denoted by $|C|$) is large, checking all $2^{|C|}$ subsets of C can be very time-consuming. In this section we first introduce a partial correlation based CI tests. And then, we explain how to use partial correlation based CI tests improving efficiency of two-phase algorithm.

3.1 Partial Correlation Based CI Test

Bayesian network learning algorithms always make some assumptions about continuous data to make it feasible to measure conditional independence. One commonly used assumption is that data following Gaussian distribution. In two-phase algorithm, we assume that data are governed by linear simultaneous equation models (SEMs). There are four advantages to make such assumption. The first is that under this linear assumption we can measure independence by correlation. The reason is that when there is only linear dependence between any two variables, correlation is equivalent to dependency. The second one is that linear model is also a commonly used assumption in many applications. Especially, Gaussian distribution is included in linear model in the sense that for every Gaussian distribution, it is always possible to find a linear SEM generating data following exactly the given Gaussian distribution. The third advantage is that conditional correlation can be estimated by partial correlation under linear assumption and partial correlation is easy to compute. The last point is that measuring conditional independence by partial correlation gives us an answer rather than yes or no, but a real number, and we can use information of this quantity to conduct heuristic CI tests, which makes two-phase algorithm more efficient. The first advantage is straightforward. We explain the second and third points in this section and leave the last points to be explained in section 3.2.

Before we explain what is partial correlation and how to use it measuring conditional independence, we have to introduce what are linear SEMs. In a linear SEM, data are generated by a batch of linear equations. Suppose that a set of random variables $X = \{x_1, x_2, ..., x_N\}$ are generated by the following procedure:

$$x_i = \sum_{j<i} c_{ij} x_j + e_i \tag{1}$$

Where disturbances e_i are continuous random variables independent with each others and coefficients c_{ij} are constants. We say that X is governed by a linear SEM. We can represent Bayesian networks by SEMs in the way that each vertex is the weighted summation of its parents plus a disturbance term. The Bayesian network in figure 1 corresponds to SEMs as follows: $x_A = e_A$, $x_B = e_B$, $x_C = c_{CA}x_A + c_{CB}x_B + e_C$, $x_D = c_{DC}x_C + e_D$ and $x_E = c_{EC}x_C + e_E$, where x_i corresponds to vertex i and c_{ij} is the weight of the edge connecting i and j.

Given the definition, we can explain why Gaussian distribution is a special case of linear SEMs. In other words, given a set $\bar{X} = \{\bar{x}_1, \bar{x}_2, ..., \bar{x}_N\}$ following multivariate Gaussian distribution $N(\boldsymbol{\mu}, \Sigma)$, there always exists a set of disturbances and coefficients making X follow exactly the same distribution as $\bar{X}$. It is obvious that if all disturbances e_i are Gaussian variables, X should follow multivariate Gaus-

sian distribution, since the summation of Gaussian variables is a Gaussian variable as well. Therefore, the only thing left is how to choose coefficients and mean and variance of each disturbance to make X has the same mean vector and variance-covariance matrix as $\bar{X}$. Since mean vector is easier to deal with than variance-covariance matrix, we get started by constructing variance-covariance matrix. We apply Cholesky decomposition to Σ. Because Σ is positive definite, it is guaranteed that there exists a lower triangular matrix $L = (l_{ij})_{N\times N}$ with strictly positive diagonal entries satisfying $\Sigma = LL'$, where L' is the transpose of L. And then, we generate a random vector $Z = (z_1, z_2, ..., z_N)'$, each of z_i following standard Gaussian distribution. We have $\hat{X} = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_N)' = LZ$ following Gaussian distribution $N(0, \Sigma)$. By expressing z_i in terms of $\hat{x}_j, j = 1, ..., i$ and setting disturbance $e_i = l_{ii}z_i$, we get a linear SEM which generates $\hat{X}$ following $N(0, \Sigma)$. Finally, we get the model $X = \hat{X} + \boldsymbol{\mu}$ which generates data following exactly $N(\boldsymbol{\mu}, \Sigma)$ as we desires. Therefore, data following linear SEMs is a more general assumption than Gaussian distribution.

Now, let us look at how to measure conditional independence under the linear SEMs. As we mention at the beginning of this section, we can use conditional correlation instead of conditional dependence under the linear model. However, conditional correlation is still not easy to compute. The following theorem, proof of which is given in [2], provides a way to estimate conditional correlation by partial correlation.

Theorem 3.1 For any two sets of random variables Y and Z, $\Sigma_{YY\cdot Z} = E(\Sigma_{YY|Z})$ if and only if $E(Y|Z) = \boldsymbol{\alpha} + \boldsymbol{\beta}Z$, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are regression coefficients.

Where $\Sigma_{YY\cdot Z}$ is the residual matrix of Y after eliminating the effect of Z and $\Sigma_{YY|Z}$ is the variance-covariance matrix of Y conditioned on Z. Theorem 3.1 basically says that if a set of random variables Y can be expressed as linear regression of another set of random variables Z, we can use partial correlation as an estimator of conditional correlation. Linear SEMs satisfy this condition. Namely, any set of variables can be expressed as linear regression of other variables. That is the reason why we can use partial correlation as an estimator of conditional correlation under the linear SEMs assumption. Partial correlation can be compute according to the definition. If a random vector $Y = (y_1, y_2, ..., y_m)'$ is a linear regression of another random vector $Z = (z_1, z_2, ..., z_n)'$, residual covariance matrix of Y given Z is defined as $\Sigma_{YY\cdot Z} = \Sigma_{YY} - \Sigma_{YZ}\Sigma_{ZZ}^{-1}\Sigma_{ZY}$. And partial correlation coefficient between y_i and y_j is defined as

$$\rho_{y_iy_j\cdot Z} = \frac{\sigma_{y_iy_j\cdot Z}}{\sqrt{\sigma_{y_iy_i\cdot Z}}\sqrt{\sigma_{y_jy_j\cdot Z}}}$$

where $\sigma_{y_iy_j\cdot Z}$ is the $(i,j)th$ item of $\Sigma_{YY\cdot Z}$. In two-phase algorithm, $\sigma_{uv\cdot V}$ being zero is an indicator of $u \perp\!\!\!\perp v|V$,

3.2 Two-Phase Algorithm with Heuristic CI Test

Partial correlation contains information more than yes or no when we use it to measure conditional independence. As a real number, partial correlation coefficient tells us how much dependency between two variables left after eliminating the effect of another set of random variables. In words of Bayesian network, partial correlation measures dependency left between two vertices after we trying to block the pathes connecting them in the Bayesian network by another set of vertices. After each CI test with different cut set, partial correlation may increase or decrease compared with previous one, which furnishes a clue of how to construct cut set in a heuristic way. The idea is as follows. Remark 2.2 provides a graphical interpretation of conditional independence, which indicates that finding a set of vertices making another two sets of vertices conditional independent is equivalent to constructing a cut set to d-separate these two sets of vertices. We can achieve this goal by blocking paths connecting these two sets of vertices one by one, until all paths between them are blocked, and hopefully decrease of absolute value of partial correlation can serve as an indicator that we successfully block a path without unblocking another one. This idea is formulated rigorously by the following concept.

Definition 3.1 Given that a Bayesian network $G_B(V, E)$ is faithful to the underlying distribution P, for a pair of vertices $\{u, v\}$ and $C \subseteq V\backslash\{u, v\}$, $|\rho_{uv\cdot C\backslash\{c\}}|$ is greater than $|\rho_{uv\cdot C}|$ for any $c \in C$, if and only if removing c from C unblocks some paths between u and v originally blocked by C. If this is true for every pair of vertices in G_B, we say that G_B and P are **monotone faithful** [5] to each other.

Remark 3.1 Monotone faithfulness enables us to analyze conditional independence quantitatively. Furthermore, it establishes a stronger connection between conditional independence and d-separation. Based on monotone faithfulness, we can convert CI tests into a graph searching problem. By blocking as many paths between two vertices as possible, we can finally find the set of variables conditioned on which these two vertices are independent, if there does exist such a set.

Based on the idea of monotone faithfulness, we give the algorithm of heuristic CI tests, called *separate*.

Algorithm 2: Separate(u, v, C)

```
1  begin
2    if |ρ_{uv·C}| < threshold then
3      return true;
4    end if
5    while |C| > 1 do
6      foreach c ∈ C do
7        if |ρ_{uv·C\{c}}| < threshold then
8          return true;
9        end if
10     end foreach
11     if min_{c∈C} |ρ_{uv·C\{c}}| > |ρ_{uv·C}|
12     then return false;
13     else c = argmin_{c∈C} ρ_{uv·C\{c}} and C = C \ {c};
14   end while
15   return false;
16 end
```

Given the condition that monotone faithfulness is satisfied, the procedure *separate* returns true if there exists a subset of C d-separating u and v. Now, we give the heuristic two-phase algorithm.

Algorithm 3: Heuristic two-phase algorithm

Data: Data set $\mathcal{D}$
Result: The underlying Bayesian network $G_B(V, E)$

```
begin
    // Phase I starts from here;
    Initialize a Markov random field G_M(Ṽ, Ẽ) by mapping each v ∈ Ṽ to a distinct variable in D and setting Ẽ = ∅;
    foreach pair {u, v} in Ṽ do
        if ρ_{uv·Ṽ\{u,v}} ≠ 0 then
            add {u, v} into Ẽ;
        end if
    end foreach
    // Phase II starts from here;
    Initialize V = Ṽ and E = Ẽ;
    foreach edge {u, v} ∈ E do
        Construct a set C by collecting all vertices on simple paths in G_B connecting u and v;
        if Separate(u, v, C) then
            remove {u, v} from E;
        end if
    end foreach
    Orient edges in E and output G_B(V, E)
end
```

4. COMPLEXITY ANALYSIS AND CORRECTNESS PROOF

4.1 Complexity Analysis

Suppose there are N random variables in the given data set. To get the approximate skeleton, the first phase of two-phase algorithm requires $O(N^2)$ CI tests, which is better than $O(N^4)$ required by TPDA. For the second phase of the framework of two-phase algorithm, there are at most $O(N^2)$ edges in E. For each edge, there are at most $O(2^N)$ cut set to be tested. Therefore, the number of CI tests required by the second phase is $O(N^2 * 2^N)$ and the total number of CI tests required by the framework of two-phase algorithm is $O(N^2 * 2^N)$ as well.

The first phase of the heuristic two-phase algorithm requires exactly the same number of CI tests as the framework. In the second phase, there are at most $O(N^2)$ edges in E and for each edge we need to call *separate* once. In the procedure *separate*, there are at most $N - 2$ vertices in the initial cut set C. At most $O(N^2)$ CI tests are required and each one takes $O(N^3)$ to calculate the inverse variance matrix on involved vertices Therefore, the time complexity of procedure *separate* is $O(N^5)$ at the worst case and the time complexity of the second phase of the heuristic two-phase algorithm is $O(N^7)$. The total time complexity of heuristic two-phase algorithm is $O(N^7)$ at the worst case. Although $O(N^7)$ seems not efficient enough, compared with those algorithms with exponential time at the worst case, this polynomial upper bound of running time is much better. In addition, the worst case only happens when the skeleton of the underlying Bayesian network is very closed to a complete graph, which is rare in practice. Most of the time, Bayesian networks are sparse and the running time of two-phase algorithm is reasonably short.

4.2 Correctness Proof

The correctness proof needs following assumptions: (i) monotone faithfulness is satisfied; (ii) measurement for conditional independence is accurate; (iii) there are no hidden vertices in the underlying Bayesian network.

Proposition 4.1 A Markov random field contains all edges in the skeleton of the Bayesian network of its MRF-BN pair.

Proof: For an edge in a Bayesian network, according to definition 2.2 and faithfulness assumption, there is no set of variables in the Bayesian network conditioned on which the endpoints of this edge are independent. Specially, the endpoints of this edge should be dependent conditioned on all other variables. According to definition 2.1, this edge should be in the corresponding Markov random field.

Proposition 4.2 For any two vertices u and v in a Bayesian network G_B, if there exits a set of vertices d-separating u and v, we are always capable to construct a cut set d-separating u and v by choosing vertices only from simple paths connecting u and v in G_B.

Proof: According to definition 2.6, to d-separate u and v, we need to block every simple path connecting u and v in G_B. Considering a particular simple path, there are two ways to block it according to definition 2.6, namely including one of noncolliders on it into the cut set or excluding one of collider and and all of its descendants from the cut set. None of simple paths requires vertices not belonging to it to block itself. Therefore, we only need to consider vertices on simple paths connecting u and v to d-separate u and v.

Proposition 4.1 and Proposition 4.2 together guarantee that the framework of two-phase algorithm is correct. To prove the heuristic two-phase algorithm is correct, we need to show that procedure *separate* always successfully find a cut set d-separating two vertices if such a set exists.

Lemma 4.1 At the end of phase one of two-phase algorithm, for any two vertices u and v, only two kinds of simple paths between u and v are unblocked given initial cut set C which contains all vertices on simple paths in G_B connecting u and v: (i) the edge connecting u and v, (ii) a path contains only one collider which is a child of both u and v.

Proof: According to proposition 4.1, all true edges are included in edge set E at the end of phase one, so the initial cut set C must contain every vertex lying on all paths connecting u and v in the underlying Bayesian network. For paths including only one noncollider, they are obviously blocked by the initial cut set. For paths containing more than one vertices between u and v, at least one of any two adjacent vertices is a noncollider according to definition 2.5, and this noncollider is included in the initial cut set. Therefore, all true paths containing more than one vertices between u and v must be blocked by the initial cut set.

Lemma 4.2 In the procedure *separate*, for any two vertices u and v, if removing a vertex c from cut set C blocks a simple path connecting u and v, it cannot unblock another path between u and v simultaneously.

Proof: If removing c blocks a path U and simultaneously unblocks another path V connecting u and v, c must be the only vertex in C which blocks V. Lemma 4.2 can be proved by showing that there must be a collider w on V, w and all its descendants not included in C, which means that V is also blocked by w. Lemma 4.2 is proved by mathematical induction.

Basis: for the first time a simple path connecting u and v is blocked by removing c from cut set C, any other simple paths connecting u and v cannot be unblocked at same time.

Proof: For the first time removing c from cut set C blocking an originally unblocked path U implies that c is a collider and the only vertex on U according to lemma 4.1 and none of descendants of c belongs to C. If removing c from the cut set unblocks another path V, c must block V when c belongs to the cut set. According to definition 2.6 c must be a noncollider on V, which means that c has descendants on V. We can prove that minimum descendants of c on V must be colliders on V (here we use "minimum descendants", since there are two minimum descendants if c is a diverging vertex on V). If minimum descendants of c on V were not colliders on V, there would exist a directed path from c to either u or v, which is a contradiction. Remember that c is a descendent of both u and v, so there should not be a directed path from c to either u or v. Otherwise there would be a cycle in Bayesian network. Therefore, minimum descendants of c on V are colliders. Without loss of generality, we call one of minimum descendants w. Since removing c from cut set C blocks path U, all descendants of c are not included in C according to definition 2.6, which implies that w and all descendants of w are not included in C. Thus, path V is also blocked by w besides c.

Inductive step: if removing previous n vertices from cut set does not block and unblock paths between u and v simultaneously, removing $(n+1)th$ vertex cannot block a path and unblock another path at same time.

Proof: Procedure *separate* removes a vertex from cut set only when partial correlation between u and v does not increase. Suppose that a path was unblocked by removing mth vertex ($m \leq n$). There should be no paths blocked simultaneously according to induction assumption, so that partial correlation between u and v would increase according to monotone faithfulness, which is a contradiction. Therefore, removing previous n vertices does not unblock any originally blocked paths, which means that the property presented by lemma 4.1 still holds during the procedure *separate*. Thus, we can use the same logic as we prove basis to show that removing $(n+1)th$ vertex cannot block a path and unblock another path at same time.

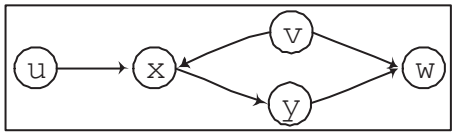


Figure 2: An illustration of correctness proof

Figure 2 illustrates the idea presented in the proof of lemma 4.2. There are two simple paths between u and v. We denote path $u{-}x{-}v$ by U and path $u{-}x{-}y{-}w{-}v$ by V. Only U is unblocked given the initial cut set $C = \{x, y, v\}$. To block U, procedure *separate* needs to remove x and all its descendants, y and w, from cut set. The minimum descendent of x, in this case w, has to be a collider on V. Otherwise, v, x, y and w form a cycle. To block U, y and w have to be removed before x. After w is removed, V is blocked by both x and w, so removing x cannot unblock V.

Proposition 4.3 If two vertices in a Bayesian network can be d-separated by some set of vertices, procedure *separate* always separates these two vertices (returns true) in the second phase of the heuristic two-phase algorithm.

Proof: According to lemma 4.2, procedure *separate* never unblocks an originally blocked path, since it would cause the partial correlation increasing. Therefore, the only thing we need to prove is that procedure *separate* can block all paths containing only one collider which is a child of both endpoints of the paths according to lemma 4.1. Consider a particular path U containing only one collider c which is a child of both endpoints of path U. Suppose that the minimum descendent of c in current cut set C is w. Using the same logic as we prove lemma 4.2, we can show that removing w from C will never unblock an originally blocked path. By keeping on removing the minimum descendent of c from C, we can finally removing c from C without unblocking any paths, which means procedure *separate* blocks path U successfully. Following the same way, procedure *separate* can block all paths containing only one collider which is a child of both endpoints of the paths.

5. EXPERIMENTAL RESULTS

We empirically evaluate two-phase algorithm and compare it with PC algorithm, TPDA algorithm and LiNGAM algorithm. Two-phase algorithm, PC algorithm and TPDA are all dependency analysis algorithms. And we implement these three algorithms based on partial correlation-based CI tests to compare efficiency and accuracy on linear Gaussian data. LiNGAM algorithm is specially designed for non-Gaussian data. Results of LiNGAM reported are all based on small networks. In this paper, we compare two-phase algorithm against LiNGAM algorithm on linear non-Gaussian data generated according to real world networks.

5.1 Data Generation Process and Evaluation Method

The networks used in the evaluation are listed in table 1, which are obtained from real decision support systems. Data are generated by linear SEMs according to these networks. Each variable is a weighted summation of its parents plus a disturbance. Weights are sampled from uniform distribution between 0.1 and 0.9. Disturbance terms are drawn from standard Gaussian, uniform and log-normal distribution, so that we can evaluate the performance of two-phase algorithm on both Gaussian and non-Gaussian setting.

We now describe how we quantify the performance of two-phase algorithm in terms of efficiency and accuracy. To evaluate efficiency, we use running time as our metric. For accuracy, we use precision, recall and F_1-measure as evaluation metrics, all of which are defined based on the similarity be-

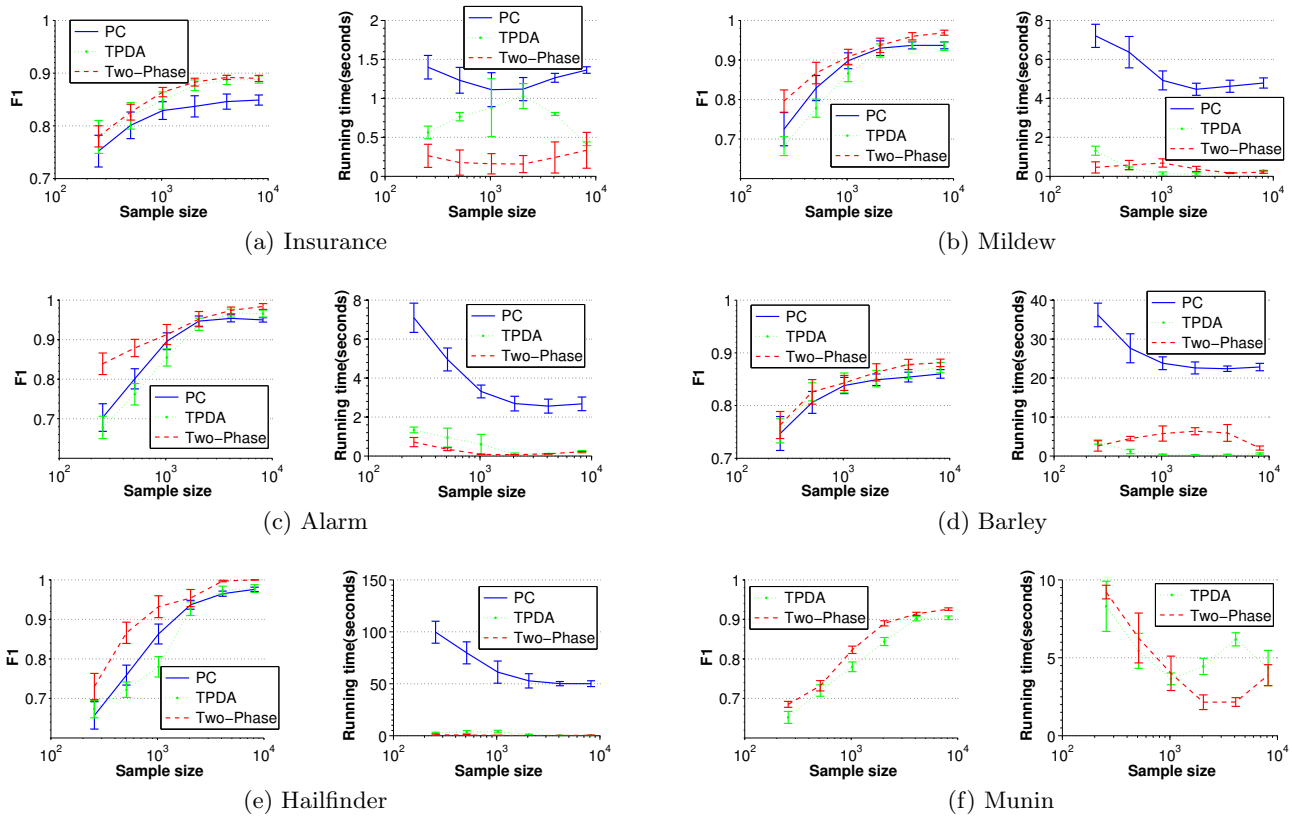


Figure 3: Compare two-phase algorithm with PC algorithm and TPDA on linear Gaussian data

Table 1: Bayesian Networks Used in Evaluation

Networks	Num. Variables	Num. Edges	Max In/Out Degree
Insurance [4]	27	52	3/7
Mildew [8]	35	46	3/3
Alarm [3]	37	46	4/5
Barley [10]	48	84	4/5
Hailfinder [8]	56	66	4/16
Munin [1]	189	282	3/15

tween the output Bayesian network and the target one. We use adjacent matrix to represent a Bayesian network. Let $\bar{A}$ denote the target adjacent matrix, and $\hat{A}$ the output one, precision P and recall R are defined as follows:

$$P = \frac{|\{(i,j) : \bar{A}(i,j) = \hat{A}(i,j) = 1\}|}{|\{(i,j) : \hat{A}(i,j) = 1\}|}$$

$$R = \frac{|\{(i,j) : \bar{A}(i,j) = \hat{A}(i,j) = 1\}|}{|\{(i,j) : \bar{A}(i,j) = 1\}|}$$

Given precision and recall, F_1 is defined as:

$$F_1 = \frac{2 * P * R}{P + R}$$

Since data are generated randomly, for each parameter setting we run 100 times, and then take average as our final result in order to get reliable evaluation.

5.2 Results and Discussion

We first compare two-phase algorithm with PC algorithm and TPDA on linear Gaussian data. Results are illustrated by figure 3. We can see that two-phase algorithm and TPDA have shorter running time than PC algorithm, especially when underlying networks contain large number of variables. Running time of PC algorithm grows dramatically with the sizes of networks, but two-phase algorithm and TPDA not. For the Munin network, PC algorithm fails to get any results within 24 hours. The reason is that monotone faithfulness assumption enables two-phase algorithm and TPDA to search the structure of Bayesian network with polynomial time complexity. For accuracy of algorithms, these three algorithms have almost same performance and two-phase algorithm performs slight better, since all of them are based on dependency analysis and should output same results if CI tests are perfectly accurate.

And then, we compare two-phase algorithm with LiNGAM algorithm to check whether partial correlation-based CI tests can deal with non-Gaussian data. We draw disturbances of linear SEMs from uniform distribution and log-normal distribution and results are shown by figure 4 and figure 5. We can see that two-phase algorithm performs better than LiNGAM algorithm in both accuracy and running time. The running time of LiNGAM grows fast as the sizes of networks increase. In addition, two-phase algorithm has almost same performance on data drawn from different distributions, which is a very nice property that enable us not to make assumption about data. However, LiNGAM algorithm does not perform stably on different distributions.

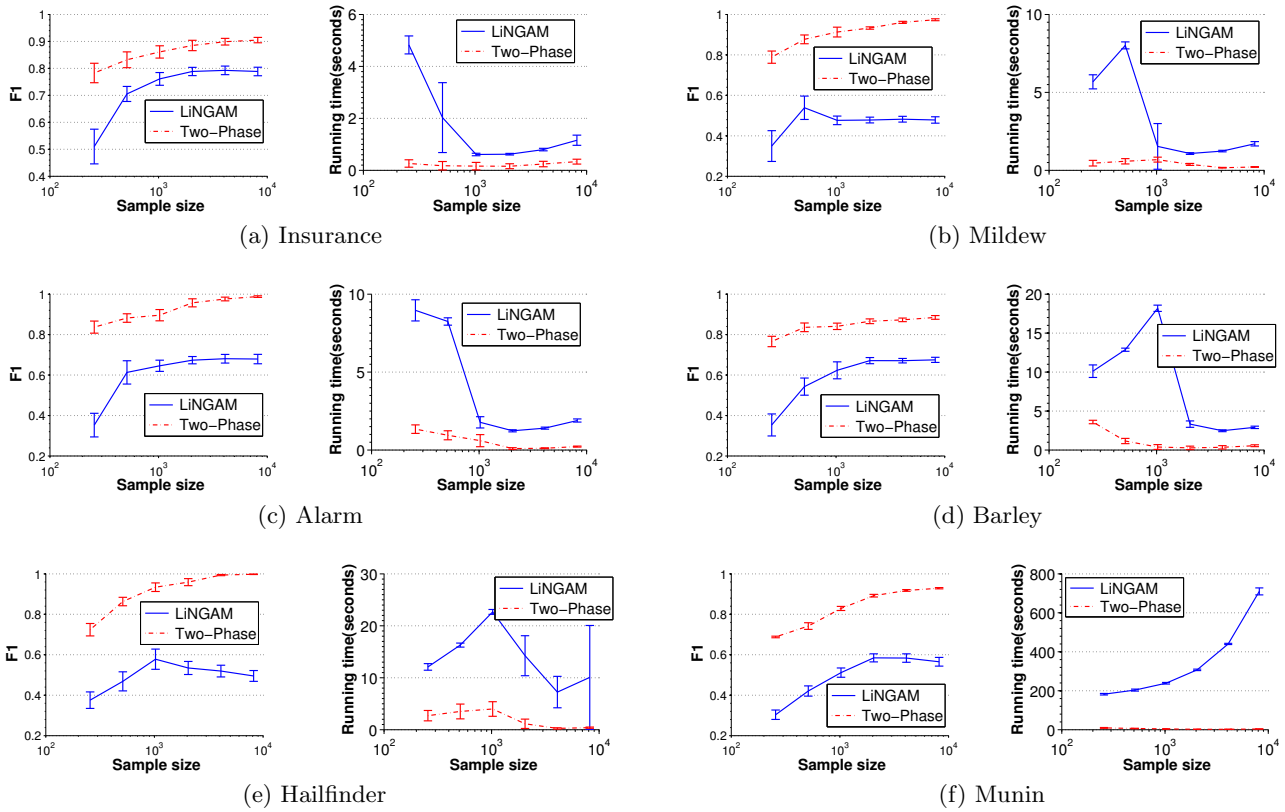


Figure 4: Compare two-phase algorithm with LiNGAM algorithm on uniformly distributed data

All results listed above provide an evidence that partial correlation-based CI tests can works well under monotone faithfulness assumption, since heuristic two-phase algorithm has almost same or better accuracy than PC algorithm which has a systematic search. Another property empirically illustrated by results here is that partial correlation-based CI tests can deal with linear non-Gaussian data as we discussed in section 3.1.

6. CONCLUDING REMARKS

In this paper, we propose a two-phase algorithm to discover causal relations on continuous data generated by linear SEMs. Compared with existing algorithms, two-phase algorithm has following advantages: (i) it can deal with continuous data with arbitrary distributions uniformly rather than by combining two separate components to deal with Gaussian and non-Gaussian cases respectively; (ii) it has polynomial running time, so that it can be applied to learning large networks in practice; (iii) it is distribution-insensitive as long as data generated by linear SEMs; (iv) it is demonstrated equaling or outperforming the existing methods in terms of accuracy on a series of real world Bayesian networks.

Furthermore, we show that partial correlation-based CI tests have two nice properties: (i) it can deal with both Gaussian and non-Gaussian cases; (ii) it works well under monotone faithfulness assumption. The first property makes it possible to apply those algorithms [17, 16, 12, 11, 14] with partial correlation-based CI tests to non-Gaussian data. With the second property, we can construct more efficient heuristic search algorithms like two-phase algorithm with correlation-based CI tests.

7. ACKNOWLEDGMENTS

The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administration Region, China.

8. REFERENCES

[1] S. Andreassen, A. Rosenfalck, B. Falck, K. G. Olesen, and S. K. Andersen. Evaluation of the diagnostic performance of the expert emg assistant munin. *Electroencephalography and Clinical Neurophysiology/Electromyography and Motor Control*, 101(2):129 – 144, 1996.

[2] K. Baba, R. Shibata, and M. Sibuya. Partial correlation and conditional correlation as measures of conditional independence. *Australian & New Zealand Journal of Statistics*, 46(4):657–664, December 2004.

[3] I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. In *Second European Conf. on Artif. Intell. in Medicine*, volume 38, pages 247–256, London, Great Britain, 1989.

[4] J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2):213 – 244, 1997.

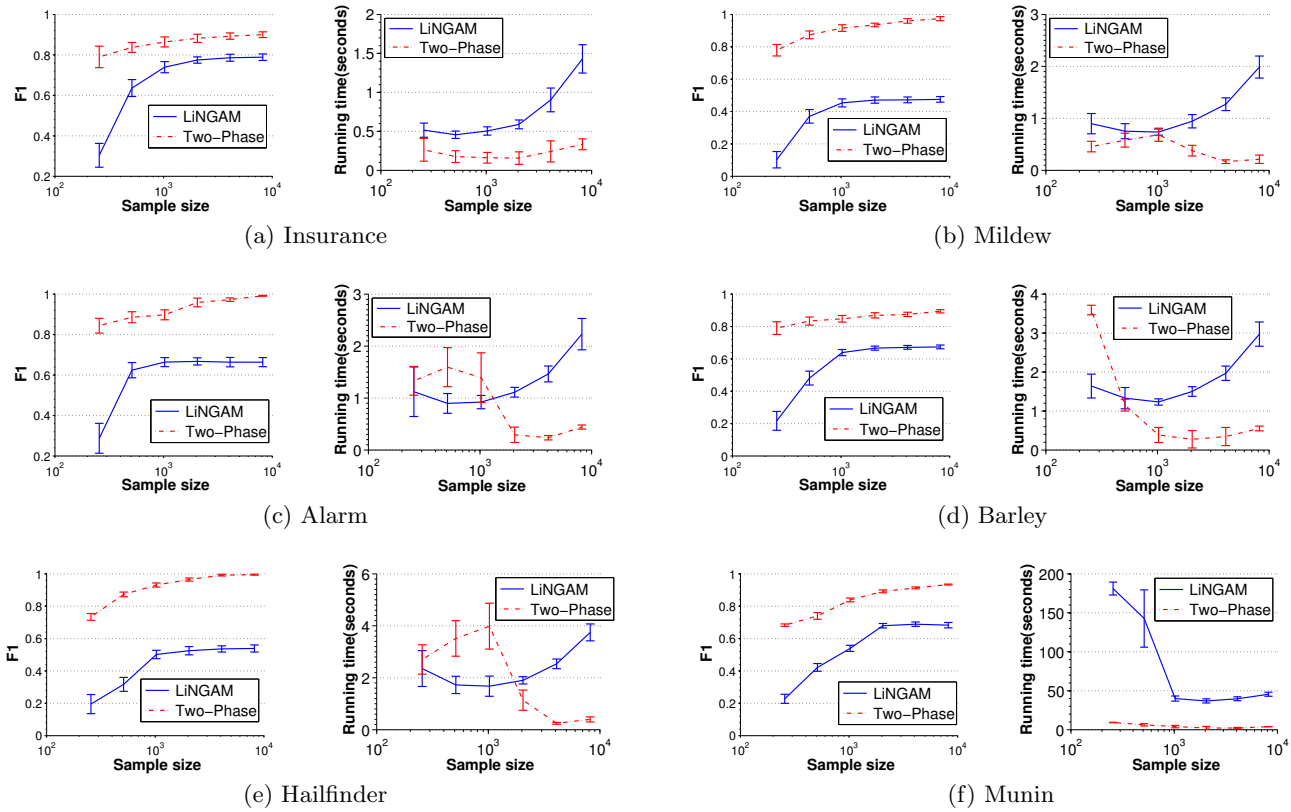


Figure 5: Compare two-phase algorithm with LiNGAM algorithm on log-normal data

[5] J. Cheng, R. Greiner, J. Kelly, D. A. Bell, and W. Liu. Learning bayesian networks from data: An information-theory based approach. *Artif. Intell.*, 137(1-2):43–90, 2002.

[6] P. Hoyer, A. Hyvärinen, R. Scheines, P. Spirtes, J. Ramsey, G. Lacerda, and S. Shimizu. Causal discovery of linear acyclic models with arbitrary distributions. In *Proc. 24th Conf. on Uncertainty in Artif. Intell. (UAI-08)*, pages 282–289, Corvallis, Oregon, 2008. AUAI Press.

[7] A. Hyvärinen, S. Shimizu, and P. Hoyer. Causal modelling combining instantaneous and lagged effects: an identifiable model based on non-gaussianity. In *Proc. of the 25th Int. Conf. on Mach. learn.*, pages 424–431, Helsinki, Finland, 2008. ACM.

[8] A. L. Jensen and F. V. Jensen. Midas: An influence diagram for management of mildew in winter wheat. In *Proc. of the Twelfth Annual Conf. on Uncertainty in Artif. Intell.*, pages 349–356, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers.

[9] R. Kindermann and J. L. Snell. *Markov Random Fields and Their Applications*. American Mathematical Society, 1980.

[10] K. Kristensen and I. A. Rasmussen. The use of a bayesian network in the design of a decision support system for growing malting barley without use of pesticides. *Computers and Electronics in Agriculture*, 33(3):197 – 217, 2002.

[11] R. Opgen-Rhein and K. Strimmer. From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Systems Biology*, 1(37):334–353, 2007.

[12] J. Pearl. *Causality : Models, Reasoning, and Inference*. Cambridge University Press, March 2000.

[13] J. Pearl and T. Verma. A theory of inferred causation. In *Proc. of the Second Int. Conf. on Principles of Knowledge Representation and Reasoning*, 1991.

[14] J.-P. Pellet and A. Elisseeff. A partial correlation-based algorithm for causal structure discovery with continuous variables. In *IDA*, pages 229–239, 2007.

[15] S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-gaussian acyclic model for causal discovery. *J. Mach. Learn. Res.*, 7:2003–2030, 2006.

[16] P. Spirtes and C. Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(1):62–72, October 1991.

[17] P. Spirtes, C. Glymour, and R. Scheines. From probability to causality. In *Proc. of Advanced Computing for the Social Sciences*, 1990.

[18] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer Verlag, Berlin, 1993.

[19] Z. Wang and L. Chan. A heuristic partial correlation-based algorithm for causal relationship discovery. In *Intell. Data Engineering and Automated Learning - IDEAL 2009*, pages 234 – 241, 2009.