Combining Stage Algorithm for Discovering Causal Models

Takashi Isozaki
Sony Computer Science Laboratories, Inc.
Tokyo, Japan
isozaki@csl.sony.co.jp

Abstract

Many constraint-based algorithms for causal discovery have suffered from statistical errors in conditional independence (CI) tests. Those errors are often inevitable in real data because of the limitation of statistical power due to finiteness of data or violations of an assumption called faithfulness or stability condition. We propose a constraint-based algorithm that can reduce avoidable CI tests with combining an adjacency identification stage with an orientation stage and then provide accurate and fast inference without loss of theoretical correctness, which we call the Combining Stage (CS) algorithm. We also, in the algorithm, introduce *unreliable direction*, which can reduce orientation errors due to locality of CI tests. Simulations are provided to demonstrate the prominent performance of the algorithm by comparison to often referred algorithms: PC, Three Phase Dependency Analysis, Sparse Candidate, and Max-Min Hill-Climbing algorithms.

1 INTRODUCTION

Discovering causal models from observational data has been an important issue in many domains of science and engineering including AI, social sciences, medicine, and bioinformatics besides statistics. We usually rely on statistical methods for this purpose because we cannot generally, in many variables, know deterministic cause-effect relationships. Directed acyclic graphs (DAGs) are suitable for representing statistical causal relationships between variables of interest.

Many algorithms for causal discovery have been proposed, which are generally categorized to the following three approaches. Those are constraint-based (Pearl, 2000; Spirtes et al., 2000), score-and-search (Cooper and Herskovits, 1992; Heckerman et al., 1999), and their hybrid (Tsamardinos et al., 2006). The constraint-based (CB) approach is a promising one because the CB methods are easily applicable to both discrete and continuous variable systems, relatively fast, and can find latent causes (Spirtes et al., 2000). In addition, we consider that identifying v-structures, which is a characteristic of the CB methods, associates some independence relationships with causality (e.g., discussed by Reichenbach (1956)). We address the CB approach for causal discovery from these points of view in this paper.

In the CB methods, causal structures are inferred from constraints derived by statistical tests of independence between variables combined with causal inference rules (Pearl, 2000; Spirtes et al., 2000). The algorithms based on the approach construct a (partial) DAG in two consecutive stages. First is identifying an undirected graph in which the edges represent associations between variables. Perfectly performing this task requires exponentially growing numbers of conditional independence (CI) tests with increasing numbers of variables, so it is then practically intractable. The PC algorithm (Spirtes et al., 2000) reduces the number of tests by employing ascending order in the number of conditioning sets while maintaining correctness if no statistical errors occur. The second stage in the approach is directing edges by finding *v-structures* and inductions rules (Pearl, 2000; Spirtes et al., 2000). However, the CB methods are generally unstable because even a small number of statistical errors causes other errors in both stages and finally yields large errors. This problem is partially attributed to poor estimation because of data-size-shortage in large conditioning variable sets. The Recursive Autonomy Identification (RAI) algorithm reduces CI tests with such large conditioning sets by decomposing a graph to substructures (Yehezkel and Lerner, 2009).

However, there remains another issue of statistical errors in this approach. That is what we call the

statistical unfaithfulness. The CB approach is based on the assumptions of the causal faithfulness condition (CFC) in addition to the causal Markov condition (Spirtes et al., 2000), which relate DAG representations to conditional independencies. For relaxing the CFC assumption, Ramsey et al. (2006) decomposed the faithfulness condition into adjacency-faithfulness and orientation-faithfulness, and assumed that only the former was valid and then improved the PC algorithm. However, both assumptions, unfortunately, often seem to be violated *statistically* given finite data, even though the hidden true distributions are *probabilistically* faithful, which is almost generally valid shown by Spirtes et al. (2000) and Meek (1995b). This is because small dependencies may not be recognized by statistical tests due to finiteness of sample size. As a result, too many edges are likely to be removed because, in the usual manner, edges remain only when CI hypotheses are rejected.

Therefore, developments of algorithms that reduce unnecessary CI tests are still needed for performing accurate causal discovery. For this purpose, we propose a CB algorithm reducing avoidable CI tests without loss of correctness. The algorithm uses *edge-direction information* in the adjacency-identification stage, which few algorithms have effectively used in the way we mean as far as we know. Furthermore, we introduce a concept of *unreliable direction* of edges for reducing the successive errors in the inductive orientation stage by the orientation rules given by Verma and Pearl (1992) and Meek (1995a). The procedure and concept would also be expected to be used or embedded in other CB algorithms developed in the future.

This paper is organized as follows: the next section provides some preliminaries for probabilistic graphical models for causal structures and causal discovery algorithms. A new algorithm is described in detail in Section 3. Its practical demonstrations are provided in Section 4, where some comparisons with prototypical algorithms are performed. We discuss related work in Section 5 and summarize our work in Section 6.

2 PRELIMINARIES

Let $\mathbb{V}$ be some set of random variables and $\mathbb{E}$ be a set of edges each of which connects a pair of distinct elements of $\mathbb{V}$. Each edge can be either directed or undirected. Each variable of $\mathbb{V}$ in a graph is also called a node. Two nodes X and Y are adjacent if an edge exists between the two. Let P be a strictly positive joint probability distribution of random variables in $\mathbb{V}$ and $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ be a directed acyclic graph (DAG) that has no directed cycles. We denote the conditional independence of variables X and Y given sets $\boldsymbol{Z}$ for a distribution P with $Ind(X; Y|\boldsymbol{Z})$ or $X \perp\!\!\!\perp Y \mid \boldsymbol{Z}$. In the case that $\boldsymbol{Z} = \emptyset$ for the conditional independence, we call that a (marginal) independence and often denote it concisely as $Ind(X; Y)$ or $X \perp\!\!\!\perp Y$.

A *path* in a graph is a sequences of edges. If a directed edge starting from Y enters X, Y is called a parent of X and X is called a child of Y. X is called a descendant of Y and Y is called an ancestor of X if there is a directed path from Y to X and X is called a non-descendant of Y if X is not a descendant of Y.

Causal Markov condition is defined as follows. In a DAG $\mathbb{G}$, where directed edges represent direct effect from the node to its children, every variable is independent of its non-descendants in $\mathbb{G}$ conditional on its parents in $\mathbb{G}$. Such DAGs are also called causal DAGs. Many causal discovery algorithms assume the converse principle called the **causal faithfulness condition** or stability condition (hereinafter called faithfulness): a causal DAG satisfies the causal Markov condition and all conditional independencies are entailed by the DAG. We assume in this work DAGs have no latent common causes and selection biases, that is, we assume causal sufficiency systems (Spirtes et al., 2000).

A term *collider* is defined as a node Z into which two edges enter, such as $X \to Z \leftarrow Y$. Otherwise Z is called a *non-collider*. If X and Y are not adjacent in the structure $X \to Z \leftarrow Y$, it is also called a *v-structure* and Z in it is called *unshielded collider*, otherwise, Z is called *shielded collider*. Conditional independence relations connected by the Markov condition are represented in a DAG by the following criterion called d-separation (Pearl, 2000).

Definition 1 (d-separation). *In a DAG, a path u between nodes X and Y is* **blocked** *by a set of nodes $\boldsymbol{Z}$ ($X, Y \notin \boldsymbol{Z}$) if (i) u contains a non-collider v such that v is in $\boldsymbol{Z}$, or (ii) u contains a collider v such that v is not in $\boldsymbol{Z}$ and such that no descendant of v is in $\boldsymbol{Z}$. A set $\boldsymbol{Z}$ is said to* **d-separate** *X from Y if and only if $\boldsymbol{Z}$ blocks every path from a node in X to a node in Y, which is denoted as $Dsep(X; Y|\boldsymbol{Z})$.*

For a set $(\mathbb{G}, P)$, called a Bayesian network (BN), $Dsep(X; Y|\boldsymbol{Z}) \Rightarrow Ind(X; Y|\boldsymbol{Z})$, and in a faithful BN, $Dsep(X; Y|\boldsymbol{Z}) \Leftrightarrow Ind(X; Y|\boldsymbol{Z})$ (Pearl, 1988). We call the set $\boldsymbol{Z}$ that satisfies $Ind(X; Y|\boldsymbol{Z})$ *separator set* under the faithfulness condition.

A triplet $\langle X, Z, Y \rangle$ in a DAG is called an *unshielded triple* if both X and Y are adjacent to Z but not each other. The following facts are used in adjacency-stage and orientation-stage of many CB algorithms including ours:[1]

[1] see e.g. Neapolitan (2004), pp.89 for proofs

Proposition 1 *In a DAG, two nodes are adjacent if and only if they are not d-separated by any subset of other nodes in the DAG.*

Proposition 2 *For any unshielded triple $\langle X, Z, Y \rangle$ in a DAG, the following are equivalent:*

- *The triple $\langle X, Z, Y \rangle$ has a collider Z.*
- *There exists a set not containing Z that d-separate X and Y.*
- *No sets containing Z d-separate X and Y.*

The following are often the basic procedures of the CB algorithms: (i) Form the complete undirected graph; (ii) for each pair of variables X and Y that are adjacent in the current graph, find a separator set $\boldsymbol{S}$ satisfying $Ind(X; Y|\boldsymbol{S})$. Once this is done, remove the undirected edge between X and Y, and record $\boldsymbol{S}$ as $Sepset(X, Y)$; (iii) for each unshielded triple $\langle X, Z, Y \rangle$ in the resultant graph, orient the edges as $X \rightarrow Z \leftarrow Y$ iff Z is not in $Sepset(X, Y)$; and (iv) execute the orientation rules (Verma and Pearl, 1992; Meek, 1995a) as follows until no more directions increase: (1) if X and Z are not adjacent and $X \rightarrow Y - Z$, then $X \rightarrow Y \rightarrow Z$ because no more v-structures are allowed; (2) if X and Y are adjacent and there is another directed path from X to Y, then $X \rightarrow Y$ because of prohibiting any directed cycles; and (3) if $X \rightarrow Z \leftarrow Y, X - W - Y, Z - W$, and X and Y are not adjacent, then $W \rightarrow Z$ because both new v-structures and directed cycles are not allowed. We call step (ii) adjacency-stage and steps (iii) and (iv) orientation-stage. However, the size of the potential separator sets in search space increases exponentially as the number of variables increases for the full search.

The PC algorithm (Spirtes and Glymour, 1991) searches with two tricks for recovering causal structures tractability: (1) it searches the separator sets in ascending order of the size ($|\boldsymbol{S}|$) starting from 0 (i.e., $\boldsymbol{S} = \emptyset$); and (2) it reduces the search space for the separator sets with the potential parents of the interested two variables: the currently adjacent nodes of the two. For larger conditioning sets for finding separator sets, the higher order statistics are needed, but then the error-probability increases for such sets. Therefore, the former trick (1) is very effective for the CB approach, and will also be critical in our algorithm. The latter (2), however, has still some problems: unnecessary CI tests are often executed, where statistical errors can occur. In practical use, these kinds of errors, in the CB algorithms such as the PC, are often observed. However, recognizing and avoiding unnecessary CI tests is a hard task because we cannot know the true causal structures in the search process.

3 COMBINING STAGE ALGORITHM

3.1 COMBINING STAGES

We consider the kinds of errors discussed in the previous section are related to the current structural identification manner in the CB approach: the CI hypotheses can be rejected but not verified by the tests, and edges remain only when the hypothesis is rejected, otherwise they are removed. This procedure can then tend to remove more edges than those in true DAGs because small dependencies may not be recognized by statistical tests due to finiteness of sample size even if the true probability distributions are faithful. We call such phenomena of statistically difficult-to-avoid errors *statistical unfaithfulness*. The errors are obviously related to both the adjacency and orientation identification, following on from Propositions 1 and 2. Thereby reducing avoidable unnecessary CI tests still remains one of the most critical issues in the CB causal discovery algorithms. We thus propose an algorithm for this purpose that combines adjacency and orientation stages. We call the algorithm Combining Stage (CS).

We first present some definitions and describe a condition with them for avoiding as many CI tests as possible.

Definition 2 *In a DAG, a* **blocker** *is defined as a node or the empty set that blocks a path between nodes X and Y. A* **minimal blocker** *is the node or the empty set that can block the path and has the minimal size of blockers of the path.*

Example 1 *In a DAG, for a path such that $X \rightarrow Z \leftarrow S \leftarrow Y$, $\emptyset$ and S are blockers and $\emptyset$ is the minimal blocker of the path.*

We notice the following fact about blockers that in a DAG, any blocker of a path between X and Y is not a collider on the path. This fact is immediately proved by Definition 1 and Definition 2. In addition, we can say more for minimal blockers as the following property:

Property 1 *In a DAG, no minimal blocker is a node on a path between X and Y that includes colliders.*

Proof. *Suppose Z is the minimal blocker node on a path between X and Y. First, Z itself is not a collider on the path because of Definition 1 and 2. Second, if Z is on the path that includes colliders, Z is not minimal because a minimal blocker is the empty set according to the definition of blockers. This is inconsistent with the assumption of the minimality.* □

The CS algorithm assumes the faithfulness and follows the procedure of searching separator sets in ascending order of their size the same as the PC. The CS can thereby find the minimal blockers in CI tests processing. The following Theorem that assures correctness of the CS algorithm can thus be stated:

Theorem 1 *Assume that an algorithm for constructing a Bayesian network starts from a complete undirected graph. In searching separator sets* $\boldsymbol{S}$ *for a pair* $\langle X, Y\rangle$*, to remove the edge between them if* $Ind\,(X; Y|\boldsymbol{S})$ *is satisfied, with ascending order of* $|\boldsymbol{S}|$ *from* $|\boldsymbol{S}| = 0$*, if all nodes in a candidate of separator sets, denoted as* $\boldsymbol{S}$*, are only on currently existing paths including colliders between* X *and* Y*,* $\boldsymbol{S}$ *can be removed from the candidates under the faithfulness condition.*

Proof. *In the searching process for separator sets, if all of the currently existing paths between* X *and* Y *include colliders, and exist even in the true DAG, the empty set is the common minimal blocker because of the Property 1. The empty set should be found in the former CI tests in the ascending order search process. Therefore, the candidate set* $\boldsymbol{S}$ *such that* $|\boldsymbol{S}| > 0$ *in that case can be ignored as a separator set for* $\langle X, Y\rangle$*. If all such paths, which include a node* Z *of the candidate set* $\boldsymbol{S}$*, do not exist in the true DAG,* Z *can also be ignored as a node for the candidate of separator sets because each node of separator sets must be in the path between* X *and* Y*, and then the set that has* Z *cannot be a separator set for* $\langle X, Y\rangle$*.* □

We call this condition for blockers described in Theorem 1 the **Minimal Blocker Condition**. That is, a candidate of a separator set can be ignored as the separator set if any of its nodes is not satisfied with the Minimal Blocker Condition. However, those found separator sets are not necessarily a union of minimal blockers: e.g., if there are paths between X and Y such as $X \to S \leftarrow Z \to Y$ and $X \to S \to Y$, minimal blockers of each path are $\emptyset$ and S but the separator set is $\{Z, S\}$. We use the Minimal Blocker Condition (MBC) in the new algorithm.

Whether a CI test of X and Y has to be done depends on whether all the paths between X and Y are satisfied with the MBC, in the adjacency identification stage with ascending order of candidate separator sets. We then construct the CS algorithm as it works as follows. The algorithm first finds sets of $Ind\,(X; Y|\emptyset)$ and constructs v-structures and adds them to a set *Vstr*. In incremented size of candidates for separator sets, whether the CI tests are performed is decided using currently known v-structures and MBC. Then, only unavoidable CI tests are performed and new v-structures are constructed and added to *Vstr*. Those procedures continue until no more separator sets exist or stopping rules are satisfied. If any edge of the triple $\langle X, Z, Y\rangle$ in *Vstr* is removed in CI tests, the triple will also be removed from *Vstr*. We can expect to increase accuracy of causal discovery because of the reason discussed in the beginning of this section. Additionally, the CS algorithm also has the aspect of an efficient algorithm from a theoretical point of view because the performance efficiency of CB methods, generally speaking, mainly depends on the performed number of CI tests.

3.2 K-MINIMAL BLOCKER CONDITION

MBC can possess a parameter and MBC with the determined parameter is called **k-Minimal Blocker Condition (k-MBC)**. The algorithm, in k-th order statistics for CI tests, uses v-structures recognized by up to (k-1)-th order CI tests. In practical use of the algorithm, the v-structures recognized by higher order statistics are less reliable and MBCs with such v-structures are also less reliable. We can then determine, in advance, a maximal order k of v-structures used in MBCs. If we set such a value k, we put off constructing new v-structures recognized by higher order CI tests than k until the end of the adjacency identification stage. This parameter is expected to make the CS more robust for practical use. Ramsey et al. (2006) and Zhang and Spirtes (2008) decomposed faithfulness condition into adjacency-faithfulness and orientation-faithfulness. It can be said that the k-MBC assumes lower order orientation-faithfulness and may render the CS robust to some extent for weak violations of higher order adjacency-faithfulness.

3.3 DEPENDENCY MEASURES AND BI-DIRECTION RESOLUTIONS

The following two subsections describe procedures to make the algorithm more robust for statistical violations of faithfulness, which we consider as necessary in almost all CB approaches. A p-value can be obtained from the CI tests performed on the CS algorithm, and the null hypothesis is rejected if the p-value is less than a significant level. We regard the variable sets in the CI tests as conditional independent if they cannot be rejected, the same way previous researchers did. The p-value is used in the CS algorithm as a metric of the degree of dependency between nodes as follows. The smaller the p-value, the higher the (conditional) dependency between the related nodes. This use of p-values resolves this problem as follows.

There often emerge, in the CB approach, inconsistent structures due to locality of CI tests and statistical errors, we hence take a measure for it. For example, if $X \perp\!\!\!\perp Y \,|\, \boldsymbol{S_1}$, $X \not\perp\!\!\!\perp Z$ and $Y \not\perp\!\!\!\perp Z$ given any sets,

and $Z \notin \boldsymbol{S_1}$ are obtained, the relationships generate $X \to Z \leftarrow Y$, and if $Z \perp\!\!\!\perp W \,|\, \boldsymbol{S_2}$, $X \not\perp\!\!\!\perp Z$ and $X \not\perp\!\!\!\perp W$ given any sets, and $X \notin \boldsymbol{S_2}$ are obtained, they generate $Z \to X \leftarrow W$. That means $X \leftrightarrow Z$, that is, it is bi-directionally inconsistent. We resolve the inconsistency in the CS algorithm using p-values: among the inconsistency sets of v-structures, we select the set with the highest p-values, remove the others from v-structure sets, and disorient edges in the loser groups. This procedure is described in Algorithm 1, where we denote *Vstr* as the v-structure sets.

Algorithm 1 Resolve Inconsistency

1: **Input** *Vstr*
2: Pick up inconsistency v-structure sets in *Vstr*
3: Calculate p-values on independency of all the sets
4: Select the highest p-values set
5: Remove the other sets from *Vstr* and disorient their edges
6: **return** *Vstr*

3.4 UNRELIABLE DIRECTION

The CB approach, in addition, has another weak point, which is due to the locality of CI tests and statistical errors, the same as the case in the previous subsection. That is, this approach may have another contradictory CI test result. This may lead to other wrong orientations caused by the orientation rules, so the induced errors should be avoided. We define, for discussing this, *unreliable direction* as follows. In a DAG $\mathbb{G}$, if two v-structures ($X \to Z \leftarrow W$ and $Y \to Z \leftarrow W$) are obtained and a v-structure ($X \to Z \leftarrow Y$) cannot be allowed, both statistically from CI tests, the two directions ($X \to Z$ and $Y \to Z$) are called **unreliable directions**. An example of them is depicted in Figure 1. In (a), $X \perp\!\!\!\perp W$ with 0.30 of p-value, $W \perp\!\!\!\perp Y$ with 0.40 of p-value but $X \not\perp\!\!\!\perp Y$ with 10^{-5} of p-value. This contradictory result can happen due to locality of CI tests and is different from the issue related to the orientation-unfaithfulness tackled by the CPC algorithm.

The CS algorithm disorient those unreliable edges if found, as depicted in Figure 1 (b), for avoiding induced errors of direction in the successive full orientation-stage.

Pseudocode for the whole general version of the CS algorithm is listed in Algorithm 2, where ADJ_X denotes the set of adjacent nodes of a node X. In practical use, we can use k-MBCs to reduce additive errors due to statistical power shortage.

Algorithm 2 Combining Stage Algorithm

1: Form a complete undirected graph $\mathbb{G}$
2: Set $n = 0$
3: **repeat**
4: **for** each $X \in \mathbb{V}$ **do**
5: **for** each $Y \in \mathrm{ADJ}_X$ **do**
6: Prepare subsets $\boldsymbol{S} \subseteq \mathrm{ADJ}_X \backslash Y$ that have n variables
7: **for** each subset $\boldsymbol{S}$ **do**
8: Set $m = 0$
9: **for** each $Z \in \boldsymbol{S}$ **do**
10: **if** a path between X and Y containing Z does not contain a v-structure **then**
11: Set $m = 1$
12: break
13: **end if**
14: **end for**
15: **if** $m = 0$ **then**
16: continue
17: **else if** a subset $\boldsymbol{S}$ is found conditional on which X and Y are independent **then**
18: Remove the edge $X - Y$ from $\mathbb{G}$
19: Add $\boldsymbol{S}$ to *Sepset*(X, Y)
20: **end if**
21: **end for**
22: **end for**
23: **end for**
24: **for** each unshielded triple $\langle X, Z, Y \rangle$ in $\mathbb{G}$ **do**
25: Orient it as $X \to Z \leftarrow Y$ iff Z is not in *Sepset*(X, Y)
26: Add them to *Vstr*(X, Z, Y)
27: **end for**
28: Set $n = n + 1$
29: **until** $|\mathrm{ADJ}_X| \leq n$ for all $X \in \mathbb{V}$
30: *Resolve Inconsistency(Vstr)*
31: Find all unreliable directions and disorient those edges
32: Orient edges using the orientation rules
33: **return** a partially DAG $\mathbb{G}$

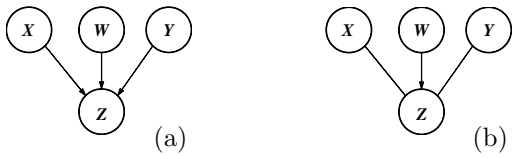


Figure 1: An example of *unreliable directions*: if CI tests derive $X \perp\!\!\!\perp W$, $Y \perp\!\!\!\perp W$, $X \not\perp\!\!\!\perp Y$ in addition X, Y, and W are adjacent to Z in (a), the CS algorithm disorient as $X - Z$ and $Y - Z$ depicted in (b).

4 EXPERIMENTS

We provide numerical simulations to evaluate the performance of the CS algorithm, which was implemented in C++ language. We also compare the CS algorithm with other often referred algorithms in terms of correctness of DAG structure recovery on discrete random variable systems, using known DAGs that are usually used in recent previous research.

4.1 EXPERIMENTAL SETUP AND RESULTS

There are three main approaches for DAG structure inference: the CB, score-and-search (SS), and their hybrid. We used the following algorithms, which are publicly available and were often used on recent researches for each approach (CB/SS/hybrid): PC (Spirtes et al., 2000), Three Phase Dependency Analysis (TPDA) (Cheng et al., 2002) (CB), Sparse Candidate (SC) (Friedman et al., 1999) (SS), and Max-Min Hll-Climbing (MMHC) (Tsamardinos et al., 2006) (hybrid). The SC and MMHC recently showed particularly high performances for the structure recovery tasks over a broad range of training sample sizes (Tsamardinos et al., 2006; Xie and Geng, 2008). We were aided by the Causal Explorer algorithm library (Aliferis et al., 2003) working on MatLab 7.8 platform for performing all algorithms other than the CS. The significant level of CI tests using the G^2 likelihood in PC and MMHC was 0.05. The threshold of CI tests using mutual information used in TPDA was 0.01. The score metric used in SC and MMHC was the BDeu (Heckerman et al., 1995) and its equivalent sample size was 10. For the SC algorithm, the maximum size of candidates was set to 10. All these settings of the metric and parameters were suggested by the authors.

We also used the G^2 tests on the CS to identify $Ind(X, Y|\boldsymbol{Z})$ with the significant level 0.01, which was decided by preliminary trials. The statistic is proven to be approximated asymptotically to a χ^2 distribution with some degrees of freedom (Kullback, 1968). Results of its approximation accuracy are wholly fixed on accuracy of the causal discovery in the CS. We used the CS with 0-MBCs, namely, with MBCs using v-structures derived from only marginal independence relationships (i.e., $Ind(X; Y|\emptyset)$), due to the reason mentioned in the previous section. Additionally, the CS algorithm selected nodes in ascending order of numbers of currently adjacent nodes in line 4 of Algorithm 2 and used heuristic 3 of the PC described in Spirtes et al. (2000, p.90) for speeding up.

We use the following five Bayesian networks (BNs), which have neither very many variables nor very many parameters because we want to evaluate algorithms excluding influences caused by the difference in used statistics as far as possible: Alarm with 37 nodes, 46 edges, 509 number of parameters (nps, for short), Carpo with 60 nodes, 74 edges, 342 nps, Hailfinder with 56 nodes, 66 edges, 2656 nps, Insurance with 27 nodes, 52 edges, 1008 nps, and Water with 32 nodes, 66 edges, 10083 nps. Extra information can be obtained through the Bayesian Network Repository[2], where structures and probability distributions of these networks are also available.

We sampled training cases from the distributions of the known networks. From each BN, 10 datasets were randomly sampled in each size of 1000, 2000, 5000, and 10000 because, in too short data, results may reflect the difference in statistics rather than goodness of algorithms, which we consider unfair and should be avoided as much as possible. Statistical quantities used here are the average over the 10 runs of an algorithm on each sample size from the identical distribution of the DAG.

The PC, TPDA, and CS algorithms generate a partially DAG (PDAG) as a Markov equivalence class (Verma and Pearl, 1990) (which is also called completed PDAG (Chickering, 1995)) of the recovered DAG, while the SC and MMHC algorithms provide a DAG. Then we transform a DAG that provided by the SC and MMHC to a completed PDAG using Chickering's algorithm (Chickering, 1995), as Tsamardinos et al. (2006) and Yehezkel and Lerner (2009) did. Tsamardinos et al. (2006) suggested evaluating the quality of recovered PDAGs using the structural Hamming distance (SHD) metric, while they pointed out that the KL divergence and the BDeu score were not always suitable for the purpose. Following Spirtes et al. (2000), Tsamardinos et al. (2006), and Yehezkel and Lerner (2009), we use five types of inference errors to evaluate recovered networks: extra edges (EE), missing edges (ME), extra direction (ED), missing direction (MD) and reversed direction (RD) errors. EE (ME) error represents extra (missing) edges that appear in the recovered (true) graph but not in the true (recovered) graph. ED (MD) error is due to edge directions that appear in the recovered (true) graph but not in the true (recovered) graph. We also use the total directional errors (DE), DE = ED + MD + RD. The SHD sums all five structural errors.

The simulation results are shown in Table 1. The numerical values are 10 run-averaged relative ones to that of the MMHC following the way of Tsamardinos et al. (2006) and Yehezkel and Lerner (2009), and the bold numbers denote the best scores in each network and sample size as the results of two-sided t-tests with the significant level 0.05. As found by recent researches, the hybrid (MMHC) method performed better than the PC and TPDA algorithms that are based on the CB method. The CS algorithm, which is also a CB approach, obviously shows superiority to the SC, PC, and TPDA for all networks and all sample sizes. In addition, the experiments show that for almost all sample

[2]http://www.cs.huji.ac.il/site/labs/compbio/Repository/

sizes of all networks, the CS algorithm is not outdone by the MMHC and is often superior to the MMHC. As far as we know, few constraint-based algorithms show such effectiveness comparable to the MMHC by using *averaged* SHD for *broad range* of sample sizes (see also Section 5).

In the results of the PC algorithm, SHDs are relatively worse than the CS, even for large sample sizes such as 5000 and 10000. It shows that our strategy of reducing CI tests via Minimal Blocker Condition and relaxing the locality-issues are effective for the structure identification.

Table 2 reports the number of CI tests performed by the CS and MMHC algorithms for a sample size of 10000 because the numbers were monotonically increasing for both algorithms and all networks. The CS performed relatively fewer CI tests *on average* and showed *more stable numbers*. Table 3 shows the normalized averages of running time on the both algorithms for the same sample size, where averaged times were normalized by smallest times in each algorithm in order to absorb the difference of programming languages in the algorithms. These timing results also show the CS ran *stably*. For the purpose of reference, the raw data of running time in seconds, without data input time, in the CS were also shown in the table. The data were obtained with the following running condition: 64 bit Windows 7 (OS), Intel Core i7 870, 2.93GHz (CPU), and 12.0GB (RAM). The results showed the CS to run sufficiently fast.

4.2 DISCUSSION

The following observations are worth noting. From Table 1, the relative SHD score of SC algorithm is often likely to become worse relative to the CS and MMHC as sample size increases. This tendency is more clearly seen in Figure 2 plotted for raw data of the SHD in Carpo and Insurance networks as examples for the SC, CS, and MMHC, where the CS and MMHC, in contrast, show decreasing numbers of errors denoted as SHD, as sample size increases relative to the SC: The decreasing number of SHD, from sample sizes of 1000 to 10000, were, in both CS and MMHC, about 25 for Carpo and about 13 for Insurance, while in SC being -11 for Carpo and 4.6 for Insurance. These large differences seem to imply that there were many false structures that had similarly good scores in large search spaces of DAG structures. In addition, it was shown empirically by Tsamardinos et al. (2006) with comparison between true and estimated structures that finding DAGs with higher BDeu scores did not always lead to more correct DAGs[3]. The MMHC algorithm initially selects parents and children sets by a CB approach (called MMPC algorithm) and then selects parents with the BDeu-score search. The MMHC and similar hybrid approaches thus also have the possibility of the same issues as are suspected to exist in the SC because a DAG with the better BDeu score but worse SHD score may exist even if the search space is reduced with CB approaches.

In performing experiments for comparing algorithms with well-known datasets such as Alarm network, we should separate effectiveness of algorithms and statistics because the BDeu can avoid over-fitting more than classical statistics such as G^2 and mutual information. Therefore, we might have compared effects of used statistics rather than of algorithms if we had used small samples or DAGs with large amounts of parameters. We thus consider that some other parameters, instead of sample sizes, may be needed for comparative study. The ratio of numbers of the parameter to sample sizes may be possible but it is not always preferable because there are often cases in which many conditional probabilities are exactly or nearly zero in the networks (e.g., Water network). Therefore, an effective number of parameters may need to be defined while taking into account entropy of each DAG.

5 RELATED WORK

Spirtes et al. (2000) discussed *undirected paths* required to contain separator sets for CI tests. Steck and Tresp (1999) also proposed the same condition. Our proposed constraint (i.e., MBCs) can thereby be an extension of the condition on undirected paths.

Two recursive decomposing algorithms of the CB approaches were recently proposed: the RAI (Yehezkel and Lerner, 2009) and the algorithm proposed by Xie and Geng (2008). The RAI algorithm may seem similar to the CS, at a glance, in using the orientation rules in the process of adjacency identification. However, the RAI uses the full orientation rules in order to divide a graph into sub-graphs, while the CS algorithm uses the direction-information to omit avoidable CI tests. In order to do this accurately, the CS can utilize the k-MBCs with v-structures identified by relatively lower order and then reliable CI tests and the whole orientation rules are not performed to the end of the adjacency stage. The iterated full use of the rules, as the RAI did, increases a risk of generating sequential errors. We consider our strategy partially contributes to good accuracy for causal inference in a

[3]This point should be elucidated by both more theoretical and experimental studies with better equivalent sample sizes.

Table 1: Averaged normalized results of the structural Hamming distance, which are normalized by dividing for a particular network and sample size (N) by the corresponding value for the MMHC algorithm. A normalized value smaller than one means fewer structural errors of the algorithm than the MMHC. The bold numbers denote the best scores as the results of two-sided t-tests with significant level 0.05.

Network	Algorithm	N=1000	N=2000	N=5000	N=10000
Alarm	CS	**1.2**	**1.2**	**0.7**	**0.5**
	PC	2.5	2.9	3.5	2.5
	TPDA	3.0	3.1	3.8	2.7
	SC	2.5	2.3	2.8	2.3
	MMHC	**1.0**	**1.0**	1.0	1.0
Carpo	CS	**0.9**	**0.9**	**0.8**	**0.8**
	PC	1.2	1.2	1.4	1.6
	TPDA	1.3	1.3	1.5	1.8
	SC	1.6	1.8	2.2	3.0
	MMHC	1.0	1.0	1.0	1.0
Hailfinder	CS	**1.2**	**0.9**	**0.9**	**1.0**
	PC	3.6	3.3	3.7	3.9
	TPDA	4.2	3.3	3.3	3.0
	SC	1.5	1.4	1.8	1.7
	MMHC	**1.0**	**1.0**	**1.0**	**1.0**
Insurance	CS	**1.0**	**1.1**	**0.9**	**1.0**
	PC	1.6	1.5	1.6	1.8
	TPDA	2.1	1.7	1.7	1.9
	SC	1.3	1.5	1.6	1.8
	MMHC	**1.0**	**1.0**	**1.0**	**1.0**
Water	CS	**1.0**	**1.0**	1.2	**0.9**
	PC	3.5	3.7	4.3	4.2
	TPDA	1.2	1.2	1.3	1.3
	SC	1.3	1.5	1.8	1.8
	MMHC	**1.0**	**1.0**	**1.0**	1.0

Table 2: Averaged number of conditional independence tests for 10000 of sample size in the CS and MMHC algorithms.

Algorithm	Alarm	Carpo	Hailfinder	Insurance	Water
CS	3.8K	5.5K	6.2K	4.9K	1.3K
MMHC	2.9K	19.5K	61.1K	5.9K	1.3K

relatively broad range of sample sizes, while the RAI showed the comparable accuracy with the MMHC only for 500 samples even with optimized thresholds by the BDeu scores in CI tests using mutual information.

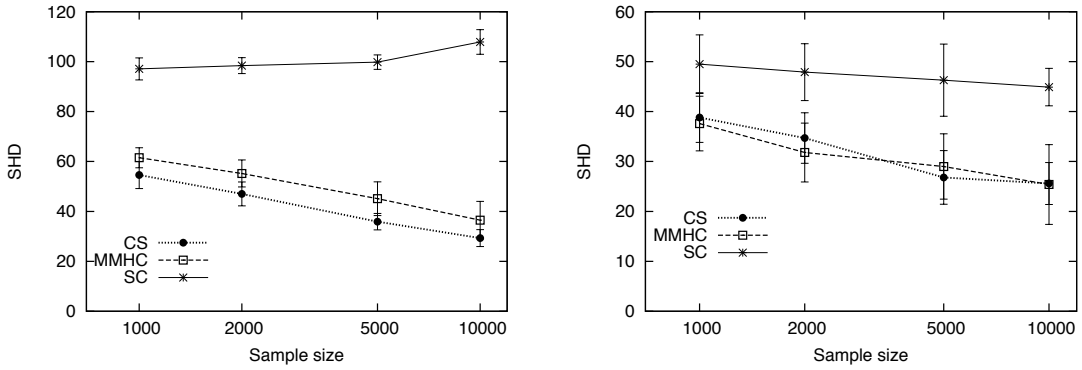
Another recursive method of Xie and Geng (2008) depends more on the assumption of correctness of the CI tests because an edge between X and Y is removed, in the method, if X and Y are independent conditionally on the set of *all other variables*. Accordingly, the method seems effective for parametric continuous variable systems because the number of parameters is generally much smaller than that of discrete variables. In fact, they showed the effectiveness in simulations on discrete variables only for better results not for averaged SHD scores or on randomized probability distributions.

6 CONCLUSION

We proposed, in this paper, Combining Stage (CS) algorithm for accurate causal discovery, which can reduce avoidable CI tests by using a Minimal Blocker Condition (MBC), introduced in this work, via partial orientation of edges. The MBC assures correctness

Table 3: Averaged normalized running time for 10000 of sample size in the CS and MMHC algorithms. These results show the CS ran stably in time. In the CS, the raw data are also shown for reference in terms of seconds (denoted as s), where the second values denote standard deviations.

Algorithm	Alarm	Carpo	Hailfinder	Insurance	Water
CS	3.0 (2.0±0.1s)	5.0 (3.3±0.3s)	8.0 (5.3±0.8s)	3.6 (2.4±0.1s)	1.0 (0.7±0.1s)
MMHC	1.8	11.7	123.1	4.2	1.0



(a) Carpo (b) Insurance

Figure 2: Performance of the CS, MMHC, and SC algorithms on the (a) Carpo and (b) Insurance networks. The absolute SHD scores are plotted against training sample size. Error bars denote the standard deviations.

of the CS under the causal faithfulness assumption. In addition, we provide procedures to resolve general problems in the constraint-based approaches due to locality of CI hypothesis tests and their limited statistical power: one is to do with p-values for bi-directional inconsistencies and the other is to do with introducing *unreliable directions* against inconsistencies related to some sets of v-structures. All those procedures introduced in the CS algorithm are expected to be usable for other constraint-based algorithms developed in the future. The algorithm practically works better, in a comparative study using some repository datasets with 1000–10000 samples, than the current prominent algorithms (PC, TPDA, and SC) and is the same or more than the MMHC, which is one of state-of-the-art algorithms. In addition, the number of CI tests was stably reduced and fewer than the MMHC on average, and then the CS ran fast stably. The CS is thus expected to be applicable to large DAGs. We plan to apply the algorithm to latent variable models with extensions and continuous non-linear systems using kernel-based CI testing (e.g. Zhang et al. (2011)).

Acknowledgements

The author thanks Manabu Kuroki for helpful discussions and Ryosuke Ohata for his assistance in the experiments. The author also thanks Mario Tokoro and Hiroaki Kitano of Sony Computer Science Laboratories, Inc. for their support.

References

Aliferis, C. F., Tsamardinos, I., Statnikov, A., and Brown, L. E. (2003). Causal Explorer: A causal probabilistic network learning toolkit for biomedical discovery. In *Proc. of International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS-03)*, pages 371–376.

Cheng, J., Greiner, R., Kelly, J., Bell, D., and Liu, W. (2002). Learning Bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, 137(1-2):43–90.

Chickering, D. M. (1995). A transformational characterization of equivalent Bayesian network structures.

In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 87–98.

Cooper, G. and Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347.

Friedman, N., Nachman, I., and Peér, D. (1999). Learning Bayesian network structure from massive datasets: The "Sparse Candidate" algorithm. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 206–215.

Heckerman, D., Geiger, D., and Chickering, D. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243.

Heckerman, D., Meek, C., and Cooper, G. (1999). A Bayesian approach to causal discovery. In Glymour, C. and Cooper, G. F., editors, *Computation, Causation, and Discovery*, pages 141–165. AAAI/MIT Press, Cambridge, MA.

Kullback, S. (1968). *Information Theory and Statistics*. Dover Publications, Mineola, NY.

Meek, C. (1995a). Causal inference and causal explanation with background knowledge. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 403–410.

Meek, C. (1995b). Strong completeness and faithfulness in Bayesian networks. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 411–418.

Neapolitan, R. E. (2004). *Learning Bayesian Networks*. Prentice Hall, Upper Saddle River, NJ.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA.

Pearl, J. (2000). *Causality, models, reasoning, and inference*. Cambridge University Press, New York, NY.

Ramsey, J., Spirtes, P., and Zhang, J. (2006). Adjacency-faithfulness and conservative causal inference. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-06)*, pages 401–408.

Reichenbach, H. (1956). *The Direction of Time*. Dover Publications, Mineola, NY. Republication of the work published by University of California Press, Berkely.

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

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

Steck, H. and Tresp, V. (1999). Bayesian belief networks for data mining. In *Workshop Data Mining und Data Warehousing als Grundlage Moderner Entscheidungsunterst´utzender Systeme (DMDW-99)*, pages 145–154.

Tsamardinos, I., Brown, L. E., and Aliferis, C. F. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78.

Verma, T. and Pearl, J. (1990). Equivalence and synthesis of causal models. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-90)*, pages 220–227.

Verma, T. and Pearl, J. (1992). An algorithm for deciding if a set of observed independencies has a causal explanation. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-92)*, pages 323–330.

Xie, X. and Geng, Z. (2008). A recursive method for structural learning of directed acyclic graphs. *Journal of Machine Learning Research*, 9:459–483.

Yehezkel, R. and Lerner, B. (2009). Bayesian network structure learning by recursive autonomy identification. *Journal of Machine Learning Research*, 10:1527–1570.

Zhang, J. and Spirtes, P. (2008). Detection of unfaithfulness and robust causal inference. *Minds and Machines*, 18(2):239–271.

Zhang, K., Peters, J., Janzing, D., and Scholköpf, B. (2011). Kernel-based conditional independence test and application in causal discovery. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI-11)*, pages 804–813.