A Hybrid Algorithm for Learning Causal Networks using Uncertain Experts' Knowledge

Christophe Gonzales christophe.gonzales@lis-lab.fr
Aix Marseille Univ, Université de Toulon,
CNRS, LIS,
Marseille, France

Axel Journe axel.journe@engie.com
CSAI Engie Lab
Stains, France

Ahmed Mabrouk ahmed.mabrouk@engie.com
CSAI Engie Lab
Stains, France

Abstract

Bayesian networks (BN) have become one of the most popular frameworks in causal studies. The causal relations between variables are encoded by the structure of the model, which is a directed acyclic graph (DAG). Unfortunately, despite the significant advances in algorithm development, learning the causal structure from data remains a very challenging task, especially for cases with a large number of variables. When the learning algorithm fails to identify the causal orientation of some edges, the human expert can provide some rough guidelines to complete the causal discovery. In many application domains, the expert knowledge might be uncertain about the right orientation of the edge. Worst, it may contradict the orientations learned from observational data, hence leading to conflicting situations. This paper presents a new hybrid algorithm combining a constraint-based approach with a greedy search, that includes specific rules to cope with uncertain domain/expert knowledge at different steps of the learning process. Experiments show the robustness of our method compared to other state-of-the-art algorithms.

Keywords: Causal Bayesian Network; Expert's Knowledge; Constraint-Based Algorithm; Score-Based algorithm; Uncertainty, Interactive Algorithm.

1. Introduction

Causal study has received a lot of attention in many practical domains. An accurate understanding of a given system often requires information about cause-and-effect relationships between variables. Causation is not merely a specific field of statistics. It can be considered as an enrichment of traditional statistics allowing to answer intervention queries of the form *"if we intervene to fix the value of X, what effect will it have on Z"*. Such queries arise naturally in a variety of settings such as prediction, decision-making, scientific discovery, *etc.* In this context, causal Bayesian networks (CBN) represent one of the most powerful framework (Pearl (2009)).A CBN has the same representation as a standard Bayesian network (BN): a directed acyclic graph (DAG) over random variables combined with a set of conditional probability distributions. But, unlike classical BNs, each variable in the CBN is characterized by a causal mechanism that determines its states based on the values of its

parents. In other words, the directions of the arrows in a CBN are meaningful, i.e., they indicate causal relations (either directly or indirectly) between variables. As a consequence, a CBN tends to be sparser and it is easier to interpret by experts than a BN.

The framework for answering intervention queries requires a fully specified CBN. For this purpose, many algorithms have been proposed in the literature to learn causal models from data. The class of constraint-based algorithms (Spirtes and Glymour (1991); Colombo and Maathuis (2014); Li et al. (2019); Ramsey et al. (2006)) consists of exploiting statistical tests to uncover the independence properties of the probability distribution that generated the data, thereby learning the graphical structure of the causal model. These algorithms have been generalized to learn causal models in the presence of latent variables (Spirtes et al. (2000); Colombo et al. (2012)). They can even, at least partially, detect the existence of these variables and their locations in the DAG. Unfortunately, all these methods have two major drawbacks: i) they rely on restrictive assumptions; and ii) they are fairly sensitive to the errors made by the statistical tests. Score-based approaches (Chickering (2003)) can be also used to make the causal discovery and are more robust than constraint-based algorithms. Instead of using statistical tests to detect independences among random variables, they use scores, essentially encoding likelihoods of the structures given the data, to determine the most likely DAG. Then, the Markov equivalence class of such DAG provides an insight into a subset of causal relationships: its arcs represent causal relations and its edges provide some causal knowledge, although the causal direction remains unknown. This explains why many causal relations cannot be identified by these approaches. In the literature, inference algorithms have also been used for causal discovery (Gretton et al. (2009); Peters et al. (2010); Janzing and Schölkopf (2010)). These approaches rely on constraint-based algorithms together with local searches for additive noise models in non-oriented substructures. Although these approaches are effective, the restrictive use of causal inference rules makes them somewhat brittle, especially when the dataset is too or not enough noisy. In addition, by considering only one pair of variables at a time, they may run the risk of learning of fully directed graph even in the presence of latent variables.

Fully constructing a causal model is therefore not straightforward. In most cases, state-of-the-art learning algorithms fail to identify the causality (causal direction) for several edges in the graph. This issue might be due to some causal relations being theoretically unidentifiable, but also to the low quality of observational data that include some noise due to measurement errors, or even to the presence of latent variables. A common approach to address the causal orientation issues consists of using interventional data. The latter is a record of phenomena manipulated by external agents, i.e, the values of some variables in the studied system are fixed by an external force, most often by a human expert. Unfortunately, obtaining interventional data is not trivial and can be problematic for multiple reasons (economical, technical, ethical, *etc.*).

To cope with this problem, an alternative approach consists of exploiting experts' knowledge to optimize the causal discovery. This option received a lot of attention in the early days of BNs because experts tend to encode their own domain understanding in a causal way. In many situations, the expert knowledge about the domain may be partial and the expert is uncertain about the right orientation of the edges. Using hard structural constraints in such circumstances may lead to nonsensical models. In (de Campos and Castellano (2007); Borboudakis and Tsamardinos (2012)), the main goal is the estimation of a model where

edge orientations are consistent with causal prior knowledge. By doing so, these methods may lead to a very poor causal approximation, notably in the presence of very uncertain or erroneous causal prior. Exploiting uncertain knowledge during model learning was tackled in (de Campos and Ji (2011), Amirkhani et al. (2017), Xu et al. (2015)). Unfortunately, in these methods, only very few orientations in the directed graph can be considered causal (those around v-structures). In this paper, we are interested in exploiting uncertain expert's knowledge when learning a causal model. We propose a three-phase learning algorithm. The first one aims to learn a first model using a constraint-based approach and, in the second step, we use a score-based approach to refine this initial partial model in order to obtain an optimal Markov equivalence class. In both steps, expert's knowledge, either certain or uncertain, is taken into consideration. Finally, we develop an interactive algorithm where causal relations are elicited from an expert, asking as few questions as possible. The rest of the paper is organized as follows. Section 2 recalls the definitions and algorithms needed to explain our approach. In section 3, we describe our approach to learn causal models and justify its correctness. The effectiveness of our method is highlighted through experiments in Section 4. Finally, some concluding remarks are given in Section 5.

2. Background and Related Works

BNs and CBNs rely on graphs to represent relations between random variables. Let us first provide some definitions about these graphs. A directed graph $G = (\mathbf{V}, \mathbf{E})$ is defined by a set of vertices $\mathbf{V}$ and a set of arcs $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$. We represent an arc $(A, B) \in \mathbf{E}$ as $A \to B$. If there exists an arc $A \to B$, A is called the parent of B and B is the child of A. A directed path $(X_1, \ldots, X_n)$ is a sequence of vertices such that, for all $i < n$, arc $X_i \to X_{i+1}$ belongs to $\mathbf{E}$, and a directed cycle is a directed path in which $X_n = X_1$. A directed acyclic graph (DAG) is a directed graph in which there exists no directed cycle. We can now define BNs:

Definition 1 (Bayesian network) *A BN is a pair $(G, \mathbf{\Theta})$ where $G = (\mathbf{V}, \mathbf{E})$ is a DAG, $\mathbf{V}$ represents a set of random variables*[1] *and $\mathbf{\Theta} = \{P(X|Pa(X))\}_{X \in \mathbf{V}}$ is the set of the conditional probability distributions (CPD) of the random variables X in G given their parents $Pa(X)$ in G. The BN encodes the joint probability over $\mathbf{V}$ as $P(\mathbf{V}) = \prod_{X \in \mathbf{V}} P(X|Pa(X))$.*

A CBN is exactly the same as a BN except for the meaning of the arcs in its graphical structure: in a CBN, Arc $A \to B$ means that A is a potential cause of B whereas, in a BN, it just represents a potential probabilistic dependence between A and B. Causality implies dependence but not the converse because causality is an asymmetric concept. The CBN structure learning algorithms therefore rely on those learning BN structures, but they require an additional care for enforcing the causal orientations of the arcs.

Let us now review some concepts exploited by these algorithms. Undirected graphs $G = (\mathbf{V}, \mathbf{E})$ (a.k.a. *skeletons*) are such that $(A, B) \in \mathbf{E}$ implies that $(B, A) \in \mathbf{E}$. Such pairs of vertices are called edges and are usually represented as $A - B$. In a graph, two vertices are said to be *adjacent* if there exists an edge or an arc between them. A *convergent connection* is a triple of vertices (X, Y, Z) where $X \to Y \leftarrow Z$. In such a structure, Y is called a

1. By abuse of notation, we use interchangeably $X \in \mathbf{V}$ to denote a node/vertex in the BN and its corresponding random variable.

collider. If, in addition, X and Z are nonadjacent, the convergent connection is called a *v-structure* and Y is an *unshielded collider*. In BNs, the graphical structure encodes sets of conditional independences among random variables and different DAGs may represent the same set of conditional independences, hence the same sets of joint probability distributions. In this case, they are called *Markov equivalent* and they share exactly the same skeleton (the graph obtained from the BN's graph by removing the orientation of its arcs) and the same set of v-structures. This is the reason why some learning algorithms first try to learn the skeleton of the BN and, then, to determine its set of v-structures. The resulting graph is called a partially directed acyclic graph (*PDAG*). A PDAG is therefore a graph containing both edges and arcs but no directed cycle. A PDAG can be completed by orienting the edges compelled to a specific direction in order to avoid creating new v-structures. The resulting graph is called a completed partially directed acyclic graph (*CPDAG*) and is a representation of a *Markov equivalence class*, i.e., the class of Markov equivalent BNs.

2.1 Related Structure Learning Algorithms

Let us now review some structure learning methods useful for our paper. They may not be able to retrieve a full DAG, but they can theoretically find the Markov equivalence class of the joint probability distribution that generated the data.

Constraint-based methods are such algorithms. They rely on statistical independence tests to determine the set of conditional independences underlying the joint distribution. In the rest of the paper, X being conditionally independent of Y given a set $\mathbf{Z}$ is denoted by $X \perp\!\!\!\perp Y|\mathbf{Z}$ or by $X \perp\!\!\!\perp Y$ if $\mathbf{Z}$ is empty. PC (Spirtes and Glymour (1991)) is representative of this class of algorithms. It is divided into three steps. First, conditional independence tests are performed to determine the skeleton of the network. Whenever PC determines that $X \perp\!\!\!\perp Y|\mathbf{Z}$, it marks $\mathbf{Z}$ as a separation set, a.k.a. a *sepset*, of (X, Y). In a second step, sepsets are exploited to convert all the structures $X - V - Y$ satisfying some conditions into v-structures $X \rightarrow V \leftarrow Y$, hence recovering a PDAG. Finally, exploiting the rules provided in Meek (1995), the CPDAG representing the set of all the BNs that generated the dataset is recovered. If the relations between the variables in the dataset are causal, then the arcs of this CPDAG are causal relations and there just remains to orient the edges in a causal direction to recover a CBN. However, PC is very sensitive to errors in the statistical tests. Hence, more robust variants have been proposed in the literature. For instance, Stable-PC (Colombo and Maathuis (2014)), which we exploit below, reduces the errors induced by the order in which the variables are examined. PC and Stable-PC rely on some hypotheses to correctly uncover the CPDAG searched for, notably one called the causal faithfulness condition. Unfortunately, when the dataset is small or somewhat unreliable, this condition may not be satisfied and the quality of the causal structure learnt can be low. Other variants like conservative-PC Ramsey et al. (2006) therefore relax such hypotheses, e.g., using the adjacency faithfulness condition, which allow them to get better results in these situations, notably when identifying the v-structures.

Higher quality can also be gained by exploiting expert's knowledge. For instance, in de Campos and Castellano (2007), experts can express their beliefs about the existence or lack of arcs or edges. These beliefs are certain and, therefore, are used as constraints in the learning algorithm: when the expert asserts that an arc or an edge exists, no independence

test is allowed to be performed to remove it; similarly, when she asserts that it does not exist, it is not allowed to be added to the learnt graph.

Score-based algorithms are another paradigm for learning the structure of BNs and, partially, of CBNs. The idea is to explore the space of DAGs in order to maximize a score that essentially represents the likelihood $P(D|G)$ of the data given the graphical structure. Exact algorithms finding the global maximum of the problem do exist but are limited in the number of vertices and, therefore, people often resort to exploit approximate local search algorithms like greedy hill climbing or local search with tabu list. For CBNs, especially when latent variables exist, score-based approaches are more limited than constraint-based approaches in the information they can extract from the data. This explains why constraint-based algorithms are more common to learn CBNs. But when there exists no latent variable, the arcs of the CPDAG corresponding to the Markov equivalence class of the DAG produced by score-based algorithms correspond theoretically to causal relations. Different scores, e.g., BDeu or BIC, have been provided in the literature. They essentially differ by the *priors* they rely on. A solution to introduce expert's knowledge in score-based algorithms lies precisely in these *priors*. In Amirkhani et al. (2017) for instance, uncertain knowledge about the structure is introduced as a prior into the BDeu score as $P(G|D, O, \gamma) \propto P(G, D, O, \gamma) = P(G)P(D|G)P(O|G, \gamma)$, where O represents the statements of the experts, γ is the estimated experts' accuracy parameter and $P(O|G, \gamma)$ is computed using a decision tree.

3. A new Algorithm for Learning CBNs with Experts' Knowledge

In this section, we introduce a new hybrid algorithm that exploits uncertain expert knowledge to learn a CBN structure. This knowledge is expressed by statements "I know that there is a direct causal relation between A and B but I am not fully sure whether A causes B or B causes A". For instance, in the medical domain, using a new drug (A) may help patients to recover (C), but blood pressure (B) may affect this recovery. A causal relationship between A and B might be established by experts but they could be uncertain whether the recovery results from drug A having an impact on blood pressure B or from blood pressure B increasing the efficiency of drug A. We suppose that the uncertainty about the direction of the causal relation is expressed by the expert as a probability. Therefore, the expert knowledge $\mathbf{K}$ can be represented as a set of pairs $(A \to B, p_{AB})$ where p_{AB} is the expert's subjective probability that the causal direction is $A \to B$. We assume that $p_{AB} \geq 0.5$, i.e., if the expert believes that there is a causal edge between A and B and that there is a probability p lower than 0.5 that A causes B, then she must also believe that there is a probability $p_{BA} = 1 - p \geq 0.5$ that B causes A, and only $(B \to A, p_{BA})$ is stored into $\mathbf{K}$. We further divide $\mathbf{K}$ into $\mathbf{K}_a = \{(A \to B, p_{AB}) \in \mathbf{K} : p_{AB} = 1\}$ and $\mathbf{K}_e = \mathbf{K} \backslash \mathbf{K}_a$. $\mathbf{K}_a$ is therefore the set of statements for which the causal direction is known with certainty by the expert and $\mathbf{K}_e$ is the set of statements for which the expert is sure that there is a causal relation between A and B but is unsure about the causal direction. Finally, we assume that certain knowledge $\mathbf{K}_a$ is coherent, i.e., the arcs it contains do not form directed cycles.

In the sequel, we assume the adjacency-faithfulness and does not contain latent or selection variables[2]. Our algorithm is divided into three steps. The first one exploits a constraint-based method in order to get a first draft of the causal structure. In the

2. See Spirtes et al. (2000) for definitions of latent and selection variables

second step, a score-based method resolves some conflicts and completes the determination of the set of v-structures. Using an interactive process with the expert, the final step of the algorithm fixes the orientation of the arcs whose causal directions could not yet be determined with certainty.

3.1 The Constraint-based Step

For learning the skeleton, we use a modified version of the stable-PC algorithm (Colombo and Maathuis (2014)) that exploits the expert's knowledge as advocated in de Campos and Castellano (2007): all the edges specified by the expert are fixed in Graph G (lines 2 and 7). In addition, the Markov property, the independence property underlying BNs, states that node X is independent of its nondescendants given its parents. Hence, we can exclude from the possible sepsets $\mathbf{Z}$ all those that include nodes in $\text{Desc}_{\mathbf{K}_a}(X)$, the set of nodes that the expert knows with certainty (in $\mathbf{K}_a$) to be descendants of X (line 9).

Algorithm 1: LearningSkeleton

Input: dataset D, expert knowledge $\mathbf{K}$
Output: an undirected graph G and a set of Sepsets

1 $\mathbf{V} \leftarrow$ all the variables/columns of D ; $\mathbf{E} \leftarrow \mathbf{V} \times \mathbf{V}$; $G \leftarrow$ complete graph $(\mathbf{V}, \mathbf{E})$
2 $\mathbf{E}_\mathbf{K} \leftarrow \{A - B : (A \rightarrow B, p_{AB}) \in \mathbf{K}\}$
3 $m \leftarrow -1$; $Sepset \leftarrow \emptyset$
4 **repeat**
5 $m \leftarrow m + 1$
6 **foreach** *vertex* X *in* $\mathbf{V}$ **do** $\mathbf{adj}_X \leftarrow \{Y \in \mathbf{V} : X - Y \in \mathbf{E}\}$;
7 **foreach** $(X, Y) \in \mathbf{V}^2$ *s.t.* $X - Y \in \mathbf{E} \backslash \mathbf{E}_\mathbf{K}$ *and* $|\mathbf{adj}_X| \geq m + 1$ **do**
8 **repeat**
9 Choose a new set $\mathbf{Z} \subseteq \mathbf{adj}_X \backslash (\{Y\} \cup \text{Desc}_{\mathbf{K}_a}(X))$ s.t. $|\mathbf{Z}| = m$
10 **if** $X \perp\!\!\!\perp Y | \mathbf{Z}$ **then**
11 Remove edge $X - Y$ from $\mathbf{E}$; $Sepset(\{X, Y\}) \leftarrow \mathbf{Z}$
12 **until** *either* $X \perp\!\!\!\perp Y | \mathbf{Z}$ *or all possible* $\mathbf{Z}$ *have been considered*;
13 **until** *all vertices* $X \in \mathbf{V}$ *are such that* $|\mathbf{adj}(X)| \leq m$;
14 **return** $G, Sepsets$

The next step of PC-like methods is to orient the v-structures. Here, we modify conservative-PC (Ramsey et al. (2006)) to exploit the expert's knowledge, as shown in Algorithm 2: we use metasymbol $\circ$ to represent either nothing or $>$, i.e., edge $X - \circ Y$ means either $X - Y$ or $X \rightarrow Y$. Line 3 computes the usual set of v-structures of G, excluding those incompatible with certain knowledge $\mathbf{K}_a$. $\mathbf{S}_K$ and $\mathbf{S}_V$ contain the set of v-structures that do not agree with uncertain knowledge $\mathbf{K}_e$ and with other v-structures respectively. The loop orients the v-structures, allowing arcs in both directions between any pair of nodes. In such a case, or when conservative-PC finds the v-structure "unfaithful", these arcs are marked as "temporary" because there still exists some uncertainty about their existence in the causal model. Otherwise, or when the arcs are known to exist with certainty by the expert, they are marked as "definitive". Usually, constraint-based approaches are completed by orienting the edges compelled to a specific direction in order to avoid creating new v-structures. For this purpose, in our algorithm, we use the rules defined in Meek (1995), with the following

adaptation: in rules R1 to R4, the edge to be transformed into an arc is, in our context, either an edge or an arc marked as temporary; the other edges (resp. arcs) considered in these rules are also edges here (resp. arcs, either definitive or temporary).

Algorithm 2: Orient V-structures

Input: dataset D, expert knowledge $\mathbf{K}$, an undirected graph $G = (\mathbf{V}, \mathbf{E})$
Output: a partially directed graph G and marks on the arcs

1 **foreach** *vertex* X *in* $\mathbf{V}$ **do** $\mathbf{adj}_X \leftarrow \{Y \in \mathbf{V} : X - Y \in \mathbf{E}\}$;
2 **foreach** $(X \rightarrow Y, p_{XY}) \in \mathbf{K}_a$ **do** substitute $X - Y$ by $X \rightarrow Y$ in $\mathbf{E}$;
3 $\mathbf{S} \leftarrow \{(X, Y, Z) \in \mathbf{V}^3 : X \multimap Y \circ\!\!-\!\!\circ Z \in \mathbf{E}, X \circ\!\!-\!\!\circ Z \notin \mathbf{E}$ and there exists $\mathbf{T} \subseteq \mathbf{adj}_X \backslash \{Y\}$ or $\mathbf{T} \subseteq \mathbf{adj}_Z \backslash \{Y\}$ s.t. $X \perp\!\!\!\perp Z | \mathbf{T}\}$
4 $\mathbf{S}_K \leftarrow \{(X, Y, Z) \in \mathbf{S} : (Y \rightarrow X, p_{YX}) \in \mathbf{K}_e$ or $(Y \rightarrow Z, p_{YZ}) \in \mathbf{K}_e\}$
5 $\mathbf{S}_V \leftarrow \{(X, Y, Z) \in \mathbf{S}$: there exists $T \in \mathbf{V}$ s.t. $(Y, X, T) \in \mathbf{S}$ or $(Y, Z, T) \in \mathbf{S}\}$
6 **foreach** $(X, Y, Z) \in \mathbf{S}$ **do**
7 **if** *there exists* $\mathbf{T} \subseteq \mathbf{adj}_X$ *or* $\mathbf{T} \subseteq \mathbf{adj}_Z$ *s.t.* $Y \in \mathbf{T}$ *and* $X \perp\!\!\!\perp Z | \mathbf{T}$ **then** /* *unfaithfulness* */
8 **if** $(X, Y, Z) \notin \mathbf{S}_K$ *and* $(X, Y, Z) \notin \mathbf{S}_V$ **then**
9 Remove edges $X - Y$ and $Y - Z$ from $\mathbf{E}$ (if they still exist)
10 Add arcs $X \rightarrow Y$ and $Y \leftarrow Z$ to $\mathbf{E}$ and mark them as temporary
11 **else if** $(X, Y, Z) \in \mathbf{S}_K$ *or* $(X, Y, Z) \in \mathbf{S}_V$ **then**
12 Remove edges $X - Y$ and $Y - Z$ from $\mathbf{E}$ (if they still exist)
13 Add arcs $X \rightarrow Y$ and $Y \leftarrow Z$ to $\mathbf{E}$ and mark them as temporary
14 **else if** $(X, Y, Z) \notin \mathbf{S}_K$ *and* $(X, Y, Z) \notin \mathbf{S}_V$ **then**
15 Remove edges $X - Y$ and $Y - Z$ from $\mathbf{E}$ (if they still exist)
16 Add arcs $X \rightarrow Y$ and $Y \leftarrow Z$ to $\mathbf{E}$ and mark them as definitive
17 **foreach** $(X \rightarrow Y, p_{XY}) \in \mathbf{K}_a$ **do** Mark $X \rightarrow Y$ as definitive ;
18 **return** *partially directed graph* G *and the marks on the arcs*

3.2 The Score-based and the Interactive Steps

The graph computed above needs to be mapped into a DAG G that can feed the score-based algorithm phase. To do so, we use Algorithm 3. The orientations of the arcs on Line 8 are somewhat arbitrary and these are the only ones that the score-based step can possibly change. The orientations of the other arcs are known at this step to be compulsory, either because they are part of or are compelled by a v-structure or because they belong to certain expert's knowledge $\mathbf{K}_a$. A score-based algorithm can now be used to improve DAG G, thereby fixing the arbitrariness of the arcs' orientations of Line 8. For this purpose, we exploit the greedy search algorithm proposed in Amirkhani et al. (2017) except that i) DAG G is the structure used at the beginning of the search; and ii) the only operations allowed are the reversals of the *temporary* arcs, all the other operations (reversals of the definitive arcs, arc additions, arc deletions) being forbidden. Let us call G^* the DAG resulting from the application of this algorithm. Theoretically, the only causal orientations actually identified in DAG G^* are its v-structures and the arcs whose orientation they compel (see Meek's rules). The orientation all the other arcs is meaningless from a causal point of view.

Let G' be the partially directed graph obtained from G^* by replacing by undirected edges $X - Y$ all the arcs $X \rightarrow Y$ that are neither part of nor compelled by a v-structure.

Algorithm 3: Conversion of the partially directed graph into a DAG.

Input: a partially directed graph $G = (\mathbf{V}, \mathbf{E})$ and marks on the arcs
Output: a DAG G and marks on the arcs

1 **foreach** $X, Y \in \mathbf{V}^2$ *s.t.* $X \to Y \in \mathbf{E}$ *and* $X \leftarrow Y \in \mathbf{E}$ *and are marked as temporary* **do**
2 Substitute $X \to Y$ and $X \leftarrow Y$ by $X - Y$ in $\mathbf{E}$
3 **foreach** $X, Y \in \mathbf{V}^2$ *s.t.* $X \to Y \in \mathbf{E}$ *and is marked as temporary* **do**
4 Mark $X \to Y$ as definitive
5 **if** $X \to Y$ *creates a directed cycle with only definitive arcs* **then**
6 Substitute $X \to Y$ by $Y \to X$ in $\mathbf{E}$ and mark it as definitive
7 **foreach** $X, Y \in \mathbf{V}^2$ *s.t.* $X - Y \in \mathbf{E}$ **do**
8 Orient edge $X - Y$ without creating any new directed cycle ; mark it as temporary
9 **return** *DAG G and marks on the arcs*

At this point, only expert knowledge can help orienting in a causal way the edges $X - Y$ of G'. Theorem 2 shows how this can be done: select any edge, orient it according to the expert's knowledge and propagate w.r.t. Meek's rules. Iterating this process, we can orient as many remaining edges as possible. We first apply it with "sufficiently strong" knowledge already in $\mathbf{K}_e$, i.e., those whose beliefs p_{XY} are greater than a given threshold p. Then, we perform the interactive process of Algorithm 4 on the resulting graph. The choice of the edges is made on Line 2. Basically, any functions u and f could be used but, in order to minimize the number of questions asked to the expert, we exploit the following functions: given an edge $A - B$, $u(A \to B)$ counts the number $n_{A \to B}$ of edges that will be oriented due to the application of Meek's rules. If the expert answers that the causal orientation is actually $A \to B$, then these $n_{A \to B}$ edges will be automatically transformed into arcs, thereby enabling to avoid asking $n_{A \to B}$ questions to the expert. To minimize the number of questions asked, we should therefore look for the edge inducing the highest number $n_{A \to B}$. Before asking the question, the orientation provided by the expert is unknown, so we assign to each edge $A - B$ the pair $(n_{A \to B}, n_{B \to A})$. Minimizing the number of questions therefore amounts to solve a bicriteria optimization problem. To do so, we used as function f the min function, hence resulting in a maximin optimization problem.

Theorem 2 *Let $G_0, \ldots, G_k$ be a sequence of partially directed graphs such that $G_0 = G'$ and, for all $i < k$, G_i is obtained from G_{i-1} by orienting any edge $X - Y$ of G_{i-1} in any direction and, then, by applying Meek's rules. Then $G_0, \ldots, G_k$ are PDAGs and can be completed into DAGs that belong to the same Markov equivalence class as G'.*

Proof $G_0 = G'$ results from DAG G^* by substituting by edges $X - Y$ all the arcs $X \to Y$ that are neither part of nor compelled by a v-structure. Hence, G_0 is a CPDAG representing a nonempty Markov equivalence class $\mathcal{C}$ (it notably contains G^*). Select any undirected edge $X - Y$ of G_0 and orient it arbitrarily, say as $X \to Y$. As $X - Y$ is undirected in CPDAG G_0, this means that its orientation is not compelled, hence there exists at least one DAG G''_{DAG} that contains all the arcs of G_0 plus $X \to Y$, and that belongs to $\mathcal{C}$. Arc $X \to Y$, all those it compels and the other arcs of G_0 necessarily belong to G''_{DAG}. Hence, G_1 is such that its undirected edges can be substituted by arcs to obtain DAG G''_{DAG}. It is therefore a

Algorithm 4: The interactive step

Input: a partially directed graph $G = (\mathbf{V}, \mathbf{E})$, two functions f and u, a threshold p
Output: a partially directed graph G

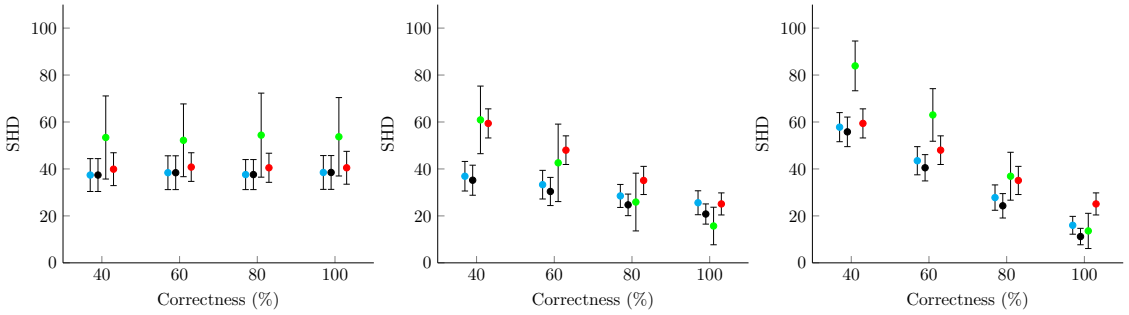
repeat
  Let $X - Y$ be an edge maximizing $\{f(u(A \leftarrow B), u(A \rightarrow B)) : A - B \in \mathbf{E}\}$
  Ask the expert her belief, say $(X \rightarrow Y, p_{XY})$, about the causal orientation of $X - Y$
  if $p_{XY} \geq p$, *i.e., the belief is sufficiently strong* **then**
    Orient $X - Y$ in $\mathbf{E}$ according to the expert's knowledge, i.e., as $X \rightarrow Y$
    Apply Meek's rules to propagate the orientations of the edges compelled by this orientation
until *Either there exists no more edges* $X - Y$ *in* $\mathbf{E}$ *or the expert has only weak beliefs* $(p_{XY} < p)$ *or no belief at all about the remaining edges*;
return Graph G

PDAG. For the same reason, the orientations of the undirected edges remaining in G_1 are, by definition, not compelled. The same reasoning can be applied, hence resulting in a new graph G_2 whose edges can be substituted by arcs in order to obtain a DAG still belonging to equivalence class $\mathcal{C}$. By induction, the property holds for all graphs G_i, $i \leq k$. ■

4. Experiments

In this section, we demonstrate the efficiency of our approach by comparing it with the algorithms provided in de Campos and Castellano (2007) and Amirkhani et al. (2017) (denoted by PSDC and GSA respectively). We investigate the impact of the expert's knowledge (either certain or uncertain) on the causal learning process and show that it can be beneficial to infer the right causal direction. To do so, we randomly generate training datasets from benchmark networks taken from a BN repository[3]. Samples of sizes ranging from 500 to 10000 records are generated. The expert's knowledge (provided by only one expert) is simulated through the selection of a subset of arcs of the original DAG (50% of the number of arcs except for Figure 2(b)) and the threshold p for Algorithm 4 is set to 0.6. To evaluate the sensitivity of our learning procedure w.r.t. errors, part of the expert knowledge is transformed into erroneous information, i.e., we introduce the opposite causal orientations of the existing arcs in the original DAG G. Our approach, PCSDC, and GSA are compared on the basis of the graphs they learn. To do so, the produced graph $\tilde{G}$ is compared with G using the Structural Hamming Distance (SHD) criterion: the connection between any pair of nodes (X, Y) can be represented as $X \circ\!\!-\!\!-\!\!\circ Y$, where $--$ is either nothing or a line, and $\circ$ are either nothing or $>$. For instance, for nonadjacent nodes, $--$ and both $\circ$ are nothing, whereas for $X \leftarrow Y$, the left $\circ$ is $>$, the right $\circ$ is nothing and $--$ is a line. The SHD of (X, Y) is the sum of the differences over the left $\circ$, the right $\circ$ and $--$ in both graphs, e.g., the SHD between $X \leftarrow Y$ and $X \rightarrow Y$ is $1 + 1 + 0 = 2$. The SHD between G and $\tilde{G}$ is half the sum over all pairs of nodes (X, Y) of the SHD of (X, Y). The half factor prevents counting the SHD of (X, Y) and (Y, X) twice. The half factor prevents counting the SHD of (X,Y) and (Y,X) twice.

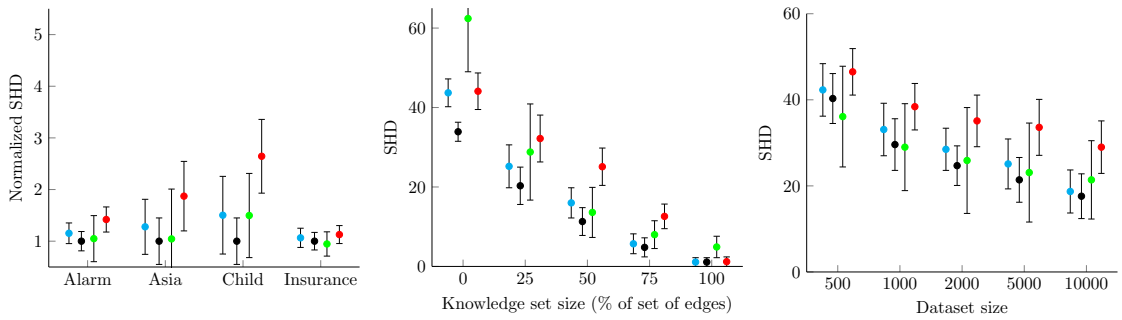
3. https://www.bnlearn.com/bnrepository/



(a) $p_{AB} = 0.5$ (b) $p_{AB} = 0.75$ (c) $p_{AB} = 1$

Figure 1: Average SHDs after the second step of our algorithm (blue), our whole algorithm (black), GSA (green) and PCSDC (red) for the Alarm BN in function of the percentage of correctness and p_{AB}.

To make the comparison more thorough, in the experiments, we vary the parameters related to the size of the dataset, the percentage of correct expert's knowledge, and the expert's confidence p_{AB} of giving a correct knowledge. Finally, for each set of the above parameters, 100 tests are performed and figures 1 and 2 display average SHD scores. All experiments are performed on a 1.9GHz Intel Core i7 computer with 16GB of memory running Windows 10. The studied learning algorithms are implemented using libraries NetworkX (Hagberg and Conway (2020)) and pyAgrum (Ducamp et al. (2020)). Figure 1 displays the average SHD scores for the ALARM BN in function of p_{AB} and the percentage of correct expert's assertions. As can be expected, when parameter p_{AB} is equal to 0.5, the expert knowledge does not lead to any significant changes in the final results whatever the percentage of correct assertions (see Figure 1(a)). The impact of the expert's knowledge is more visible when p_{AB} is equal to 0.75 (see Figure 1(b)). In this case, the final result of our algorithm outperforms other approaches whatever the percentage of correct assertions: with a percentage of correctness equal to 40%, PCSD and GSA are doomed to be ineffective to handle incorrect assertions. The average SHD of our algorithm is actually smaller than those of PCSDC and GSA by 24 and 25 operator changes respectively. Generally, the greater the expert's uncertainty and the percentage of incorrect assertions, the larger the SHD difference between our approach and the other methods. This is due to the original way in which the expert's knowledge is handled during model learning even in the presence of incorrect information. When the correctness percentage lies between 40% and 80% and p_{AB} is equal to 1 (see Figure 1(c)), we can see that the quality of the causal model approximation degrades for GSA and our approach w.r.t. to the previous results. While the expert's certainty is a plausible parameter, there is nevertheless the risk of biasing the estimated causal model, especially when the expert introduces too many mistakes. In Figure 2(a), we display the average normalized SHD results for several benchmark BNs. Note that the normalization is obtained by dividing each SHD score by the corresponding SHD of our approach on the same dataset and reporting average scores over the simulated datasets. Average normalized SHD scores greater than one correspond to algorithms with more structural errors than our approach. As can be seen, for all benchmark networks, our approach finds better models. For the Child BN, SHD scores for PCSDC and GSA are roughly 3 and 1.5 times higher than



(a) Normalized SHDs (b) Average SHDs (c) Average SHDs

Figure 2: Normalized SHDs for various BNs (a) and average SHDs for the Alarm BN in function of the size of the knowledge (b) or of the dataset (c): after the score-based-based step of our algorithm (blue), our whole algorithm (black), GSA (green) and PCSDC (red).

that of our approach. Again, these results highlight the effectiveness of our proposed rules to handle expert's assertions during the learning phases and the way that knowledge is elicited from the expert. Finally, the accuracy of our algorithm can also be demonstrated by varying the knowledge assertions and dataset sizes. In Figure 2(b), the correctness percentage and the confidence p_{AB} are fixed at 100% and 1 respectively. Overall, our approach is on par, or better than all algorithms: with a knowledge set size equal to 25%, the SHD score of our algorithm is actually lower than that of PCSDC and GSA by about 12 and 8 operation changes respectively. When p_{AB} and the correctness rate are respectively equal to 0.75 and 80%, except for the cases where the dataset size is small (between 500 and 1000), our algorithm always outperforms the PCSDC and GSA methods (see Figure 2(c)).

5. Conclusion

In this paper, we proposed a new algorithm for learning the structure of causal models in the presence of uncertain expert's knowledge. This algorithm relies on effective rules to handle the inherent uncertainty during the different learning steps. As shown in the experiments, our approach outperforms other causal learning approaches. For future work, we plan to develop an extended version of the algorithm to address the existence of latent variables.

References

H. Amirkhani, M. Rahmati, P. J. F. Lucas, and A. Hommersom. Exploiting experts' knowledge for structure learning of Bayesian networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2154–2170, 2017.

G. Borboudakis and I. Tsamardinos. Incorporating causal prior knowledge as path-constraints in Bayesian networks and maximal ancestral graphs. In *Proc. of the International Conference on Machine Learning*, pages 427–434, 2012.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2003.

D. Colombo and M. H. Maathuis. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 15:3921–3962, 2014.

D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40 (1):294–321, 2012.

C. de Campos and Q. Ji. Efficient structure learning of Bayesian networks using constraints. *Journal of Machine Learning Research*, 12:663–689, 2011.

L. M. de Campos and J. G. Castellano. Bayesian network learning algorithm using structural restrictions. *International Journal of Approximate Reasoning*, 45:233–254, 2007.

G. Ducamp, C. Gonzales, and P.-H. Wuillemin. aGrUM/pyAgrum: a toolbox to build models and algorithms for probabilistic graphical models in python. In *Proc. of the International Conference on Probabilistic Graphical Models*, pages 609–612, 2020.

A. Gretton, P. Spirtes, and R. Tillman. Nonlinear directed acyclic structure learning with weakly additive noise models. In *Proc. of the International Conference on Neural Information Processing Systems*, pages 1847–1855, 2009.

A. Hagberg and D. Conway. Networkx: Network analysis with python, 2020.

D. Janzing and B. Schölkopf. Causal inference using the algorithmic Markov condition. *IEEE Transactions on Information Theory*, 56(10):5168–5194, 2010.

H. Li, V. Cabeli, N. Sella, and H. Isambert. Constraint-based causal structure learning with consistent separating sets. In *Proc. of the International Conference on Neural Information Processing Systems*, pages 14257–14266, 2019.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proc. of the Conference on Uncertainty in Artificial Intelligence*, pages 403–410, 1995.

J. Pearl. *Causality.* Cambridge University Press, 2nd edition, 2009.

J. Peters, D. Janzing, and B. Schölkopf. Identifying cause and effect on discrete data using additive noise models. In *Proc. of the International Conference on Artificial Intelligence and Statistics*, pages 597–604, 2010.

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

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

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search.* MIT press, 2nd edition, 2000.

J. Xu, Y. Zhao, J. Chen, and C. Han. A structure learning algorithm for Bayesian network using prior knowledge. *Journal of Computer Science and Technology*, 30:713–724, 2015.