ELSEVIER International Journal of Approximate Reasoning 24 (2000) 11–37

INTERNATIONAL JOURNAL OF APPROXIMATE REASONING

www.elsevier.com/locate/ijar

A new approach for learning belief networks using independence criteria ☆

Luis M. de Campos *, Juan F. Huete

Dpto. de Ciencias de la Computación e Inteligencia Artificial, E.T.S.I. Informática, Universidad de Granada, 18071 Granada, Spain

Received 1 November 1998; accepted 1 June 1999

Abstract

In the paper we describe a new independence-based approach for learning Belief Networks. The proposed algorithm avoids some of the drawbacks of this approach by making an intensive use of low order conditional independence tests. Particularly, the set of zero- and first-order independence statements are used in order to obtain a prior skeleton of the network, and also to fix and remove arrows from this skeleton. Then, a refinement procedure, based on minimum cardinality d-separating sets, which uses a small number of conditional independence tests of higher order, is carried out to produce the final graph. Our algorithm needs an ordering of the variables in the model as the input. An algorithm that partially overcomes this problem is also presented. © 2000 Elsevier Science Inc. All rights reserved.

1. Introduction

Belief Networks (also called Bayesian networks, causal networks or influence diagrams) are Knowledge-Based Systems that represent uncertain knowledge by means of both graphical structures, namely directed acyclic graphs (dags), and numerical parameters associated to these graphs. In a belief network, the qualitative component (the graph) represents dependence and independence

☆ This work has been supported by the CICYT under Project TIC96-0781.

* Corresponding author. Tel.: +34-958-244019; fax: +34-958-243317.

E-mail addresses: lci@decsai.ugr.es (L.M. de Campos), jhg@decsai.ugr.es (J.F. Huete).

PII: S 0 8 8 8 - 6 1 3 X (9 9) 0 0 0 4 2 - 0

statements: the absence of some arcs means the existence of certain conditional independence relationships between variables, and the presence of arcs may represent the existence of direct dependence relationships (if a causal interpretation is given, then the arcs signify the existence of direct causal influences between the linked variables). The quantitative component is a collection of uncertainty measures, which shape the relationships, where the dominant approach is based on the use of probability measures [25]. These structures have been used in different applications, as fault diagnosis, medical expert systems, and software debugging [17].

Once a complete belief network has been built, it constitutes an efficient device to perform inferences [24,25]. However, there still remains the previous problem of building such a network, i.e., to provide the graph structure and the numerical parameters necessary for characterizing the network. So, an interesting task is then to develop automatic methods capable of learning the network directly from raw data, as an alternative or a complement [16] to the (difficult and time-consuming) method of eliciting opinions from experts.

Learning belief networks includes different problems: learning the structure, the parameters given a structure, hidden variables whose values are not present in the data, etc. In this paper we consider the problem of learning the structure of a belief network from data using independence-based criteria. The main contribution of this research lies in the development of a new algorithm which, by first extracting as much information as possible of conditional independence tests of very low order (zero and one), and next using a small number of additional independence tests of higher order (but keeping this order as low as possible), tries to reduce the computational complexity of the learning process and increase the reliability in the learned structure. We also explore the use of conditional independence tests of order zero and one to approximate a causal ordering of the variables in the model. This ordering can be used to reduce considerably the search space.

The rest of the paper is organized as follows: in Sections 1.1 and 1.2 we briefly recall how independence statements can be represented in belief networks and some general ideas about the learning of these structures. In Section 2 we describe a new approach for learning, mainly based on an intensive use of conditional independence tests of low order. Section 3 describes some preliminary experiments with the new method. Section 4 contains the concluding remarks. Finally, Appendix A contains the proofs of all the results stated in the paper.

1.1. Preliminaries

Belief networks [25] allow us to represent our knowledge about a given domain by means of directed acyclic graphs. In an abstract way, a belief network can be considered as a representation of a *Dependency Model*, i.e., a pair

$M = (U, I)$, where U is a set of variables [1] and I is a rule which assigns truth values to the predicates $I(X, Y \mid Z)$ (read 'X is independent of Y, given Z'), where X, Y and Z are disjoint subsets of variables in U. The intended interpretation of $I(X, Y \mid Z)$ is that having observed Z, no additional information about X could be obtained by also observing Y. A criterion for testing independence statements is an essential component of any dependency model. For example, a probability distribution can be considered as a dependency model, where the predicate I is defined through the concept of stochastic independence, i.e., $I(X, Y \mid Z)$ holds if and only if $P(\mathbf{x} \mid \mathbf{y}, \mathbf{z}) = P(\mathbf{x} \mid \mathbf{z})$, whenever $P(\mathbf{y}, \mathbf{z}) > 0$, for every instantiation $\mathbf{x}, \mathbf{y}, \mathbf{z}$ of the subsets of variables X, Y, Z, respectively. Throughout the paper, subsets of variables will be denoted by capital letters, whereas single variables will be represented by lowercase letters. The connection between belief networks and dependency models may be stated by means of the so-called d-separation criterion [25,33], which may be considered as a graphical definition of conditional independence (which turns a belief network into a dependency model itself).

Definition 1.1. Given a directed acyclic graph G, a trail C (a trail in a directed graph is a sequence of adjacent nodes, the direction of the arrows does not matter) from node x to node y is said to be blocked by the set of nodes Z, if there is a node $z \in C$ such that, either

- $z \in Z$ and arrows of C do not meet head to head at z, or
- $z \notin Z$, nor has z any descendants in Z, and the arrows of C do meet head to head at z.

A trail that is not blocked by Z is said to be opened by Z.

Definition 1.2 (*d-separation*). Let X, Y and Z be three disjoint subsets of nodes in a dag G. Then, X and Y are said to be d-separated (or graphically independent) by Z, and this is denoted by $\langle X, Y \mid Z \rangle_G$, if all trails between the nodes in X and the nodes in Y are blocked by Z. Otherwise, X and Y are graphically dependent, given Z.

The *skeleton* of a dag G is the undirected graph obtained from G by replacing the arrows by edges. When we speak about cycles in a dag, we always refer to undirected cycles in the skeleton of G (a dag cannot contain directed cycles). Note that in a dag every cycle must contain at least one *head to head connection*, i.e., a subgraph of the form $x \rightarrow z \leftarrow y$; the middle node in a head to head connection is called a *head to head node*.

We say that $I(X, Y \mid Z)$ is a conditional independence statement of *order k* if the cardinality of the conditioning set Z is equal to k.

[1] In this paper we only consider discrete variables with a finite domain of values.

A belief network may also represent quantitative knowledge. Let us suppose that this knowledge is probabilistic. In that case, for each variable, x_i, we need a family of conditional probability distributions $P(\mathbf{x}_i \mid \pi(\mathbf{x}_i))$, where $\mathbf{x}_i$ represents any assignment of values to the variable x_i, $\Pi(x_i)$ is the parent set of x_i in the network, and $\pi(\mathbf{x}_i)$ denotes any assignment of values to the variables in the set $\Pi(x_i)$. Then, the joint probability distribution, P, can be obtained by means of the following expression:

$$P(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \prod_{i=1}^{n} P(\mathbf{x}_i \mid \pi(\mathbf{x}_i)).$$

If we start out from a joint probability distribution P (a dependency model), we can look for a belief network G that is a ‘good’ graphical representation of P. In that case, we may find different situations [25]: if every d-separation condition in G corresponds to a valid conditional independence relationship in P, then G is said to be an *I-map* of P; if every conditional independence relationship in P corresponds to a d-separation condition in G, then G is a *D-map* of P; finally, if G is both an I-map and a D-map of P, then we say that G is a *perfect map* of P and that P is *dag-isomorphic*. For structural learning it is commonly assumed that there exists a perfect map associated with the underlying probability distribution of the data.

1.2. Learning belief networks

In general, learning from data involves a search from an exponential number of network structures. Regardless of the methodology used to guide the search process, two problems have to be considered [6]:

- *Sample complexity*, related to the number of cases required for training. With a large sample, we have more confidence on the reliability of the learned model, but greater complexity to process the data. Some results from computational learning theory [15] show that, under many conditions, learning a model close to the ‘true’ model is possible when the sample size increases exponentially with the number of variables involved in the problem. This learning is possible by many algorithms that asymptotically converge to the ‘truth’. With a more realistic medium-sized sample, some algorithms should perform better than others, depending on how well their particular biases align with the ‘true’ model.
- *Computational complexity*, related to the time or space required for optimization. Given that there is an exponential number of possible networks, it should not be surprising that the learning process is a NP-Hard problem [11]. Learning can be considered as a maximization problem where we intend to find a network which maximizes some quality measure or scoring metric (sample likelihood, entropy, Bayesian, description-length, etc.

[12,16,22,28]) reflecting the goodness-of-fit of the structure to the model. A different approach is obtained when independence criteria are used to learn the network [8,10,13,14,25,26,29,30]. Those algorithms attempt to recover the network that represents most of the independences observed in the data. Finally, there are also hybrid algorithms that combine conditional independence and scoring metrics [3,27,31]. Greedy algorithms, iterated local search or branch and bound techniques (among others) have been employed to guide this search.

As learning belief networks is a NP-Hard problem, in order to make easier the learning process, the use of previous information about the model can be helpful to recover it. This previous information may be of different types: For instance, consider that we know an ordering θ among the variables in the model [3,10,12,20,25,32], or that some variables are causally connected [22,32], or that we know some properties about the structure of the model to be recovered [7,8,30], etc. This knowledge might be included in the learning algorithm in order to reduce (some times considerably) the search space. Usually, this information is obtained from an expert in a large elicitation process. However, the development and the use of specific tools allowing to include 'a priori' knowledge is welcome.

In this paper, our interest is focused on studying the methods based on independence criteria for learning the network structure. However, these methods present two main drawbacks: (1) they may need an exponential number of conditional independence tests (related with the computational complexity problem) and (2) these tests involve a great number of variables (related to both the sample [2] and computational complexity [3] problems).

2. A learning algorithm based on conditional independence tests of low order

In this section we describe the proposed learning algorithm. As we have already said, it belongs to the family of independence-based (also called constraint-based) algorithms. In order to reduce the two aforementioned problems of these algorithms, we intend to use, as much as possible, conditional independence tests of low order, involving less variables and thus increasing the sample cardinality for each possible outcome. In this way, the output of each test may be quite reliable without the need of having an enormous amount of data.

[2] The reliability of the tests drastically decreases as the number of variables increases.

[3] Testing conditional independence statements between two variables requires a time which is exponential with the number of variables in the conditioning set.

2.1. Outline of the algorithm

At a first step, we shall use only zero- and first-order conditional independence tests. The information provided by these tests will allow us to fix a prior graphical model (by removing from an initial complete graph those arcs whose terminal nodes display an independence relationship of order 0 or 1), thus reducing the search space. Then, the problem is simplified to look for a subgraph of this prior graph that reflects (as much as possible) the independence statements in the sample. Also, this model could be used as the starting point for other learning algorithms based on scoring metrics [18].

Next, and once again using only the information about conditional independences of order 0 and 1 (shortened, 0–1 independences), the algorithm will be able to remove from the prior graph some additional arrows and fixing some other arrows as permanent.

Finally, once we cannot extract more information from the 0–1 independences, the algorithm will have to use a polynomial number (quadratic, in the worst case) of conditional independence tests of order greater than 1, but always trying to keep this order as low as possible. This is achieved by using d-separating sets of minimum cardinality.

To further reduce the computational complexity problem by using some kind of previous information, our algorithm will start from an ordering among the variables, which can be obtained, for example, from an expert; alternatively, this ordering could be automatically constructed using a specific algorithm [5,9]; more about this will be said in Section 2.7. As we have already commented in Section 1, an ordering of the variables is a common requirement for many learning algorithms.

There are other independence-based learning algorithms that are concerned about the problem of performing independence tests of high order [10,30]. PC [30] tests for conditional independences of higher order only if all the tests of lower order (not discarded yet) have been previously checked. It progress systematically from low order to high order tests. The algorithm in [10] creates a draft of the network based on mutual information (which is roughly equivalent to use marginal independence tests). Then it ‘thickens’ and ‘thins’ the graph using conditional independence tests of (approximately) minimum size.

However, these two algorithms do not exploit all the information provided by low order tests. The first step of our algorithm carries out more tests (all the independence tests of order 0 and 1) but this fact allows it to reduce the number of tests of higher order which are necessary at the final step. Our algorithm is able to use the 0–1 independences not only to remove those arcs linking nodes displaying an independence relationship of order 0 or 1, but also to remove some additional arcs whose terminal nodes exhibit more complicated patterns of independence. Moreover, our algorithm can also fix some arrows, so that it is not necessary to perform additional tests of higher order to decide about their inclusion in the network. We decided to use conditional independences of order 0 and 1 because they are highly expressive for a great amount of models, and specially for those that can be well represented by sparse graphs. Thus, our algorithm should be quite efficient when the database has a sparse underlying network (as it is usually the case for real world applications). In fact, this kind of independence relationships is enough to characterize completely some types of dependency models (those isomorphic to singly connected graphs [7] and simple graphs [8]). Therefore, it seems worth striving for studying how these independence relationships can be used in order to learn more general models. Nevertheless, it is clear that the information obtained only from the 0–1 independences does not allow us to accurately learn a general dag. For that reason the last step of our algorithm has to perform some other tests of order greater than one, but the order of these additional tests is as low as possible because we use a procedure based on finding minimum d-separating sets.

As it is usual for many independence-based learning algorithms [7,8,10,26,29,30], and in order to simplify the exposition, we shall also suppose that the dependency model underlying the database is dag-isomorphic. [4]

2.2. Learning the prior graphical model

We shall use the following notation: Let $M = (U, I)$ be a dependency model. I^M_{0-1} denotes the set of true 0–1 independences in M, i.e.,

$$I^M_{0-1} = \{I(x,y \mid \emptyset), I(x,y \mid z);\; x,y,z \in U\}.$$

Given any dag G, I^G_{0-1} denotes the set of 0–1 graphical independences that can be obtained (via d-separation) from G

$$I^G_{0-1} = \{\langle x,y \mid \emptyset\rangle_G, \langle x,y \mid z\rangle_G;\; x,y,z \in U\}.$$

The algorithm takes as the input the set of 0–1 independences, I^M_{0-1}, which can be obtained easily from a data set. These independence statements are used to learn the skeleton of the prior model (Fig. 1 expresses this idea), i.e., an undirected graph, denoted by G_{0-1}, such that the edge x–y does not belong to G_{0-1} if and only if $I(x,y \mid \emptyset)$ or $I(x,y \mid z)$ (for some variable z) are in I^M_{0-1}

$$G_{0-1} = (U, \{x - y\,;\; x,y \in U,\; \neg I(x,y \mid \emptyset) \text{ and } \neg I(x,y \mid z)\;\; \forall z \in U\}).$$

Assuming that a perfect ordering [5] is known, the undirected graph can be directed using this ordering θ: If $x - y \in G_{0-1}$, and $\theta(y) < \theta(x)$ then direct the

[4] Although not reported here, this assumption can be considerably relaxed without compromising the soundness of our algorithm.

[5] An ordering compatible with the topological ordering of the dag isomorphic to the underlying model.

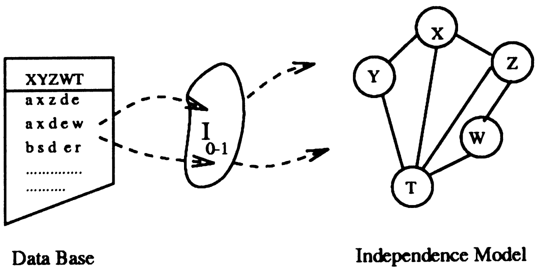

Fig. 1. Building G_{0-1}.

edge as $y \to x$. It can be easily proven that the directed graph G^{θ}_{0-1} is an I-map of the model, i.e., any d-separation statement in G^{θ}_{0-1} is a valid independence relationship in the model.

Proposition 1. *Let G_M be a dag associated to a dag-isomorphic dependency model M, and let θ be an ordering compatible with the topological ordering induced by G_M. Let us define the graph G^{θ}_{0-1} as follows:*

$$G^{\theta}_{0-1} = \{y \to x \mid x - y \in G_{0-1} \text{ and } \theta(y) < \theta(x)\}.$$

Then G^{θ}_{0-1} is an I-map of M.

Now, the problem is to look for a subgraph of G^{θ}_{0-1} which is a minimal I-map of the model. Observe that the search space could have been considerably reduced.

In the next two sections we shall see how the set of 0–1 independences can also be used to remove and fix some arrows from the graph G^{θ}_{0-1}.

2.3. Removing arrows

We are trying to answer the following question:

Can it be guaranteed that any dependence in the graph G^{θ}_{0-1} is a valid dependence in the model?

The answer is no. Let us consider the following example, where the graph in the left-hand side of Fig. 2 represents the original model. Then, the right-hand

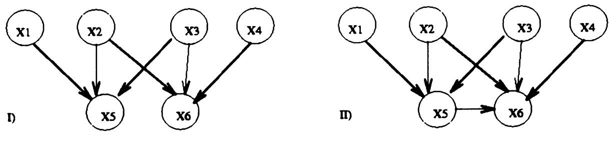

Fig. 2. Removing arrows.

side displays the learned graph G^{θ}_{0-1}. Looking at this figure we find that x_1 and x_6 are (marginally) dependent in G^{θ}_{0-1}, i.e., $\neg\langle x_1, x_6 \mid \emptyset\rangle_{G^{\theta}_{0-1}}$, whereas these two variables are (marginally) independent in the original model.

The reason for this behavior is that, since we are using only partial information, some non-necessary edges inducing false dependences can be found in G^{θ}_{0-1}. This reasoning gives us a *rule* for detecting, and also removing, some false arrows from G^{θ}_{0-1} (as $x_5 \to x_6$ in Fig. 2 (II)):

Proposition 2. *Let $G \subseteq G^{\theta}_{0-1}$ be an I-map of a dag-isomorphic dependency model M, $y \to x$ an arrow in G and $G^* = G \setminus \{y \to x\}$. If $I^{G}_{0-1} \subset I^{G^*}_{0-1} \subseteq I^{M}_{0-1}$ then G^* is still an I-map of M.* [6]

This result allows us to iteratively remove additional arrows from the current graph (the starting graph is G^{θ}_{0-1}, which is an I-map by virtue of Proposition 1) without performing conditional independence tests of order greater than 0 or 1: whenever we remove an arrow of G and find a new 0–1 d-separation statement (which was not true in G) that corresponds with a true independence statement in the model, we are sure that this arrow is superfluous and can be removed permanently.

2.4. Fixing arrows

Now, the question we are considering is:

Using the same set of 0–1 *independences, can some arrows in G^{θ}_{0-1} be fixed without doing new independence tests*?

Let us consider the following example (see Fig. 3). From left to right we show the original model M, the learned graph G^{θ}_{0-1} and the graph $G^* = G^{\theta}_{0-1} \setminus \{x \to y\}$. Then, a new 0–1 d-separation relationship appears in G^* ($\langle x, y \mid w\rangle_{G^*}$), which does not hold in the model ($I(x, y \mid w) \notin I^{M}_{0-1}$). So, the arrow $x \to y$ is necessary in the graph, in order to preserve a true dependence relationship in the model. Therefore, it can be fixed in the graph as a permanent arrow. The following proposition formalizes this idea.

Proposition 3. *Let G be an I-map of a dag-isomorphic dependency model M, $y \to x$ an arrow in G and $G^* = G \setminus \{y \to x\}$. If $I^{G^*}_{0-1} \nsubseteq I^{M}_{0-1}$ then the arrow $y \to x$ belongs to any I-map of M included in G.*

By virtue of the previous result, if after removing (provisionally) an arrow from the current graph, we find a 0–1 d-separation statement which does not correspond with a true independence statement in the model, we can safely fix

[6] Note that the symbol $\subset$ means strict inclusion.

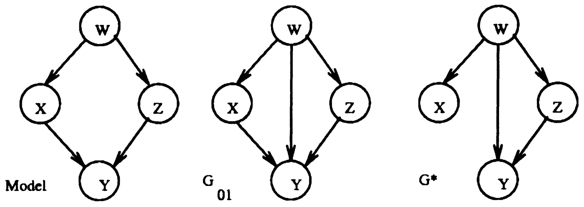

Fig. 3. Fixing arrows.

this arrow as permanent (i.e., it is not necessary to try to remove it in subsequent steps).

2.5. Using d-separating sets of minimum cardinality

Sections 2.3 and 2.4 give rules to fix and remove some arrows from the graph G^{θ}_{0-1}. For the example in Fig. 2, this process would have eliminated the incorrect arrow $x_5 \rightarrow x_6$ and would have fixed all the other arrows in G^{θ}_{0-1}, thus giving the true model as the result. Similarly, in the example in Fig. 3, the arrow $w \rightarrow y$ is the only one being not fixed. In a general case, in the current graph there still exists a set of arrows that could be neither removed nor fixed (this happens when $I^{G}_{0-1} = I^{G^*}_{0-1} \subseteq I^{M}_{0-1}$) and these arrows can be correct arrows or not, i.e., although the graph obtained after these steps (let us call it $G^{\theta o}_{0-1}$) is always an I-map of the model, it may be a non-minimal I-map. So, it is necessary to do a further step to solve this uncertainty.

Given that we know an ordering θ of the variables, a direct approach could be to search for the boundary strata [33] for each node x in the graph, that is to say, we would need to find a minimal subset B_x, $B_x \subseteq \text{Pred}_{\theta}(x)$, such that $I(x, \text{Pred}_{\theta}(x) \setminus B_x \mid B_x)$ holds in the model, where $\text{Pred}_{\theta}(x)$ is the set of nodes that precede x in the ordering θ. Finding such subsets B_x guarantees that the graph formed by designing, for each x, the variables in the set B_x as it parents, is a minimal I-map. The use of the 0–1 independences allows us to restrict the search for B_x to subsets of the parent set, $\text{Pa}(x)$, of node x in $G^{\theta o}_{0-1}$, since $B_x \subseteq \text{Pa}(x)$. Also, the minimal I-mapness is guaranteed.

This approach has two main drawbacks: First, each one of these independence statements involves all the variables that precede x in the ordering, and therefore the independence test will take an exponential time (and its result will be quite unreliable); second, the search for such subset B_x (i.e., the number of tests we have to carry out) is exponential in the cardinality of $\text{Pa}(x)$.

In order to reduce the number of independence tests and also trying to reduce the size of the d-separating sets, we propose the following approach (in [3], a similar approach was used to develop a hybrid learning algorithm): Given two nodes x and y such that there exists an arrow $y \rightarrow x$ in $G^{\theta o}_{0-1}$, instead of trying to find the set B_x, we could use the set of minimum size that still d-separates x and y in $G^{\theta o}_{0-1} \setminus \{y \rightarrow x\}$, denoted by $S_m(x,y)$. Of course, finding this set will take some additional cost, but it will be compensated by decreasing the number of independence tests needed (and in general, by also decreasing the size of the conditioning set, thus increasing the reliability of the result). The method we use for efficiently finding the set $S_m(x,y)$ is described in [1,2].

Therefore, if $I(x,y \mid S_m(x,y))$ is a true independence relationship in the model, the arrow $y \rightarrow x$ can be removed from the graph $G^{\theta o}_{0-1}$, otherwise the arrow can be fixed. The next proposition guarantees the correctness of this procedure.

Proposition 4. *Let G be an I-map of a dag-isomorphic dependency model M, $y \rightarrow x$ an arrow in G and $G^* = G \setminus \{y \rightarrow x\}$. Let $S_m(x,y)$ be a minimum d-separating set of x and y in G^* (i.e., $\langle x,y \mid S_m(x,y)\rangle_{G^*}$ and $\neg\langle x,y \mid Z\rangle_{G^*}$ $\forall Z \subset S_m(x,y)$). Then $I(x,y \mid S_m(x,y))$ is true if and only if G^* is an I-map of M.*

For example, consider a model that can be represented by the graph in Fig. 4. The graph $G^{\theta o}_{0-1}$ is displayed in Fig. 5, where solid lines represent the arrows that have been previously fixed (using Proposition 3) and dashed lines represent arrows that still are candidates to be removed or fixed from the graph.

Let us suppose that we are considering the arrow $A \rightarrow G$ (in Fig. 5) for being fixed or removed. In that case, we perform the following steps: (1) Construct G^* by removing the arrow from $G^{\theta o}_{0-1}$; (2) Search for the minimum d-separator set $S_m(G,A)$ in G^*, in this case we find that $S_m(G,A) = \{F,C\}$; (3) Test whether the independence statement $I(G,A \mid \{F,C\})$ is valid in the model; (4) Remove definitively $A \rightarrow G$ from $G^{\theta o}_{0-1}$.

Moreover, considering the resultant graph (which is an I-map of the model) and applying Proposition 3, we find that the arrow $F \rightarrow G$ could be fixed (by removing this arrow, $\langle F,G \mid C\rangle$ is a valid d-separation in the resultant graph, but $I(F,G \mid C)$ is not a valid independence statement in the model).

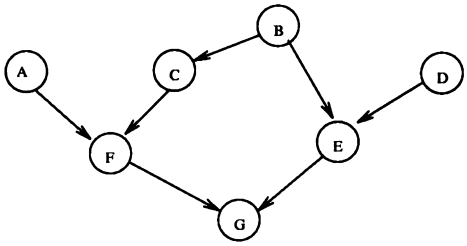

Fig. 4. Original model.

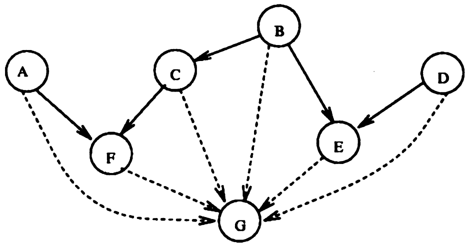

Fig. 5. $G^{\theta_0}_{0-1}$.

Table 1
Comparing orderings

Ordering d_1			Ordering d_2		
N	S_m	Ac	N	S_m	Ac
A	$\{B,C\}$	r	B	$\{C,E,D\}$	r
F	$--$	f	C	$\{B,F,A\}$	r
D	$\{E,B\}$	r	E	$\{B,D\}$	f
E	$--$	f	D	$\{E,B\}$	r
B	$\{C,E\}$	r	F	$\{A,C\}$	f
C	$\{F,B\}$	r	A	$\{F,C\}$	r

It must be noted that the ordering in which the arrows are considered is relevant for the cardinality of the separator set and also for the number of tests that we need to carry out. [7] For instance, consider again the model represented in Fig. 5. Let us suppose that we select the arrows in the ordering [8] $d_1 = A \prec F \prec D \prec E \prec B \prec C$ and the ordering $d_2 = B \prec C \prec E \prec D \prec F \prec A$.

In Table 1 we present the results obtained in this step: The first column represents the ordering in which the arrows are considered, second column represents the minimum separator set and the third column shows the action (**f**ix or **r**emove an arrow) executed. Thus, using d_1 we only need to do four conditional independence tests, each one of order two, and the arrows $F \rightarrow G$ and $E \rightarrow G$ can be fixed without making independence tests, using Proposition 3. On the other hand, using d_2 we need to make six independence tests and the cardinality of the conditioning sets has also increased.

[7] Do not confuse this ordering with θ. Now we are referring to the ordering in which the arrows are examined by the procedure based on Proposition 4.

[8] As the head node of all the arrows is the same, G, we identify an ordering of arrows with an ordering of the nodes in $\mathrm{Pa}(G)$.

```
1.- Construct the graph $G^{\theta}_{0-1}$ (Proposition 1)
2.- Construct the output graph $G$
  2.1.- $G$ <- Empty graph
  2.2.- FOR each node $x$, in ordering $\theta$, DO
      2.2.1.-  $Pa(x)$ <- Parents of $x$ in $G^{\theta}_{0-1}$
      2.2.2.-  FOR each node $p$ in $Pa(x)$ DO
                 Attempt to remove $p$ from $Pa(x)$ (Proposition 2)
                 Attempt to fix $p$ in $Pa(x)$ (Proposition 3)
      2.2.3.-  FOR each non fixed node $p$ in $Pa(x)$ DO
                 Search the set $S_m(x,p)$
                 IF $I(x,p|S_m(x,p))$ holds in $M$ (Proposition 4)
                   Remove node $p$ from $Pa(x)$
                   Attempt to fix nodes $q$ in $Pa(x)$
                 ELSE fix $p$ in $Pa(x)$
3.- Return the output graph $G$
```

Fig. 6. The learning algorithm.

2.6. The algorithm

It is necessary to remark that the proposed methodology requires an intensive use of zero- and first-order conditional d-separation tests: Each time an arrow is tentatively suppressed from the current graph (trying to decide whether to fix or to remove it) all the d-separation relationships of order 0 and 1 in the reduced graph have to be calculated. However, the number of d-separation tests needed can be reduced considerably if we organize the calculations appropriately to benefit from the ordering θ, i.e., learning the structure in an incremental way, progressing node by node using the ordering.

Thus, after computing all the 0–1 independences from the database and building the graph G^{θ}_{0-1}, we proceed as follows: to decide, for each arrow $p \rightarrow x$ in G^{θ}_{0-1}, whether the arrow can be either removed or fixed, instead of computing all the 0–1 d-separation statements in $G^{\theta}_{0-1} \setminus \{p \rightarrow x\}$, we only calculate those 0–1 d-separation statements involving nodes in $\mathrm{Pred}_{\theta}(x) \cup \{x\}$ [9] (and compare them with the corresponding 0–1 independence statements). This is equivalent to apply the results in Propositions 2 and 3 to the subgraph induced by the set $\mathrm{Pred}_{\theta}(x) \cup \{x\}$. Moreover, in case that the arrow $p \rightarrow x$ can be neither removed nor fixed, as the minimum d-separating set of x and p always belongs to $\mathrm{Pred}_{\theta}(x)$ [1], the result in Proposition 4 can also be applied to the subgraph induced by $\mathrm{Pred}_{\theta}(x) \cup \{x\}$. A sketch of the proposed learning algorithm is displayed in Fig. 6.

[9] For this purpose, a specific algorithm that performs such computation efficiently has been designed. Anyway, the time needed for computing these d-separation statements may be considerably lesser than the exponential time necessary for doing independence tests which, in general, have to consult the whole database.

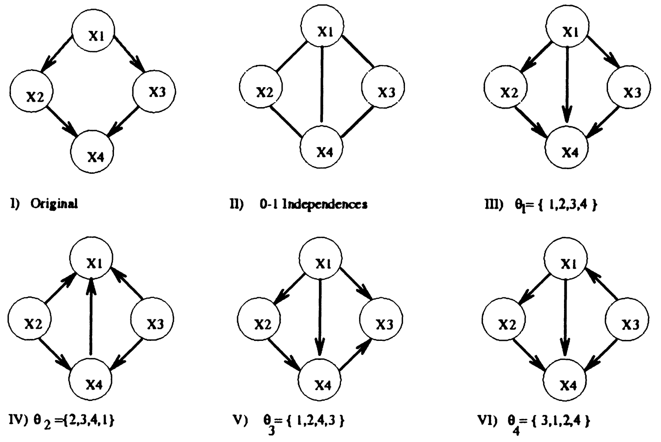
X1
X2
X3
X4
I) Original
II) 0-1 Independences
III) θ_1= { 1,2,3,4 }
IV) θ_2 ={2,3,4,1}
V) θ_3= { 1,2,4,3 }
VI) θ_4= { 3,1,2,4 }

Fig. 7. Different orderings in G_{0-1}.

2.7. *Searching a good ordering of the variables*

We have already stated that the use of previous information about the model to be recovered helps to reduce the computational complexity problem. Particularly, our algorithm assumes that a perfect ordering between the variables is known. In this section we present an algorithm that allows us to recover a 'good' (topological) ordering of the variables.

In previous sections we showed that the of 0–1 independences are highly representative for a great amount of models and how their use can reduce considerably the search space for learning belief networks. Now, we shall see how these independence statements can also be used for searching a topological ordering. If we intend to learn a perfect ordering, θ, then we find that this problem is similar to the problem of learning belief networks, and therefore it is time consuming [5,23]. However, it is possible to obtain a 'good' ordering, not necessarily optimal, by means of genetics algorithms, where the fitness operator is based on the set of 0–1 independences [9].

The initial step is similar to the one considered in the learning algorithm above, i.e., given a model, use the set of 0–1 independences, I^M_{0-1}, to obtain a skeleton of the model: the undirected graph G_{0-1}. For example, let us suppose that the original model can be represented by the graph (I) in Fig. 7. In this case, the set of 0–1 independences is $I^M_{0-1} = \{I(x_2, x_3 \mid x_1)\}$. [10] This step gives the undirected graph (II) in Fig. 7 as the output.

[10] Also included in this set is the symmetric independence relationship, i.e., $I(x_3, x_2 \mid x_1)$.

In a second step, a genetic algorithm is used with the purpose of getting the 'optimal' ordering. Each chromosome in the genetic algorithm encodes an ordering of the variables. Given a chromosome θ_i, we direct the skeleton in the following way:

If $x - y \in G_{0-1}$ and $\theta_i(x) < \theta_i(y)$ then direct the edge as $x \rightarrow y$.

In our example, we are considering the orderings $\theta_1 = \{1,2,3,4\}$, $\theta_2 = \{2,3,4,1\}$, $\theta_3 = \{1,2,4,3\}$ and $\theta_4 = \{3,1,2,4\}$. The graphs (III)–(VI) in Fig. 7 will be obtained after directing G_{0-1} using the previous orderings.

The fitness criterion we use is the degree, $g(\theta)$, in which, after directing G_{0-1} according to the ordering θ (obtaining a dag G^{θ}_{0-1}), the 0–1 independences are preserved in the dag. More precisely, we use the following measure:

$$g(\theta) = \sum_{\substack{x,y \\ x \neq y}} \left(I(x,y \mid \emptyset) \otimes \langle x,y \mid \emptyset \rangle_{G^{\theta}_{0-1}} \right) + \sum_{\substack{x,y,z \\ x \neq y \neq z}} \left(I(x,y \mid z) \otimes \langle x,y \mid z \rangle_{G^{\theta}_{0-1}} \right),$$

where $I(x,y \mid *)$ takes the value 1 if x and y are independent (marginal or conditionally) and the value 0 otherwise, and similarly $\langle x,y \mid \; * \rangle$ takes the value 1 if x and y are d-separated in the directed graph and the value 0 otherwise; $\otimes$ represents the logic exclusive-or operator. Therefore, $g(\theta) = 0$ means that I^M_{0-1} and the directed graph have the same set of zero- and first-order dependence and independence relationships. The genetic algorithm will search for those orderings that minimize the fitness criterion. For the example in Fig. 7 we have $g(\theta_1) = 0$ $g(\theta_2) = 2[(\neg I(x_2,x_3 \mid \emptyset) \otimes \langle x_2,x_3 \mid \emptyset \rangle) + (I(x_2,x_3 \mid x_1) \otimes \neg \langle x_2,x_3 \mid x_1 \rangle)]$, $g(\theta_3) = 1$ $[(I(x_2,x_3 \mid x_1) \otimes \neg \langle x_2,x_3 \mid x_1 \rangle)]$ and $g(\theta_4) = 0$, thus θ_1 and θ_4 are the best orderings.

In general, the ordering produced by the genetic algorithm is a very good approximation of a topological ordering of the original graph (if this graph is sparse) [9]. Anyway, the use of an expert checking this ordering might be useful.

3. Experiments

First, let us consider a simple but illustrative example, where the number and the order of the conditional independence tests that our algorithm needs to learn the dag-isomorphic model displayed in Fig. 8 is compared with those ones needed by the PC algorithm [30]. PC is one of the most competitive algorithms for learning belief networks, among those based on conditional independence criteria. PC starts from a complete graph, then thins that graph by removing edges with zero-order conditional independence relationships, thins again with first-order conditional independence relationships, and so on. The

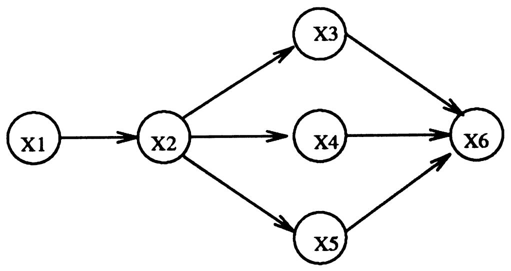

Fig. 8. Original model.

set of variables conditioned on needs only be a subset of the set of variables adjacent to one or the other of the variables conditioned.

Table 2 displays the number of tests carried out by the two algorithms for the model in Fig. 8 (for this case, depending on the ordering in which pairs of nodes are examined, PC needs a different number of tests; Table 2 gives the worst and best cases).

Observe that our algorithm carries out both less number of tests and of lower order than PC.

In order to analyze the performance of the proposed learning algorithm in a more complex and realistic situation, we have selected the Alarm network [4]. Fig. 9 shows this hand-constructed belief network (37 nodes and 46 arrows) used for diagnosis in a medical domain. This network has been considered as a benchmark for evaluating learning algorithms.

The idea of the first (rather unrealistic) experiment with Alarm is to compare the number and size of the conditional independence tests (of order greater than one) needed by our learning algorithm with those needed when an 'ideal' (independence-based) learning algorithm is used. Both algorithms will start from the graph $G^{\emptyset}_{0-1}$ (containing 101 arrows). Previously, we shall describe such 'ideal' process: Consider that given two variables in the model as the input, an 'oracle' can give, as the output, the true minimal separator set in the *real model*

Table 2
Comparison with PC

Order	Number of tests	
	Ours	PC
0	15	15
1	60	36–53
2	0	39
3	1	5–23
Total	76	95–130

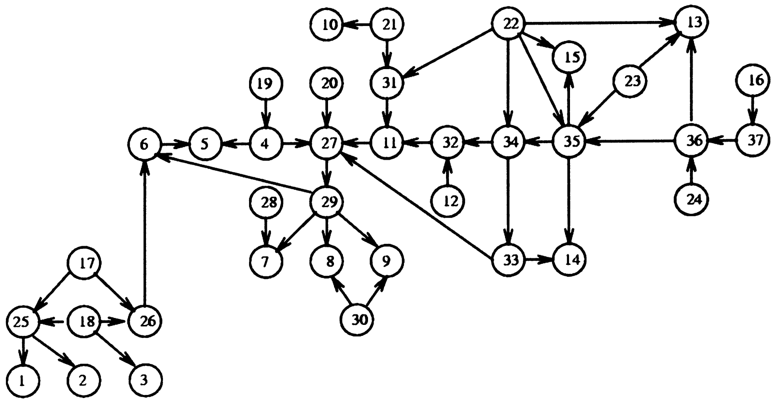

Fig. 9. Alarm network.

that would make the two variables (conditionally) independent. This implies, of course, that this oracle knows the real model to be recovered.

In order to remove those arrows that do not belong to the Alarm network (the real model) the 'ideal' algorithm would need 55 conditional independence tests (54 of order two and 1 of order three). For example, to remove the arrow $22 \to 11$ from G^{θ}_{0-1} we would test that $I(11, 22 \mid \{31, 32\})$ holds.

Now, we shall consider the number and the order of the conditional independence tests used by our learning algorithm. As it was commented in Section 2.5, given a node x, the ordering in which its potential parents $\mathrm{Pa}(x)$ are considered is relevant to our purpose. In all the experiments we used the following heuristic rule:

Select the node p in $\mathrm{Pa}(x)$ *having minimum cardinality separator set* $S_m(p, x)$. *If there exists a tie, select the node p such that, by removing the arrow* $p \to x$ *from the graph, make it possible to decrease the size of some set* $S_m(p_i, x)$, $p_i \in \mathrm{Pa}(x)$.

The obtained result is the following: All the arrows that actually belong to the Alarm network have been fixed (using Proposition 3) is some step of the algorithm (performing no additional tests for this purpose). So, we only need to perform 55 conditional independence tests: 40 of order two, 14 of order three and, finally, 1 of order four. Comparing these results with the ones obtained by the 'ideal' process, we use in the 74% of times a minimum cardinality conditional independence test, whereas when we need greater separating sets we only have to use one (14 times) or two (once) additional variables. These results give us an idea of the performance of the proposed algorithm (in an ideal situation, i.e., assuming that the results of the independence tests are never wrong), with respect to the number and the order of the independence tests necessary.

The following experiments aim to test our algorithm in more realistic circumstances. We use four databases DB1, DB2, DB3 and DB4 obtained by selecting, respectively, the first 5000, 6000, 7000 and 10 000 cases of the original Alarm data set [19].

In order to obtain the conditional independence statements from the databases, we use the mutual information test [21]. It uses the Kullback–Leibler cross-entropy, which measures the mutual information (or degree of dependence) between variables x_i and x_j conditioned to the set of variables Z

$$\mathrm{Inf}(x_i, x_j \mid Z) = \sum_{\mathbf{x}_i, \mathbf{x}_j, \mathbf{Z}} P(\mathbf{x}_i, \mathbf{x}_j, \mathbf{Z}) \log \frac{P(\mathbf{x}_i, \mathbf{x}_j \mid \mathbf{Z})}{P(\mathbf{x}_i \mid \mathbf{Z}) P(\mathbf{x}_j \mid \mathbf{Z})},$$

where P represents the probability distribution estimated from the database. The value $2 * N * \mathrm{Inf}(x_i, x_j \mid Z)$ approximates a χ^2 distribution with $|Z|(|x_i| - 1)(|x_j| - 1)$ degrees of freedom, where N is the number of cases in the database and $|H|$ represents the number of possible values for the set of variables H. In our experiments, the confidence level is set to 99%.

In our following experiment we still consider that a perfect ordering of the variables is previously known (the parents of any node x in the Alarm network appear before x in the ordering). Therefore, our algorithm takes a database of cases and the ordering as the input and gives the structure of the learned network as the output.

Before presenting the experimental results, it is necessary to take into account the following fact: To safely fix and remove arrows from the graph G^{θ}_{0-1} using only 0–1 independences, it is necessary that G^{θ}_{0-1} is an I-map of the model (as prescribed by Propositions 2 and 3). However, this condition may be false when we obtain the independence statements from a database. For example, considering the database with 10 000 cases (DB4), the marginal mutual information for nodes 21 and 31 is 7×10^{-6}. So, these variables are considered as marginally independent and the arrow $21 \rightarrow 31$ is removed in the initial step. Moreover, both nodes 21 and 10 are also found to be marginally independent of any other node. This fact makes the statement $\langle 21, 22 \mid 31 \rangle$ true in the graph, whereas testing the corresponding relationship $I(21, 22 \mid 31)$ in the model (the database) produces a dependence statement (the mutual information is 0.0016). In other words, the graph is not an I-map of the model. In these circumstances, if we would apply Proposition 3 we would have to (erroneously) fix all the arrows in G^{θ}_{0-1} (because after removing any arrow we would find a 0–1 d-separation statement which does not correspond with a true independence). To solve this problem, when removing and fixing arrows using 0–1 independences we will proceed, in a heuristic way, by considering (momentarily) that

Table 3
Mutual information values for missing and added arcs

Database	Missing arcs		Added arcs
	$21 \to 31$	$12 \to 32$	$27 \to 9$
DB1	6×10^{-6}	7×10^{-5}	–
DB2	1.2×10^{-5}	$2.9 \times 10^{-}5$	–
DB3	2.2×10^{-5}	2.3×10^{-5}	0.0012
DB4	6×10^{-6}	3×10^{-5}	–

Table 4
Results of the experiment using a perfect ordering

Database	0–1 indep.		Cond. indep. tests				Heuristic	
	Number	Fixed	#2	#3	#4	#5	#2	#3
DB1	63	31	18	6	5	1	14	9
DB2	62	32	17	6	5	1	13	9
DB3	63	33	16	7	5	1	13	9
DB4	64	31	21	5	3	2	15	9

all the 0–1 independences represented in the graph G^{θ}_{0-1} are valid independence statements in the model (i.e., we proceed as if G^{θ}_{0-1} were an I-map of the model).

Table 3 summarizes the obtained results, where we are considering the original Alarm network as the reference in the comparison. In all the experiments we have lost two true arrows ($21 \to 31, 12 \to 32$) since these variables become marginally independent in the databases. Also, using DB3, an additional arrow was included, because the independence test $I(27, 9 \mid 29)$ failed on this database. In Table 3 we show the mutual information values obtained.

In Table 4 we present the number and the order of the conditional independence tests needed. Second column represents the number of arrows in G^{θ}_{0-1} after removing all the 0–1 independence statements (no additional arrow has been removed using these independences). Third column represents the number of fixed arrows in this step. Columns 4–7 represent the number of tests carried out, of orders 2, 3, 4 and 5, respectively, when we do not use any specific criterion to determine the ordering in which the parents of a given node are examined. Finally, the last two columns represent the number of tests needed when we use the aforementioned heuristic rule. As it could be expected, the number and the order of the conditional independence tests have been considerably reduced by using this heuristic rule. For example, instead of testing $I(22, 11 \mid 31, 13, 35, 15, 34)$ we test $I(22, 11 \mid 31, 34)$.

In the next experiment we do not assume any knowledge about the causal ordering of the variables. In that case, we use the ordering obtained by the

Table 5
Fitness values obtained by the genetic algorithm

Database	Fitness
DB1	97.4%
DB2	97.7%
DB3	97.1%
DB4	96.6%

genetic algorithm described in Section 2.7. The parameters used when running the algorithm were: population size 61, crossover probability 1, mutation probability 0.05 and number of generations 1000. Table 5 shows the fitness values (percentage of 0–1 independences preserved) for the best orderings obtained by the genetic algorithm, for each database.

It is necessary to remark that there are many different orderings having the same fitness value. The values in Table 5 has been reached by the following ordering:

20, 29, 21, 28, 17, 18, 31, 37, 23, 25, 24, 36, 22, 34, 10, 35, 12,
32, 1, 30, 33, 13, 15, 26, 11, 9, 2, 3, 4, 27, 14, 16, 6, 7, 8, 19, 5

Moreover, two general comments about the orderings obtained in our experiments must be done: (i) There is no preference on the relative ordering among variables $35, 15, 34$ (all the orderings produced by changing just the relative positions of these variables attain the same fitness value). This is because there are not 0–1 independence relationships among these variables. (ii) Variables 29 and 9 should precede variable 27 in the ordering (although variable 27 precedes both variables 29 and 9 in the Alarm's original ordering): if it is not the case, then the fitness value decreases to approx. 81%. The reason is that the 0–1 dependence relationships between both variables, 29 and 9, and many of the variables which are ancestors of 27 in the original Alarm network are not reflected in our databases.

The structure of the networks learned using these orderings is strongly related with the two facts above. In the comparison we take as the reference the network learned by our algorithm when using the 'correct' ordering. Considering (ii), we found that the edge 27–29 was inverted in all the experiments (for DB3, we also got the arrow $9 \leftarrow 27$). Depending on the specific ordering being used, some other arrows can be inverted with respect to the original Alarm network; however, they are not incorrectly inverted because give rise to isomorphical structures (for example, using the ordering above, we get the arrow $4 \rightarrow 19$; note that the direction of this arrow does not matter).

On the other hand, considering (i) and the different relative orderings of the variables $35, 15$ and 34, for DB4 (for the other databases the results are

Table 6
Results for DB4 using different orderings

Ordering	Added	Missing	Inverted
$35 \prec 34 \prec 15$	None	None	27 – 29
$35 \prec 15 \prec 34$	None	None	27 – 29
$34 \prec 35 \prec 15$	23 – 4; 36 – 34	None	34 – 35; 27 – 29
$15 \prec 35 \prec 34$	23 – 15; 36 – 15	None	15 – 35; 27 – 29
$34 \prec 15 \prec 35$	34 – 15; 36 – 1	36 – 35	34 – 35; 15 – 35
	36 – 34; 23 – 34		27 – 29
$15 \prec 34 \prec 35$	15 – 34; 23 – 15	36 – 35; 23 – 35	15 – 35; 27 – 29
	23 – 34; 36 – 15	34 – 35	
	36 – 34		

similar), we obtain the results displayed in Table 6. On the average, we add 2.16 edges, miss 0.66 edges and invert 1.83 edges.

4. Concluding remarks

In this paper we have explored the potentiality and expressiveness of low order conditional independence relationships to the development of learning algorithms for belief networks. There exist several reasons to take this approach. The first one is reliability: as marginal independences involve only two variables and conditional independences of order one involve three variables, the independence values are highly reliable when they are tested from a medium-sized database. The second reason is efficiency: there exist $O(n^2)$ independence values for $I(x,y \mid \emptyset)$ and $O(n^3)$ values for $I(x,y \mid z)$ (and these values can be computed efficiently). We have shown that the information provided by conditional independence tests of order 0 and 1 can be used, not only to remove arrows whose terminal nodes exhibit this kind of independences, but also to remove additional arrows linking nodes displaying higher order independence relationships, as well as to determine a set of arrows that necessarily must belong to the network. We have proposed a learning algorithm that exploits all the information provided by the set of 0–1 independences. Our algorithm also needs to use a number of independence tests (at most $O(n^2)$) of order greater than one, but this number may be quite small for sparse graphs; moreover, the priority is always given to low order tests. We believe that our method to maintain the order of the tests as low as possible, based on d-separating sets of minimum cardinality, is quite useful (the cost of calculating these sets compared with the cost of performing higher order tests is minimum). Our algorithm guarantees finding a perfect map of the dependency model, provided that the model is dag-isomorphic and we start from a perfect ordering of the variables (and assuming valid statistical decisions).

The preliminary experiments show that our learning algorithm may be quite efficient and competitive with other existing algorithms, even in those cases where the theoretical conditions which guarantee its soundness are not satisfied. Nevertheless, a more systematic experimentation is necessary to reach definitive conclusions. For future work, we also plan to study conditions of applicability of our algorithm weaker than dag-isomorphism, as well as heuristic rules to determine the ordering in which arrows have to be examined in order to be fixed or removed.

Acknowledgements

This work has been supported by the Spanish Comisión Interministerial de Ciencia y Tecnología (CICYT) under Project TIC96-0781. We are grateful to Silvia Acid for her help with the algorithm implementation.

Appendix A

In this appendix, we include the proofs of all the results stated in the main body of the article.

Proposition 1. *Let G_M be a dag associated to a dag-isomorphic dependency model M, and let θ be an ordering compatible with the topological ordering induced by G_M. Let us define the graph G^{θ}_{0-1} as follows:*

$$G^{\theta}_{0-1} = \{y \to x \mid x - y \in G_{0-1} \text{ and } \theta(y) < \theta(x)\}.$$

Then G^{θ}_{0-1} is an I-map of M.

Proof. Let us suppose that, for some nodes x, y and some subset H, it is $\langle x, y \mid H\rangle_{G^{\theta}_{0-1}}$ but $\neg I(x, y \mid H)$. Then, as G_M is an I-map of M, we have $\neg\langle x, y \mid H\rangle_{G_M}$. So, in G_M there is a trail from x to y opened by H. For every arrow $a \to b$ in this trail we have $\neg\langle a, b \mid \emptyset\rangle_{G_M}$ and $\neg\langle a, b \mid z\rangle_{G_M}$ $\forall z$, hence $\neg I(a, b \mid \emptyset)$ and $\neg I(a, b \mid z)$ $\forall z$. Therefore, these arrows $a \to b$ must also belong to G^{θ}_{0-1}. In this way, in G^{θ}_{0-1} we find a trail from x to y opened by H, i.e., $\neg\langle x, y \mid H\rangle_{G^{\theta}_{0-1}}$, in contradiction with the hypothesis. $\square$

Proposition 2. *Let $G \subseteq G^{\theta}_{0-1}$ be an I-map of a dag-isomorphic dependency model M, $y \to x$ an arrow in G and $G^* = G \setminus \{y \to x\}$. If $I^{G}_{0-1} \subset I^{G^*}_{0-1} \subseteq I^{M}_{0-1}$ then G^* is still an I-map of M.*

Proof. Let us prove the result for the case in which the new 0–1 d-separation statement found in G^* is a marginal d-separation, i.e., we assume that for some pair of nodes u and v it is $\langle u,v \mid \emptyset\rangle_{G^*}$, $\neg\langle u,v \mid \emptyset\rangle_G$ and $I(u,v \mid \emptyset)$ (the case $\langle u,v \mid w\rangle_{G^*}$, $\neg\langle u,v \mid w\rangle_G$ and $I(u,v \mid w)$ can be proven analogously).

Let us suppose that G^* is not an I-map. Then, as G is an I-map, the arrow $y \to x$ must belong to the minimal I-map included in G (which in this case coincides with the perfect map G_M, since M is dag-isomorphic).

From $\neg\langle u,v \mid \emptyset\rangle_G$ we deduce that in G there is a trail C linking u and v without head to head nodes. If $y \to x$ does not belong to C, then C is also a trail without head to head nodes in G^*, hence $\neg\langle u,v \mid \emptyset\rangle_{G^*}$, in contradiction with the hypothesis. So, the arrow $y \to x$ belongs to C.

On the other hand, as $y \to x \in G$, this arrow was not removed when testing 0–1 independences, so that we have $\neg I(y,x \mid \emptyset)$ and $\neg I(y,x \mid \gamma)$ $\forall\gamma$. Moreover, we know that in G^* there are not 0–1 d-separations which correspond with false 0–1 independence statements ($I_{0-1}^{G^*} \subseteq I_{0-1}^{M}$); therefore, we deduce $\neg\langle y,x \mid \emptyset\rangle_{G^*}$ and $\neg\langle y,x \mid \gamma\rangle_{G^*}$ $\forall\gamma$. So, in G^* (and also in G_M) there is at least a trail D from x to y without head to head nodes. By composing D with C (excluding $y \to x$) we get another trail from u to v in G^*. As this trail has to contain at least a head to head node and neither C nor D contain it, the head to head node has to be the node t where the two trails join. The situation is as follows:

$$\begin{tikzcd}[column sep=small] u \arrow[r, dash] & \cdots \arrow[r, dash] & a \arrow[r] & t \arrow[r] \arrow[rrr, dotted, leftrightarrow, bend right] & \cdots \arrow[r] & y \arrow[r] & x \arrow[r] & \cdots \arrow[r] & v. \end{tikzcd}$$

As the arrow $a \to t$ has not been removed from G, it is $\neg I(a,t \mid \emptyset)$ and $\neg I(a,t \mid \gamma)$ $\forall\gamma$. Therefore, there are trails without head to head nodes linking a and t in G_M (the arrow $a \to t$ could be such a trail, in case that $a \to t$ were a true arrow in G_M). By composing one of these trails from a to t with the trail $t \to \cdots \to y$ and with the arc $y \to x$, we get a trail from a to x in G_M without head to head nodes, i.e., $\neg I(a,x \mid \emptyset)$.

From $\neg I(a,t \mid \gamma)$ we can reason as follows: we can find (in G_M) a trail C_γ from a to t not blocked by γ. For any γ which is not one of the nodes in the trail $t \to \cdots \to y \to x$, if we compose C_γ with this trail, we get a trail linking a and x not blocked by γ. For any γ which is a node in the trail $t \to \cdots \to y \to x$, by composing C_γ with the trail $t \leftarrow \cdots \to x$ (which does not pass through y) we also obtain a trail from a to x not blocked by γ. Therefore, we have $\neg I(a,x \mid \gamma)$ $\forall\gamma$.

So, the arrow $a \to x$ has not been removed by using 0–1 independences. If $a \to x$ is still an arc of G (and also of G^*), then we find in G^* the trail $u - \cdots - a \to x \to \cdots \to v$ without head to head nodes, hence $\neg\langle u,v \mid \emptyset\rangle_{G^*}$, in contradiction with the hypothesis. If the arc $a \to x$ has been removed from G, we can repeat the previous reasoning, where now the node a plays the role of node y (i.e., $\neg\langle a,x \mid \emptyset\rangle_{G^*}$, $\neg\langle a,x \mid \gamma\rangle_{G^*}, \ldots$) and we would find another node b in the trail C, nearer to u than a, such that $\neg I(b,x \mid \emptyset)$ and $\neg I(b,x \mid \gamma)$ $\forall \gamma$. By repeating this reasoning we necessarily find a trail in G^* linking u and v without head to head nodes, thus contradicting the hypothesis. □

Proposition 3. *Let G be an I-map of a dag-isomorphic dependency model M, $y \rightarrow x$ an arrow in G and $G^* = G \setminus \{y \rightarrow x\}$. If $I_{0-1}^{G^*} \not\subseteq I_{0-1}^{M}$ then the arrow $y \rightarrow x$ belongs to any I-map of M included in G.*

Proof. Let $\langle u,v \mid \emptyset \rangle_{G^*}$ or $\langle u,v \mid w \rangle_{G^*}$ be d-separation statements which belong to $I_{0-1}^{G^*}$ but do not correspond with true independence statements in M. Let H be an I-map of M included in G. If $y \rightarrow x$ does not belong to H, then $H \subseteq G \setminus \{y \rightarrow x\} = G^*$. From $\langle u,v \mid \emptyset \rangle_{G^*}$ or $\langle u,v \mid w \rangle_{G^*}$ we deduce $\langle u,v \mid \emptyset \rangle_{H}$ or $\langle u,v \mid w \rangle_{H}$, and therefore, as H is an I-map, we have $I(u,v \mid \emptyset)$ or $I(u,v \mid w)$, in contradiction with the hypothesis. □

Proposition 4. *Let G be an I-map of a dag-isomorphic dependency model M, $y \rightarrow x$ an arrow in G and $G^* = G \setminus \{y \rightarrow x\}$. Let $S_m(x,y)$ be a minimum d-separating set of x and y in G^* (i.e., $\langle x,y \mid S_m(x,y) \rangle_{G^*}$ and $\neg \langle x,y \mid Z \rangle_{G^*}$ $\forall Z \subset S_m(x,y)$). Then $I(x,y \mid S_m(x,y))$ is true if and only if G^* is an I-map of M.*

Proof.
- *Sufficient condition.* It is immediate, because we have $\langle x,y \mid S_m(x,y) \rangle_{G^*}$ and G^* is an I-map, hence $I(x,y \mid S_m(x,y))$.
- *Necessary condition.* Let $\Pi_G(x)$ be the parent set of x in G. As $y \in \Pi_G(x)$, the parent set of x in G^* is $\Pi_{G^*}(x) = \Pi_G(x) \setminus \{y\}$. Let us define $R(x,y) = \Pi_{G^*}(x) \setminus S_m(x,y)$.

  (i) First, we are going to prove that $\langle y, R(x,y) \mid S_m(x,y) \rangle_{G^*}$: If it is $\neg \langle y, R(x,y) \mid S_m(x,y) \rangle_{G^*}$, then there exists a trail in G^* linking y and some node $z \in R(x,y)$ which is not blocked by $S_m(x,y)$. Therefore, this trail, composed with the arc $z \rightarrow x$ is a trail in G^* linking y and x not blocked by $S_m(x,y)$, in contradiction with the hypothesis.

  (ii) Second, we are going to prove that $\langle y, R(x,y) \mid S_m(x,y) \rangle_{G}$: Let us suppose $\neg \langle y, R(x,y) \mid S_m(x,y) \rangle_{G}$. Then, there is a trail in G from y to some node $z \in R(x,y)$ not blocked by $S_m(x,y)$. If the arrow $y \rightarrow x$ does not belong to this trail, then it is also a trail in G^* not blocked by $S_m(x,y)$, hence $\neg \langle y, R(x,y) \mid S_m(x,y) \rangle_{G^*}$, in contradiction with the result in (i). On the other hand, if $y \rightarrow x$ is an arrow of the trail, as $z \in \Pi_{G^*}(x)$, the trail has to contain at least a head to head node. Consider the head to head node u nearest to x in the trail. In this case, in order to open the trail, u (or one of its descendants) must belong to $S_m(x,y)$. Then, we have a directed path from x to a node in $S_m(x,y)$, but this situation is not possible because the nodes in $S_m(x,y)$ have to be ancestors of x [1].

(iii) Third, it is clear that $\langle x, S_m(x,y) \setminus \Pi_{G^*}(x) \mid \Pi_{G^*}(x) \cup \{y\}\rangle_G$, because the nodes in $S_m(x,y) \setminus \Pi_{G^*}(x)$ are ancestors (but they are not parents) of x in G, hence they are d-separated from x by $\Pi_G(x) = \Pi_{G^*}(x) \cup \{y\}$.

(iv) Fourth, let us prove $I(x,y \mid \Pi_{G^*}(x))$: As G is an I-map of M, from the results in (ii) and (iii) we obtain $I(y, R(x,y) \mid S_m(x,y))$ and $I(x, S_m(x,y) \setminus \Pi_{G^*}(x) \mid \Pi_{G^*}(x) \cup \{y\})$. From $I(y, R(x,y) \mid S_m(x,y))$ and $I(x,y \mid S_m(x,y))$ we obtain (using the property of *composition*, which is true for dag-isomorphic models) $I(y, \{x\} \cup R(x,y) \mid S_m(x,y))$, and now using *weak union* we get $I(y,x \mid S_m(x,y) \cup R(x,y)) \equiv I(y,x \mid \Pi_{G^*}(x) \cup (S_m(x,y) \setminus \Pi_{G^*}(x)))$. From this last statement, together with $I(x, S_m(x,y) \setminus \Pi_{G^*}(x) \mid \Pi_{G^*}(x) \cup \{y\})$, we get (using *intersection*) $I(x, \{y\} \cup S_m(x,y) \setminus \Pi_{G^*}(x) \mid \Pi_{G^*}(x))$; now, using *decomposition*, we obtain $I(x,y \mid \Pi_{G^*}(x))$.

Now, we are in position to prove that G^* is an I-map of M. As G is an I-map, we have $I(x, \mathrm{Pred}_\theta(x) \setminus \Pi_G(x) \mid \Pi_G(x))$ (which may also be expressed as $I(x, \mathrm{Pred}_\theta(x) \setminus \Pi_G(x) \mid (\Pi_G(x) \setminus \{y\}) \cup \{y\})$). This statement, together with $I(x,y \mid \Pi_{G^*}(x))$ (i.e., $I(x,y \mid \Pi_G(x) \setminus \{y\})$), produces $I(x, (\mathrm{Pred}_\theta(x) \setminus \Pi_G(x)) \cup \{y\} \mid \Pi_G(x) \setminus \{y\})$ using *contraction*, and this last statement can be expressed as $I(x, \mathrm{Pred}_\theta(x) \setminus \Pi_{G^*}(x) \mid \Pi_{G^*}(x))$. For any other node $z \neq x$, $\Pi_{G^*}(z) = \Pi_G(z)$, hence $I(z, \mathrm{Pred}_\theta(z) \setminus \Pi_{G^*}(z) \mid \Pi_{G^*}(z))$ is a true independence statement. So, G^* is an I-map of M. $\square$

References

[1] S. Acid, L.M. de Campos, An algorithm for finding minimum d-separating sets in belief networks, in: E. Horvitz, F. Jensen (Eds.), Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, 1996, pp. 3–10.

[2] S. Acid, L.M. de Campos, An algorithm for finding minimum d-separating sets in belief networks, Tech. Report #96-02-14 DECSAI, Universidad de Granada, 1996.

[3] S. Acid, L.M. de Campos, Benedict: An algorithm for learning probabilistic belief networks, in: Proceedings of the Sixth Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU), 1996, pp. 979–984.

[4] I. Beinlich, Suermondt, H.R. Chavez, G. Cooper, The ALARM monitoring system: a case study with two probabilistic inference techniques for belief networks, in: Proceedings of Artificial Intelligence in Medical Care, 1989, pp. 247–256.

[5] R. Bouckaert, Optimizing causal orderings for generating dags from data, in: D. Dubois, M.P. Wellman, D.B. D'Ambrosio (Eds.), Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, 1992, pp. 9–16.

[6] W. Buntine, A guide to the literature on learning probabilistic networks from data, IEEE Trans. Knowledge Data Engineering 8 (1996) 195–210.

[7] L.M. de Campos, Independency relationships and learning algorithms for singly connected networks, J. Exp. Theoret. Artificial Intell. 10 (4) (1998) 511–549.

[8] L.M. de Campos, J.F. Huete, On the use of independence relationships for learning simplified belief networks, Internat. J. Intell. Sys. 12 (7) (1997) 495–522.

[9] L.M. de Campos, J.F. Huete, Aproximación de una ordenación de variables en redes causales mediante algoritmos genéticos (in Spanish), Inteligencia Artificial 4 (4) (1998) 30–39.

[10] J. Cheng, D.A. Bell, W. Liu, An algorithm for Bayesian belief network construction from data, in: Proceedings of AI & STAT'97, 1997, pp. 83–90.
[11] D. Chickering, Learning bayesian networks is NP-Complete, in: D. Fisher, H.J. Lenz (Eds.), Learning from Data, Lecture Notes in Statistics, 112, Springer, Berlin, 1995, pp. 121–130.
[12] G.F. Cooper, E. Herskovits, A bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347.
[13] R.M. Fung, S.L. Crawford, Constructor: A system for the induction of probabilistic models, in: Proceedings of the Eighth National conference on Artificial Intelligence, MIT Press, Boston, MA, 1990, pp. 762–769.
[14] D. Geiger, A. Paz, J. Pearl, Learning simple causal structures, Internat. J. Intell. Sys. 8 (1993) 231–247.
[15] D. Haussler, Decision theoretic generalizations of the PAC model for neural net and other learning applications, Inform. control 100 (1) (1992) 78–150.
[16] D. Heckerman, D. Geiger, D.M. Chickering, Learning bayesian networks: the combination of knowledge and statistical data, Machine Learning 20 (1995) 197–243.
[17] D. Heckerman, A. Mamdani, M. Wellman, Real-world applications of bayesian networks: Introduction, Communications of the ACM 38 (3) (1995).
[18] D. Heckerman, Bayesian network for data mining, Data Mining and Knowledge Discovery 1 (1) (1997) 79–119.
[19] E.H. Herskovits, Computer-based probabilistic networks construction, Ph.D. Thesis, Medical Information Sciences, Stanford University, 1991.
[20] E.H. Herskovits, G.F. Cooper, Kutató: An entropy-driven system for the construction of probabilistic expert systems from databases, in: P. Bonissone (Ed.), Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence, Cambridge, 1990, pp. 54–62.
[21] S. Kullback, Information Theory and Statistics, Dover, New York, 1968.
[22] W. Lam, F. Bacchus, Using causal information and local measures to learn bayesian belief networks, in: D. Heckerman, A. Mamdani (Eds.), Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, 1993, pp. 243–250.
[23] P. Larrañaga, C.M. Kuijpers, R.H. Murga, Y. Yurramendi, Learning bayesian network structure by searching for the best ordering with genetic algorithms, IEEE Trans. Sys. Man Cybernetics – Part A: Sys. Humans 26 (4) (1996) 487–493.
[24] S.L. Lauritzen, D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their applications to expert systems (with discussion), J. Roy. Stat. Soc. (B) 50 (1988) 157–224.
[25] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible inference, Morgan Kaufmann, San Mateo, 1988.
[26] J. Pearl, T. Verma, A theory of inferred causation, in: J.A. Allen, R. Fikes, E. Sandwall (Eds.), Principles of Knowledge Representation and Reasoning, Proceedings of the Second International Conference, Morgan and Kaufmann, San Mateo, 1991, pp. 441–452.
[27] M. Singh, M. Valtorta, Construction of bayesian networks structures from data: a survey and an efficient algorithm, Internat. J. Approximate Reasoning 12 (1995) 111–113.
[28] D. Spiegelhalter, A. Dawid, S.L Lauritzen, R. Cowell, Bayesian analysis in expert systems, Statistical Sci. 8 (1993) 219–283.
[29] P. Spirtes, C. Glymour, R. Scheines, An algorithm for fast recovery of sparse causal graphs, Soc. Sci. Comput. Rev. 9 (1991) 62–72.
[30] P. Spirtes, C. Glymour, R. Scheines, Causation, Prediction and Search, Lecture Notes in Statistics, 81, Springer, New York, 1993.
[31] P. Spirtes, T. Richarson, C. Meek, Learning Bayesian networks with discrete variables from data, in: Proceedings of the First International Conference on Knowledge Discovery and Data Mining, 1995, pp. 294–299.

[32] S. Srinivas, S. Russell, A. Agogino, Automated construction of sparse bayesian networks from unstructured probabilistic models and domain information, in: M. Henrion, R.D. Shachter, L.N. Kanal, J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence vol. 5, North-Holland, Amsterdam, 1990, pp. 295–308.

[33] T. Verma, J. Pearl, Causal networks: Semantics and expressiveness, in: R.D. Shachter, T.S. Levitt, L.N. Kanal, J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence, vol. 4, North-Holland, Amsterdam, 1990, pp. 69–76.