Efficient Score-Based Learning of Equivalence Classes of Bayesian Networks

Paul Munteanu and Denis Cau

Centre de Recherche du Groupe ESIEA
38 rue des Docteurs Calmette et Guérin
53 000 Laval, France
{munteanu, cau}@esiea-ouest.fr

Abstract. The use of bayesian networks for knowledge discovery requires learning algorithms which emphasize not only the predictive power but also the structural fidelity of the discovered networks.
Previous work on score-based methods for learning equivalence classes of bayesian networks showed that they generally provide better results than classical algorithms, that explore the space of bayesian networks. However, they are considerably slower, mainly because they use more complicated search operators and because they have to build instances of the equivalence classes in order to check their consistency and in order to calculate their score.
We propose here a new greedy learning algorithm that explores the space of equivalence classes with a reduced set of operators and realizes the verification of the consistency and the computation of the score without any need for instantiation. We show on five experimental tasks that this algorithm is rather efficient, obtains better scores and discovers structures closer to the "gold-standard" than classical greedy and tabu search in the space of bayesian networks.

1 Introduction

Learning bayesian networks from data is one of the most ambitious approaches to Knowledge Discovery in Databases. Unlike most other data mining techniques, it does not focus its search on a particular kind of knowledge but aims to found all the (probabilistic) relations which hold between the considered variables.

From a statistical viewpoint, a bayesian network efficiently encodes the joint probability distribution of the variables describing an application domain. This kind of knowledge allows making rational decisions involving any arbitrary subset of these variables on the basis of the available knowledge about another arbitrary subset of variables.

Moreover, bayesian networks may be represented in a graphical annotated form which seems quite natural to human experts for a large variety of applications. The nodes of a bayesian network correspond to domain variables and the edges which connect the nodes correspond to direct probabilistic relations between these variables. Under certain assumptions [1], these relations have causal semantics (a directed edge $A \rightarrow B$ may be interpreted as *A is a direct cause of* B), while most other data mining approaches deal exclusively with correlation.

There are two main approaches to learning bayesian networks with unknown structure. The first one is to build the network according to the conditional independence relations found in data (*e.g.,* [1]). Traditionally, these methods aim at discovering causal relations between the variables and, therefore, emphasize the structural fidelity of the bayesian networks they learn. Unfortunately, they suffer from the lack of reliability of high-dimensional conditional independence tests.

The other approach to learning bayesian networks is to define an evaluation function (or score) which accounts for the quality of candidate networks with respect to the available data and to use some kind of search algorithm in order to find, in a "reasonable" amount of time, a network with an "acceptable" score (we use the terms "reasonable" and "acceptable" because this learning task have been proven to be NP-hard for the evaluation functions mentioned in the following section). These algorithms are less sensitive to the quality of the available data and their results can be successfully used in various decision making tasks.

However, as we will see in the following section, the exploration of the space of bayesian network structures by a greedy search algorithm may end with a structure which fails to reveal some independence relations between the variables and, therefore, may be rather different from the true one. The space of equivalence classes of bayesian network structures seems to be better suited for this kind of search. Learning algorithms which explore this space have already been proposed by some authors, as described in section 3. Unfortunately, these algorithms are considerably slower than classical ones, mainly because they use more complicated search operators, and because they have to build instances of the equivalence classes in order to check their consistency and in order to calculate their score.

Section 4 introduces a new algorithm, EQ1, that explores the space of equivalence classes with a reduced set of operators and realizes the verification of the consistency and the computation of the score without instantiating the equivalence classes constructed during the search. The experimental results presented in section 5 confirm the fact that EQ1 is able to efficiently produce better results than classical greedy and tabu search in the space of bayesian networks.

2 Heuristic Search in the Space of Bayesian Networks

Like in many other machine learning approaches, the quality of a bayesian network with respect to the available data is evaluated on the basis of a score issued from the information theory or from bayesian inference. Some of these scores, like MDL [2] and BDe [3] have been proven to be asymptotically correct and have some nice mathematical properties that can be exploited by the search algorithms:

– the score of a bayesian network may be expressed as a sum of local scores involving only a node and its parents;

– the bayesian networks belonging to the same equivalence class (*i.e.*, representing the same conditional independence relations) have the same score.

Since the unconstrained learning of bayesian networks is NP-hard, most authors propose the use of heuristic search algorithms, which explore the space of bayesian network structures. The transformation operators are the addition, the suppression, and the reversal of edges, submitted to the constraint that the resulting network contains no cycle (on the basis of score decomposability, these operators ensure that no more than two local scores have to be re-evaluated in order to evaluate the resulting network). The search strategy is based on some general-purpose method (greedy search, simulated annealing or tabu search).

Unfortunately, there are many local optima in the space of bayesian networks and heuristic search algorithms may easily be trapped in one of them. The main reason for this difficulty is the equality of the score of equivalent networks. We illustrate this statement by a learning task which is very simple but nevertheless confusing for greedy search (fig. 1). In this example, we have three variables

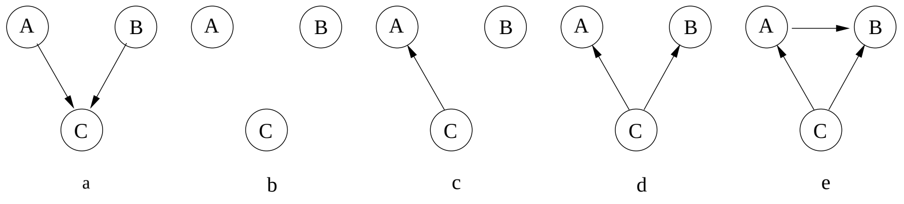

Fig. 1. Greedy search

distributed according to the bayesian network 1a (which is the only instance of its equivalence class). Let us consider that there is enough data, such as the evaluation function we use (which is asymptotically correct) assigns the best score to the network 1a, among all possible network structures.

Suppose the search starts with the totally unconnected network of fig. 1b. Hopefully, the search algorithm will immediately find that adding edges between A and C and between B and C improves the score of the network. Suppose, for instance, that the addition of an edge between A and C produces the greatest improvement of the score. Since the structures $(A \rightarrow C \quad B)$ and $(A \leftarrow C \quad B)$ belong to the same equivalence class (as described in the next section), the addition of the edge $A \rightarrow C$ and the addition of the edge $A \leftarrow C$ have the same impact on score at the beginning of the search. Therefore, there is a 50 % chance[1] that the search algorithm adds the first edge in the wrong direction $(A \leftarrow C)$. If

[1] From our experience with publicly available software, it appears that most programmers seem to neglect this issue and simply apply the first best operator found. Since the order of evaluation of edge additions generally depends on node order and publicly available networks often declare nodes in their topological order, the results of

it does so, the direction of the edge between B and C also becomes indifferent from the score viewpoint, as the structures $(A \leftarrow C \rightarrow B)$ and $(A \leftarrow C \leftarrow B)$ belong to the same equivalence class. Globally, there is a 25 % chance that the network found after the second iteration is $(A \leftarrow C \rightarrow B)$. In this case, since A and B become dependent when C is known (as shown by fig. 1a), an edge will be added between these nodes (*e.g.,* $A \rightarrow B$). This makes the search stop in an incorrect state, because no further single edge reversal or removal can improve the score. In larger networks, this kind of early wrong decisions are very probable and their effects can cumulate and make the final network very different from the ideal one.

3 Heuristic Search in the Space of Equivalence Classes of Bayesian Network Structures

Bayesian networks that represent the same conditional independence relations form an equivalence class. All bayesian networks belonging to the same equivalence class have the same *skeleton* (undirected graph resulting from ignoring the directionality of edges) and the same *v-structures* (triples of nodes A, B, C such that A and B are not adjacent and are connected to C by the edges $A \rightarrow C \leftarrow B$ (as in fig. 1a) [4]. Note the structure of fig. 1e does not represent a v-structure and does not contain any v-structure.

Equivalence classes are generally represented as partially directed graphs (*essential graphs*, *completed pdags* or *patterns*) defined as follows:

– edges that may appear in either direction in networks belonging to the same equivalence class are represented as undirected edges;
– the other edges are represented as directed edges.

These conditions define a unique representation for equivalence classes but do not ensure that the equivalence classes represented this way are legal (*i.e.,* can be instantiated).

In order to overcome the difficulties presented in the previous section, we can realize the search in the space of equivalence classes. Intuitively, this approach consists in allowing the addition of undirected edges when no direction is preferred by the score. Edge orientation is delayed until the interactions between edges make possible the choice of a direction on the basis of the score. Since the obtained partially directed graphs may be interpreted as equivalence classes, this solution consists in a modification of the search space: the search algorithm explores the space of equivalence classes of bayesian networks instead of the space of bayesian networks.

This kind of solution has already been studied in [5]. The conclusion of this work was that the search in the space of equivalence classes generally provides better results than the search in the space of bayesian networks but, unfortunately, it is much more time consuming.

these programs on “gold-standard” benchmarks are over-optimistic. Reversing the order of nodes declarations can lead to serious degradations of their performance.

One of the difficulties Chickering met in his work was the evaluation of equivalence classes. Since available evaluation functions have been designed to score bayesian networks (and not equivalence classes), his solution consists in the generation of an arbitrary instance of the equivalence class to be evaluated and the evaluation of this instance according to the classical formulas. Additionally, Chickering's algorithm relies on this procedure in order to prevent the construction of illegal equivalence classes (without instances).

This procedure is much more time-consuming than the evaluation of a bayesian network because it needs some additional time to generate an instance from the equivalence class and because more than two local scores may have to be evaluated in order to evaluate the generated instance. Furthermore, Chickering's algorithm uses a complex set of transformation operators, that produce a great number of candidate graphs at each iteration of the search.

4 The EQ1 Algorithm

EQ1 is a learning algorithm that explores the space of equivalence classes with a reduced set of operators and realizes the verification of the consistency and the computation of the score without instantiating the equivalence classes constructed during the search.

EQ1 greedily uses the following transformation operators:

– *Operator 1 :* addition of a directed edge $X \rightarrow Y$ between two nodes which are not adjacent;
– *Operator 2 :* addition of a v-structure $X \rightarrow Y \leftarrow Z$ between three nodes in configuration $X \quad Y - Z$, where X and Z are not adjacent (addition of a directed edge together with the orientation of a previously undirected edge);
– *Operator 3 :* addition of an undirected edge $X - Y$ between two nodes which are not adjacent.

Since "repair" operators (edge deletions and reversals), generally used in traditional learning algorithms, are mainly needed for correcting the errors of edge orientation made in the early phases of the search, we decided not to use them in EQ1.

4.1 Constraints on the Transformation Operators

As discussed earlier, not all partially directed graphs represent legal equivalence classes. A partially directed graph G represents a legal equivalence class if and only if it satisfies the following conditions [6]:

1. G is a chain graph (*i.e.*, it contains no directed or partially directed cycle);
2. every chain component of G is chordal (*i.e.*, on every undirected cycle of length ≥ 4 there are two non-consecutive nodes connected by an undirected edge - a *chord*);

3. the configuration $A \rightarrow B - C$ does not occur as an induced subgraph of G;
4. every edge $A \rightarrow B$ must occur in at least one of the four configurations of fig. 2 as an induced subgraph of G.

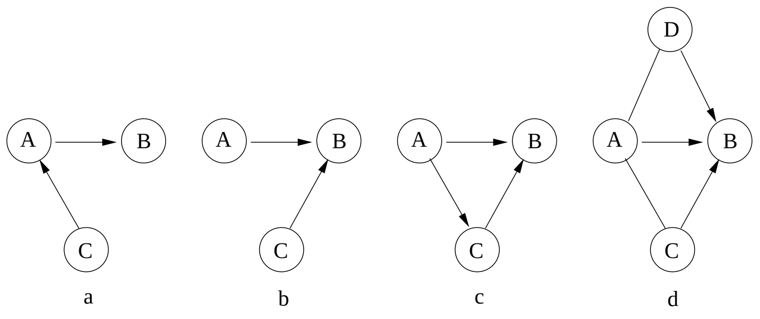

Fig. 2. Possible configurations for directed edges

It can be easily verified that this characterization entails the following constraints on the transformation operators of EQ1 :

- *Operator 1 :* X and Y must have different sets of parents. $X \rightarrow Y$ may not introduce any directed or partially directed cycle.
- *Operator 2 :* $X \rightarrow Y$ may not introduce any directed or partially directed cycle (after the orientation of $Y \leftarrow Z$).
- *Operator 3 :* X and Y must have the same (possibly empty) sets of parents. If $X - Y$ introduces an undirected cycle, it must also introduce an undirected triangle.

Furthermore, the addition of a v-structure may require the orientation of some previously undirected edges, in order to satisfy the third condition of the above theorem. Similarly, the addition of a directed edge (*e.g.*, $C \rightarrow B$ in fig. 2a), or of an undirected edge (*e.g.*, $A - C$ in fig. 2b or $C - D$ in fig. 2d) may request the removal of the orientation of some previously directed edges ($A \rightarrow B$ in fig. 2a, $A \rightarrow B$ and $C \rightarrow B$ in fig. 2b, $A \rightarrow B$, $C \rightarrow B$ and $D \rightarrow B$ in fig. 2d) in order to satisfy the fourth condition of the theorem. The addition or the removal of an edge orientation may have cascading effects.

Since these additions or removals of edge orientations do not have any influence on the evaluation of the transformations (presented in the next subsection), they are implemented as post-processing operations, performed by EQ1 only for the best found transformation, which is really applied on the current graph.

4.2 Scoring the Transformations

Since an equivalence class contains instances with the same score, we can define the score of an equivalence class as being the score of any of its instances. In the following, $score(G)$ denotes the score of the (directed or partially directed) graph G. Remember the score of a bayesian network is decomposable on nodes. $score(X \mid \mathcal{P})$ denotes the score of node X with the set of nodes $\mathcal{P}$ as parents. $Pa_G(X)$ ("parents" of X in G) denotes the set of nodes Y such that $X \leftarrow Y$ belongs to graph G. $Br_G(X,Y)$ ("common brothers" of X and Y in G) denotes the set of nodes Z such that $X - Z$ and $Y - Z$ belong to G.

All formulas used by EQ1 for scoring the transformations can be derived on the basis of the following argument: Let G and G' be the partially oriented graphs corresponding to the current equivalence class and to the transformed equivalence class. We will see in the following paragraphs that, for all transformation operators, we can build some instances I of G and I' of G' such that all nodes have the same parents in I and I', except for a single node, Y. It follows that the difference between the scores of G' and G is equal to the difference between the score of Y in I' and the score of Y in I. All transformations can therefore be evaluated by computing the score of a single node in two different configurations.

First consider the addition of a directed edge $X \rightarrow Y$ (*Operator 1*). We can build I such that all undirected edges adjacent to Y in G, $Y - Z$, are oriented as $Y \rightarrow Z$ in I. Let I' be the directed graph obtained by adding $X \rightarrow Y$ to I. It can be easily verified that I' is an in instance of G'. We have, therefore :

$$\Delta score(G', G) = score(Y \mid Pa_G(Y), X) - score(Y \mid \ Pa_G(Y))$$

If the addition of the edge $X \rightarrow Y$ is done together with the orientation of a previously undirected edge $Y - Z$, which becomes $Y \leftarrow Z$ (*Operator 2*), we build I such that $Y - Z$ is already oriented as $Y \leftarrow Z$ and all the other undirected edges adjacent to Y in G, $Y - W$, are oriented as $Y \rightarrow W$. An instance of G', I', can be obtained from I by adding the edge $X \rightarrow Y$. We have, therefore :

$$\Delta score(G', G) = score(Y \mid Pa_G(Y), Z, X) - score(Y \mid \ Pa_G(Y), Z)$$

Let us now consider the addition of an undirected edge $X - Y$ (*Operator 3*). Since instances have only directed edges, I' must be produced from I by adding a directed edge, for instance $X \rightarrow Y$. In order to avoid directed cycles in I', we have to orient all edges $Y - Z$, where Z is a common brother of X and Y, as $Y \leftarrow Z$. This will not introduce any spurious v-structure in I and I', because X and Y have the same parents in G and all their common brothers are interconnected (*cf.* constraints on *Operator 2*). All other undirected edges adjacent to Y in G, $Y - W$, are oriented as $Y \rightarrow W$. The formula used for scoring this transformation is, therefore:

$$\begin{aligned}\Delta score(G', G) = & \ score(Y \mid Pa_G(Y), Br_G(X,Y), X) \\ & - score(Y \mid Pa_G(Y), Br_G(X,Y))\end{aligned}$$

5 Experimental Results

In order to evaluate the performances of EQ1, we have compared it experimentally to greedy search and tabu search in the space of bayesian networks

(GreedyBN and TabuBN). TabuBN uses a tabu list of 10 states and stops after 10 consecutive iterations without score improvement.

All algorithms use the MDL score (see [2] for more details and justification):

$$score(G) = \sum_{i=1}^{n} (\log(n) + \log \binom{n}{|Pa_i|} + \|Pa_i\|(\|X_i\| - 1)\frac{\log(N)}{2} + NH(X_i|Pa_i))$$

where $X_1, \cdots, X_n$ are the nodes of G, $Pa_i = Pa_G(X_i)$ are their sets of parents, $|Pa_i|$ is the number of parents of X_i, $\|X_i\|$ is the number of values of X_i, $\|Pa_i\|$ is the number of different instantiations of the set of variables Pa_i, N is the number of examples and $H(X_i|Pa_i)$ is the conditional entropy of X_i, given Pa_i.

The comparison has been realized on learning tasks involving five publicly available bayesian networks of various sizes: *Cancer* (5 nodes, 5 edges), *Asia* (8 nodes, 8 edges), *CarStarts* (18 nodes, 17 edges), *Alarm* (37 nodes, 46 edges), *Hailfinder* (56 nodes, 66 edges).

In order to improve the statistical significance of the experimental results, we have compared the algorithms on thirty different data sets for each network (1,000 examples for the small networks *Cancer* and *Asia*, and 10,000 for the others, generated according to the probability distributions modeled by the networks).

The first criterion we have evaluated is the score. Tables 1 and 2 present:

- the network used to generate the examples;
- the means of the score of the compared algorithms[2];
- the number of data sets on which the first algorithm obtained a score inferior, equal or superior to the score of the second one (the MDL score has to be *minimized*);
- the probability p that the means of the score of the two algorithms are equal (paired t-test);
- the algorithm which obtained the better mean of the score, if $p < 0.01$.

Table 1. GreedyBN vs. EQ1

Network	GreedyBN	EQ1	$<$	$=$	$>$	p	Best
Cancer	3266.29	3261.57	0	15	15	1.12E-05	EQ1
Asia	3343.20	3335.82	0	14	16	2.06E-04	EQ1
CarStarts	33563.80	33517.19	0	4	26	2.69E-07	EQ1
Alarm	139719.52	139198.80	5	0	25	2.76E-06	EQ1
Hailfinder	720712.31	720038.42	0	0	30	3.07E-11	EQ1

[2] Since this work deals exclusively with the optimization of evaluation functions proposed elsewhere, only the scores obtained on train sets are reported. Nevertheless, we have also evaluated the scores on independent test sets and observed the same patterns of relative behavior of the algorithms as those reported in tables 1 and 2.

Table 2. TabuBN vs. EQ1

Network	TabuBN	EQ1	$<$	$=$	$>$	p	Best
Cancer	3262.61	3261.57	0	27	3	8.32E-02	
Asia	3336.69	3335.82	1	24	5	1.05E-01	
CarStarts	33553.79	33517.19	0	11	19	1.32E-05	EQ1
Alarm	139558.87	139198.80	6	0	24	2.72E-04	EQ1
Hailfinder	720383.23	720038.42	4	2	24	9.94E-06	EQ1

These results clearly show that EQ1 is statistically more successful than GreedyBN, and overpasses even TabuBN on non-trivial tasks.

The differences between scores do not seem very important but they represent different local optima that may correspond to rather different structures. In order to appreciate the fidelity of the discovered structures, we have compared them with the networks used for generating the data sets ("gold-standards").

Table 3 present these comparisons on the basis of:

- the number of edges of the skeleton of the learned network that do not exist in the skeleton of the "gold-standard" (A+);
- the number of edges of the skeleton of the "gold-standard" that do not exist in the skeleton of the learned network (A-);
- the number of v-structures of the learned network that do not exist in the "gold-standard" (VS+);
- the number of v-structures of the "gold-standard" that do not exist in the skeleton of the learned network (VS-);

In order to make easier the interpretation of this table, the best results for each of these criteria are presented in bold face.

Table 3. Structural differences

	GreedyBN				TabuBN				EQ1			
Network	A+	A-	VS+	VS-	A+	A-	VS+	VS-	A+	A-	VS+	VS-
Cancer	**0.40**	0.50	0.50	**0.00**	**0.40**	0.10	0.10	0.03	**0.40**	**0.00**	**0.00**	0.03
Asia	1.50	0.83	0.80	0.43	1.40	0.47	0.27	0.27	**1.23**	**0.23**	**0.17**	**0.17**
CarStarts	3.40	2.80	10.27	1.67	3.40	1.97	9.10	1.57	**2.90**	**0.10**	**7.33**	**0.00**
Alarm	2.73	14.10	17.60	5.43	2.67	12.53	15.83	5.00	**2.07**	**4.13**	**5.90**	**1.27**
Hailfinder	21.90	25.73	23.70	5.33	21.70	20.87	19.03	5.30	**20.83**	**17.23**	**16.37**	**4.60**

Once again, EQ1 is clearly more successful than the other two algorithms, notably on the *Alarm* network.

The last table present the comparison of the average execution times of the three algorithms. They are all programmed in Java, using the same base classes, the same methods for computing scores and the same caching schemas. The

Table 4. Execution times

Network	Greedy	Tabu	EQ1
Cancer	1.21	1.27	0.96
Asia	2.32	2.81	2.15
CarStarts	53.48	62.29	64.25
Alarm	319.55	496.47	364.80
Hailfinder	695.00	1490.02	811.84

tabu list of TabuBN is implemented as a hash table. The comparison has been realized on a PIII 500Mhz CPU. The results are given in seconds.

The execution times of EQ1 are comparable to those of GreedyBN and globally smaller than those of TabuBN. These results are very contrasting with those reported by Chickering [5] (his equivalence class learning algorithm was 10 to 20 times slower than greedy search in the space of bayesian networks).

6 Conclusion

The main result presented in this paper is that the efficiency of the search in the space of equivalence classes of bayesian networks can be considerably improved if the verification of the consistency and the computation of the score of candidate structures can be made locally, without instantiation.

This paper also confirm the interest of exploring the space of equivalence classes, even with a reduced set of transformation operators, both from the viewpoint of the score and of the structural fidelity of the learned networks.

References

1. P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search.* Springer-Verlag, 1993.
2. N. Friedman and M. Goldszmidt. Learning bayesian networks with local structure. In *Proceedings of the Twelfth Intenational Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann, 1996.
3. D. Heckerman, D. Geiger, and D.M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.
4. T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence.* Elsevier, 1990.
5. D.M. Chickering. Learning equivalence classes of bayesian-network structures. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann, 1996.
6. S.A. Andersson, D. Madigan, and M.D. Perlman. A characterization of markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25:505–541, 1997.