Learning Bayesian Networks Consistent with the Optimal Branching

Alexandra M. Carvalho and Arlindo L. Oliveira
IST, TULisbon/INESC-ID
Lisboa, Portugal
{asmc,aml}@inesc-id.pt

December 21, 2007

Abstract

We introduce a polynomial-time algorithm to learn Bayesian networks whose structure is restricted to nodes with in-degree at most k and to edges consistent with the optimal branching, that we call consistent k-graphs (CkG). The optimal branching is used as an heuristic for a primary causality order between network variables, which is subsequently refined, according to a certain score, into an optimal CkG Bayesian network. This approach augments the search space exponentially, in the number of nodes, relatively to trees, yet keeping a polynomial-time bound. The proposed algorithm can be applied to scores that decompose over the network structure, such as the well known LL, MDL, AIC, BIC, K2, BD, BDe, BDeu and MIT scores. We tested the proposed algorithm in a classification task. We show that the induced classifier always score better than or the same as the Naive Bayes and Tree Augmented Naive Bayes classifiers. Experiments on the UCI repository show that, in many cases, the improved scores translate into increased classification accuracy.

1 Introduction

Bayesian networks [14] allow efficient and accurate representation of the joint probability distribution over a set of random variables. For this reason, they have been widely used in several domains of application where uncertainty plays an important role, like medical diagnosis and modeling DNA binding sites. Learning Bayesian networks consists of finding the network that best fits, for a certain scoring function, the data. This problem is not straightforward. Cooper [3] showed that the inference of a general Bayesian network is a NP-hard problem, and later, Dagum and Luby [5] showed that even finding an approximate solution is NP-hard.

These results led the community to search for the largest subclass of Bayesian networks for which there is an efficient structure learning algorithm. First attempts confined the network to tree structures and used Edmonds [7] and Chow-Liu [2] optimal branching algorithms to learn the network. More general classes of Bayesian networks have eluded efforts to develop efficient learning algorithms. Indeed, Chickering [1] showed that learning the structure of a Bayesian network is NP-hard even for networks constrained to have in-degree at most 2. Later, Dasgupta [6] showed that even learning 2-polytrees is NP-hard. Due to these hardness results exact polynomial-time bounded approaches for learning Bayesian networks have been restricted to tree structures.

Consequently, the standard methodology for addressing the problem of learning Bayesian networks became heuristic search, based on scoring metrics optimization, conducted over some search space. Many algorithms have been proposed along these lines, varying both on the formulation of the search space (network structures, equivalence classes of network structures and orderings over the network variables), and on the algorithm to search the space (greedy hill-climbing, simulated annealing, genetic algorithms, tabu search, etc). Although searching in the space of network structures was commonly considered as the standard choice, more recently it has been shown that searching the space of orderings [16] empirically outperforms the standard baseline of greedy hill-climbing, modified with a tabu list and random restarts, over the space of network structures.

There are several reasons why orderings have recently been attracting so much attention [16, 9]. First, orderings provide a first clue on the causality of the network variables, which can then be refined in subsequent processing. By itself, this observation is of limited use, since determining an appropriate ordering is a difficult problem. Nevertheless, if the search is conducted in the space of orderings, instead of networks structures, there is a severe decrease in the search space, which eases the task. Second, given an ordering on the network variables, finding an optimal bounded in-degree network consistent with it is not NP-hard [16] (unless P=NP). Indeed, if the in-degree of a node is bounded to k, this task can be accomplished in $O(n^k)$ time, where n is the number of variables in the network. Finally, given a network consistent with some ordering on the variables, there is no need to check for network cycles, since it is guaranteed that the network will always be acyclic.

The contribution of this paper assents in taking the topological order of the optimal branching as an heuristic for a causality order between the network variables. Moreover, the optimal network consistent with the topological order will always score better than, or the same as, the optimal branching. By taking these two observations into account, we obtain an exact polynomial-time algorithm for learning subclasses of Bayesian networks which are more general than branchings and intersect, but are not contained in, polytrees. This class consists of directed acyclic graphs of in-degree at most k that are consistent with the optimal branching, henceforward called *consistent k-graphs* (CkG). We show that this class is exponentially larger, in the number of variables, when compared to trees. The proposed algorithm copes with scoring functions that decompose over the network structure. Well known scores with this property are those based on information theory, such as *log likelihood* (LL), *Akaike information criterion* (AIC), *Bayesian information criterion* (BIC), *minimum description length* (MDL); and Bayesian scoring function such as K2, *Bayesian Dirichlet* (BD) and its variants (BDe, BDeu), and *mutual information test* (MIT).

The paper is organized as follows. In Section 2, we briefly revise Bayesian networks and present the Chow-Liu tree learning algorithm and its extension for general decomposable scores. In Section 3, we introduce the main contribution of this paper, the exact optimization algorithm for searching CkG's. In Section 4, we apply our approach to classification and present some experimental results. Finally, in Section 5 we draw some conclusions and discuss future work. The proofs of all statements presented in this paper are given in Appendix.

2 Basic concepts and results

In this section we introduce some notation, while recalling relevant concepts and results concerning Bayesian networks which are directly related with the contribution of this paper.

2.1 Bayesian networks

A *Bayesian network* is a triple $B = (\mathbf{X}, G, \Theta)$. The first component $\mathbf{X} = (X_1, \ldots, X_n)$ is a finite random vector where each random variable X_i ranges over a finite domain D_i. We denote the joint domain $\mathbf{D} = \Pi_{i=1}^n D_i$. The second component $G = (N, E)$ is a directed acyclic graph with nodes $N = \{X_1, \ldots, X_n\}$ and edges E representing direct dependencies between the variables. The third component Θ encodes the parameters $\{\theta_{x_i|\Pi_{x_i}}\}_{\mathbf{x}\in\mathbf{D}}$ of the network, where $\theta_{x_i|\Pi_{x_i}} = P_B(x_i|\Pi_{x_i})$ for each possible value x_i of X_i, and Π_{x_i} of Π_{X_i}, where Π_{X_i} denotes the set of parents of X_i in G. A Bayesian network defines a unique joint probability distribution over $\mathbf{X}$ given by

$$P_B(X_1, \ldots, X_n) = \prod_{i=1}^{n} \theta_{X_i|\Pi_{X_i}}.$$

We denote the set of all Bayesian networks with n variables by $\mathcal{B}_n$.

Informally, a Bayesian network encodes the independence assumptions over the component random variables of $\mathbf{X}$. An edge (i, j) in E represents a direct dependency of X_j to X_i. Moreover X_i is independent of its non descendants given its parents Π_{X_i} in G.

The problem of learning a Bayesian network given data T consists on finding the Bayesian network that best fits the data T. In order to quantify the fitting of a Bayesian network a *scoring function* $\phi : \mathcal{B}_n \times \mathbf{D}^m \to \mathbb{R}$ is considered. In this context, the problem of learning a Bayesian network can be recasted to the following optimization problem. Given a dataset $T = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ and a scoring function ϕ, the *problem of learning a Bayesian network* is to find a Bayesian network $B \in \mathcal{B}_n$ that maximizes the value ϕ for T.

Several scoring functions have been proposed in the literature [4, 10, 12, 15]. The discussion of the advantages and disadvantages of each of these functions is outside the scope of this paper.

2.2 Chow-Liu tree learning algorithm and extensions

A *tree Bayesian network* is a Bayesian network where the underlying directed acyclic graph is a directed tree. Finding the tree Bayesian network that maximizes the *LL* score given data T can be done in polynomial time by the Chow-Liu tree learning algorithm [2]. The main idea of the algorithm is to consider a complete weighted undirected graph, where each undirected edge between X_i and X_j is weighted with the conditional mutual information between X_i and X_j. Given this, the problem reduces to determining a maximal weighted (undirected) spanning tree. After computing such spanning tree, a direction has to be assigned to each edge of the tree. This is done by choosing an arbitrary node as the tree root and then setting the direction of all edges to be outward from it. The resulting directed tree is called *Chow-Liu tree* or *optimal branching* and it is computed in $O(n^2 \log(n) + n^2 m)$ time.

The Chow-Liu tree learning algorithm was originally proposed for maximizing the LL score but it can be easily adapted to deal with any scoring function that is *decomposable* and *score equivalent*. We recall that a scoring function ϕ is *decomposable* if it can be written as

$$\phi(B, T) = \sum_{i=1}^{n} \phi_i(\Pi_{X_i}, T). \tag{1}$$

Moreover, a scoring function is said to be *score equivalent* if it assigns the same value to all directed acyclic graphs that are represented by the same essential graph. All interesting scoring functions in the literature are decomposable, since it is unfeasible to learn undecomposable scores. LL, AIC, BIC and MDL are decomposable and score equivalent, whereas K2, BD, BDe, BDeu and MIT are decomposable but not score equivalent.

According to Heckerman et al [10], finding the optimal branching for decomposable and score equivalent scoring functions reduces to weighting each undirected edge between X_i and X_j by $\phi_j(\{X_i\}, T) - \phi_j(\emptyset, T)$, which is equal to $\phi_i(\{X_j\}, T) - \phi_i(\emptyset, T)$ by score equivalence of ϕ, and to find a maximal weighted (undirected) spanning tree. In this case the optimal branching is computed in $O(n^2 \log n + n^2 \gamma(T))$ time, where $\gamma(T)$ is an upper bound for computing $\phi_j(\{X_i\}, T)$ and $\phi_j(\emptyset, T)$ (in most cases $\gamma(T) = |T| = m$).

Again, according to Heckerman et al [10], learning the optimal branching for scoring functions that are only decomposable, but not score equivalent, can also be done in polynomial time. In this case, however, an edge between X_i and X_j may score differently depending on its direction, and so a directed spanning tree must be found (instead of an undirected one). The idea is to weight each directed edge from X_i and X_j with $\phi_j(\{X_i\}, T) - \phi_j(\emptyset, T)$ and then, for each node X_r, find the optimal directed spanning tree rooted at X_r with Edmonds' algorithm [7] (which takes $\mathrm{O}(n^2 \log(n)$ time). As Edmnond's algorithm assumes a fixed root, by ranging over all potential roots we can find an optimal branching in $\mathrm{O}(n^3 \log(n) + n^2 \gamma(T))$ time.

3 Consistent k-graph Bayesian networks

The contribution of this paper assents in the following observations. First, the topological order of the optimal branching is a simple and effective heuristic for a causality order between the network variables. Second, the score of the optimal network (of bounded in-degree) consistent with the topological order is always greater than or equal to the score of the optimal branching. Third, there is an exact polynomial-time algorithm for learning such optimal consistent network. Finally, the class of the networks consistent with the optimal branching is exponentially larger, in the number of variables, when compared to trees.

We start by introducing some auxiliary concepts. A *k-graph* is a graph where each node has indegree at most k. Trees and forests are 1-graphs.

Definition 3.1 (Consistent k-graph) Given a directed tree R over a set of nodes N, a graph $G = (N, E)$ is said to be a *consistent k-graph* (CkG) w.r.t R if it is a k-graph and for any edge in E from X_i to X_j the node X_i is in the path from the root of R to X_j. Henceforward, we denote by $\mathcal{C}_R^k$ the set of all CkG's w.r.t. R.

The idea of the CkG learning algorithm, presented in Algorithm 1, is to start with the optimal branching R and modify it in such a way that: (i) important dependencies are added to R (which were lost due to the tree structure restriction); (ii) irrelevant dependencies are removed from R (which were also present due to the tree structure restriction). To this effect, after computing the optimal branching R, the algorithm ranges over each node X_i, generates the set α_i of all nodes in the path from the root of R to X_i and takes as parents of X_i the set $S \subseteq \alpha_i$ such that $\phi_i(S, T)$ is maximal over all subsets of α_i with at most k nodes.

Algorithm 1 Learning CkG networks

```
1. Run a (deterministic) algorithm $\mathcal{A}_\phi$ that outputs an optimal branching $R$.
2. For each node $X_i$ in $R$ do:
   (a) Compute the set $\alpha_i$ of ancestor of $i$, that is, the set of nodes connecting the root of $R$
       and $X_i$.
   (b) For each subset $S$ of $\alpha_i$ with at most $k$ nodes
       i. Compute $\phi_i(S, T)$.
       ii. If $\phi_i(S, T)$ is the maximal score for $X_i$, set $\Pi_i$ to $S$.
3. Output the directed graph such that the parents of a node $X_i$ are $\Pi_i$.
```

Theorem 3.2 Algorithm 1 constructs a CkG Bayesian network that maximizes the ϕ-score given data T, which is always greater than, or equal to, the ϕ-score of the optimal branching computed by $\mathcal{A}_\phi$. Moreover, the optimal CkG Bayesian network is obtained in $\mathrm{O}(n^{k+1}\gamma(k, T))$ time where $\gamma(k, T)$ is an upper bound for computing $\phi_i(S, T)$.

Theorem 3.2 shows the soundness and polynomial-time bound of the CkG learning algorithm. At this point it remains to show that, despite considering an optimal branching to confine the search space, the number of graphs searched increases exponentially, in the number of nodes, when compared to trees.

Proposition 3.3 Given a tree R with n nodes there are at least $(2^{(n-1)} - 1)$ non-trees in $\mathcal{C}_R^k$.

Figure 1 depicts relationships in terms of expressiveness between the different model networks discussed in this paper. Observe that the search space of the CkG learning algorithm consists of

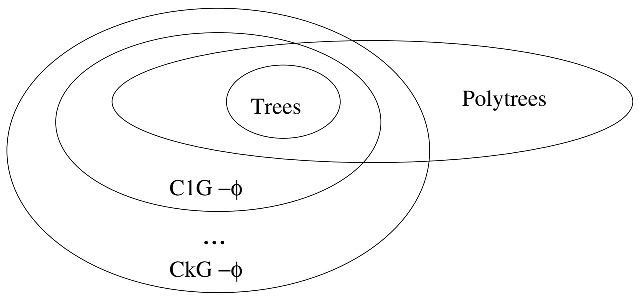

Figure 1: Expressiveness of the network models discussed in this work. Polytree includes trees and intersect the search space of the CkG algorithm (each scoring function ϕ induces a different CkG class).

the union of all trees together with $\mathcal{C}^k_R$ where R is the optimal branching for the ϕ-score given by a (deterministic) algorithm $\mathcal{A}_\phi$. Therefore, the search space of the algorithm depends on the scoring function ϕ, and in particular on $\mathcal{A}_\phi$.

4 Experimental methodology and results

We adapted the CkG learning algorithm to classification following the work of Friedman et al [8], where the Chow-Liu tree learning algorithm was adapted to develop the Tree Augmented Naive Bayes (TAN) classifier. We compared the C2G classifier, with TAN and the well known Naive Bayes (NB) classifier.

As suggested by Friedman et al [8] we improved the performance of both TAN and CkG classifiers by introducing an additional smoothing operation. This is particularly important in small datasets where the estimation of the conditional probabilities, given the parent attributes plus the class variable, is unreliable. NB classifiers are almost not affected by this problem since the data is partitioned according to the class variable and, usually, the class variables are adequately represented in the training set. The parameters of TAN and CkG networks were smoothed using *Dirichlet priors* [10]. This amounts to adding 5 pseudo instances with conditional probabilities, given the parent attributes plus the class variable, distributed according to the frequency of the corresponding attribute in the training set.

We ran our experiments on several data sets from the UCI repository [13]. The accuracy of each classifier is based on the percentage of successful predictions on the test sets of each data set. Accuracy was measured via the *holdout method* for larger training sets, and via *5-fold cross-validation* for the smaller ones [11]. Results are presented in Table 1, where the accuracy is annotated by a 95% confidence interval.

To avoid overfitting, we used the MDL score in smaller data sets. On the other hand, for larger ones, which in principle are closer to the asymptotic limit, we used the LL score. In smaller data sets we observed that almost all the learned C2G networks were forests, which were translated in a small gain of accuracy compared with TAN networks. On larger data sets, namely, chess, letter and segment, a significant gain was achieved and obviously the resulting C2G networks were indeed proper C2G networks. These C2G networks were expected, since the LL score always prefers more complex structures.

5 Conclusions

The main contribution of this paper is to consider the topological order of the optimal branching as a simple and effective heuristic for a primary causality order between the network variables. This order is then refined in subsequent structure learning such that: (i) important dependencies are added; (ii) irrelevant dependencies are removed.

Data set	n	$\|D_C\|$	Train	Test	NB	TAN	C2G
letter	16	26	15000	5000	74.80±1.20	84.93±0.99	88.01±0.90
satimage	36	6	4435	2000	82.99±1.64	88.53±1.39	88.68±1.39
chess	36	2	2130	1066	88.13±1.93	92.24±1.60	95.70±1.20
segment	19	7	1540	770	71.76±1.21	75.55±1.14	79.33±1.07
vehicle	18	4	846	CV-5	62.96±1.66	71.24±1.56	71.24±1.56
diabetes	8	2	768	CV-5	76.99±1.52	78.17±1.49	78.69±1.48
soybean-large	35	19	562	CV-5	91.07±1.21	90.54±1.24	91.25±1.19
vote	16	2	435	CV-5	91.30±1.86	93.04±1.68	93.04±1.68
waveform	21	3	300	4700	81.23±1.12	80.63±1.13	81.49±1.11
heart	13	2	270	CV-5	83.71±2.25	83.71±2.25	84.07±2.23
glass	9	7	214	CV-5	94.29±1.61	94.29±1.61	95.24±1.47
iris	4	3	150	CV-5	93.33±2.04	93.33±2.04	94.00±1.94
lymphography	18	4	148	CV-5	78.62±3.40	84.83±2.98	82.07±3.19
hepatitis	19	2	80	CV-5	93.75±2.71	90.00±3.35	91.25±3.16

Table 1: Description of the data sets used in the experiments jointly with the experimental results of the approaches discussed in this paper. Larger training sets were tested with the LL score, whereas smaller ones were tested with the MDL score. The results are presented by decreasing size of the training set.

As a consequence, we introduce a new class of networks, the *consistent k-graph* Bayesian networks (CkG), which are more general than trees and intersect, but are not contained in, polytrees (c.f. Figure 1). Moreover, we show that an optimal CkG can be found in polynomial time, while augmenting the search space exponentially, in the number of nodes, relatively to trees. The CkG learning algorithm can be applied to any decomposable score. We show that the score of the optimal CkG is always greater than or equal to the score of the optimal branching.

The algorithms presented in this paper were implemented and applied to classification. Preliminary experiments show that, in many cases, the improved scores translate into increased classification accuracy. A detailed account of these results is being prepared.

Future work can proceed in different directions. First, it would be interesting to consider a total order, instead of a partial one (as the one induced by the topological order of the optimal branching) since it will increase, in general, the search space significantly. The breath-first search order of the optimal branching seems to be a good candidate. Second, it would also be interesting to combine and compare more exhaustively our approach with other state-of-the-art Bayesian network learning methods, namely, with ordering based approaches.

Acknowledgments

The first author thanks EU FEDER through FCT grant SFRH/BD/18660/2004. This work was partially supported by EU FEDER via FCT project POSC/EIA/ 57398/2004.

References

[1] D. M. Chickering. *Learning Bayesian networks is NP-Complete*, pages 121–130. Learning from data: AI and statistics V. Springer, 1996.

[2] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Info. Theory*, 14(3):462–467, 1968.

[3] G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artif. Intell.*, 42(2-3):393–405, 1990.

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

[5] P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artif. Intell.*, 60(1):141–153, 1993.

[6] S. Dasgupta. Learning polytrees. In *UAI'99*, pages 134–141, 1999.

[7] J. Edmonds. Optimum branchings. *J. Research of the National Bureau of Standards*, 71B:233–240, 1967.

[8] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.

[9] N. Friedman and D. Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1-2):95–125, 2003.

[10] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[11] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI'95*, pages 1137–1145, 1995.

[12] W. Lam and F. Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Comp. Intell.*, 10:269–294, 1994.

[13] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases, 1998.

[14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[15] J. Suzuki. A construction of Bayesian networks from databases based on an MDL principle. In *UAI'93*, pages 266–273, 1993.

[16] M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning bayesian networks. In *UAI'05*, pages 584–591, 2005.

Appendix

Proof of Theorem 3.2: *We show that Algorithm 3.2 is sound.* Since all potential parents for each node are checked, the algorithm returns the k-graph G consistent with he optimal branching R with the highest score. Moreover, this graph is acyclic since the parents of a node X_i must be in α_i, that is, must belong to the path in R from its root to X_i (excluding X_i). Moreover, it is easy to see that for any path $X_{i_1}, X_{i_2}, \dots X_{i_k}$ in G we have that $X_{i_j} \in \alpha_{i_k}$ for $1 \leq j < k$. If there existed a cycle $X_{i_1}, X_{i_2}, \dots X_{i_1}$ it would imply that $X_{i_1} \in \alpha_{i_1}$ which is absurd.

Next, we show that the ϕ-score of a CkG w.r.t. the optimal branching R, obtained by Algorithm 1, is always greater than, or equal to, the ϕ-score of R. Start by noticing that the soundness of Algorithm 1 assures that the resulting CkG w.r.t R is the maximal among all CkG's in $\mathcal{C}_R^k$. Moreover, observe that the optimal branching R is consistent with itself, that is, $R \in \mathcal{C}_R^k$ for all $k \geq 1$. Hence, the soundness of Algorithm 1 guarantees that $\phi(G,T) \geq \phi(R,T)$.

Finally, we analyze the complexity of Algorithm 1. Step 2a) takes $\mathrm{O}(n)$ time, while step 2b) takes $\mathrm{O}(n^k\gamma(S,T))$ because it ranges over all subsets S with at most k elements (which takes $\mathrm{O}(n^k)$ time) and for each of this sets it computes $\phi_i(S,T)$ (which takes $\mathrm{O}(\gamma(S,T))$. Thus, the overall complexity of the algorithm is $\mathrm{O}(n^{k+1}\gamma(S,T))$.

Proof of Proposition 3.3: *We analyze the lower bound for the number of non-tree graphs in* $\mathcal{C}_R^k$. Since a consistent k-graph is obtained from a tree R, every non-root node X_i has at least a node X_j such that $X_j \in \alpha_i$ and $i \neq j$. This means that both $\emptyset$ and $\{X_j\}$ are potential sets of parents of X_i. Hence, there are at least $2^{(n-1)}$ consistent k-graphs (in this case, just forests). Moreover, only one of these consistent k-graphs is a tree. So there are at least $2^{(n-1)-1}$ consistent k-graphs that are not trees.