Heuristics for Initializing Order-Based Bayesian Network Structure Learning

Walter Perez Urcia
and
Denis Deratani Mauá

Universidade de São Paulo, Brasil
wperez@ime.usp.br,denis.maua@usp.br

Abstract. A popular and effective method for learning Bayesian network structures is to perform a greedy search on the space of variable orderings followed by an exhaustive search over the restricted space of compatible parent sets. Usually, the greedy search is initialized with a randomly sampled order. In this article we develop heuristics for producing informed initial solutions to order-based search motivated by the Feedback Arc Set Problem.

Categories and Subject Descriptors: I.2.6 [**Artificial Intelligence**]: Learning

Keywords: Bayesian networks, machine learning, local search

1. INTRODUCTION

Bayesian Networks are models that represent efficiently probability distributions over the atributes of a data set. They are defined by two components:

—A directed acyclic graph (DAG), where the nodes are the atributes of the data set and the edges encode the (in)dependence relationships among the atributes.
—The conditional probability distributions of the atributes given their parents. The latter are defined by the network's structure.

Formally, a Bayesian network is defined by:

$$G = (V, E),$$

where $V = \{X_1, X_2, \ldots, X_n\}$ is the set of attributes and

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid Pa_G(X_i)),$$

where $Pa_G(X_i)$ is the set of atributes that are parents of X_i in G. This definition shows that having less parents for an atribute decrease the number of parameters required in the specification, but learning Bayesian networks from data is a NP-hard problem [David M. Chickering 2004] even when the in-degree (maximum number of parents) is bounded, for this reason, the common approach to solve this problem is to use local search methods, commonly using a scoring function, like the bayesian information criterion (BIC) [Cover and Thomas 1991] or the minimum description length (MDL) [Wai Lam 1994] or Bayesian Dirichlet score (BD) [D. Heckerman and Chickering 1995].

In this article we propose new heuristics for generating good initial solutions to be fed into local search methods. We focus on order-based methods, but our technique can be exploited by any Bayesian net search procedure. We propose a new methods for generating informed initial solutions motivated by solutions of the Feedback Arc Set Problem (FASP). Our experiments show that using these new methods it is possible to learn better networks from data sets.

The article is structured as follows: We begin in Section 2 explaining the Greedy Search algorithm. In Section 3, we describe the new algorithm for gener3ating initial solutions. Section 4 shows the experiments using both approaches and comparing them (in time and scoring) with multiple data sets. Finally, in Section 5 we give some conclusions about the new methods.

2. LEARNING BAYESIAN NETWORKS

2.1 Definition of the problem

The problem of learning the structure of a Bayesian network is stated as: Given a training data set D, select the network (DAG) G that maximizes the scoring function $sc(G, D)$. A general definition of a scoring function is as follows:

$$sc(G, D) = F(G) + \varphi(N) \times P(G),$$

where N is the size of the data set D, $F(G)$ is a data fitness function, $\varphi(N)$ is a function of data size and $P(G)$ measures the model complexity of G.
For instance, using the Bayesian information criterion (BIC) as scoring function we have:

$$sc(G, D) = BIC(G, D) = LL(G) + \frac{\log N}{2} size(G),$$

where

$$LL(G) = \sum_{i=1}^{n} \sum_{k} \sum_{j} N_{ijk} \log \frac{N_{ijk}}{N_{ij}},$$

$$size(G) = \sum_{i=1}^{n} (|\Omega_i| - 1) \prod_{X_j \in Pa(X_i)} |\Omega_j|,$$

N is the number of instances in the data set, n the number of atributes, N_{ijk} the number of instances where attributes has the values i, j and k (the same way for N_{ij}). Finally, Ω_i is the size of the set of possible values for the attribute X_i. Most scoring functions, BIC included, has the property to be decomposable, so we can reduce the problem to the following formula:

$$G = \max_{G} \sum_{i=1}^{n} sc(X_i, Pa(X_i)),$$

where $Pa(X_i)$ is the set of parents of attribute X_i in G. In other words, to get a better network, we need to pick a configuration that has the maximum sum of scores among their attributes. But the main problem with searching $Pa(X_i)$ is that we have 2^{n-1} possible set of parents for each attribute. It means, an exponential number of possible sets based on the number of attributes in the data set. In order to avoid this problem is added a new constraint d, where d is the maximum number of parents for each attribute, so we have only 2^d possible sets. This means $|Pa(X_i)| \leq d$ for all attributes.

2.2 Greedy Search Algorithm

Finally, a Order-based Greedy Search is a popular and effective solution to the problem of learning the structure of a Bayesian network. Algorithm 1 shows its pseudocode, where the function *swap* (in line 6) swaps the values $L[i]$ and $L[i + 1]$ in the order L.

Algorithm 1: Order-based Greedy Search

```
1   L = GetOrder( X_1 , ... , X_n , conf )
2   best_score = score( L )
3   For a number of iterations K or until it converges do
4           current_sol = L
5           For each i = 1 to n-1 do
6                   L_i = swap( L , i , i+1 )
7                   if score( L_i ) > score( current_sol )
8                           current_sol = L_i
9           if score( current_sol ) > score( L ) :
10                  L = current_sol
11  Return L
```

The main idea of this algorithm is to generate an initial order of the attributes based on the *conf* parameter (for instance, *conf* can say that the order has to be random generated). After that, for a number of iterations K or until the solution converges (the best solution's score does not improve) perform swaps between adjacent attributes and calculate their score in order to find a better solution. Finally, return the best solution when the stop condition holds.
So we need to calculate the score for an order of the attributes and we know from subsection 2.1 that, in general, to obtain the score for a network is an NP-hard problem because of the exponential number of possible sets of parents. But given an order the problem can be reduced as follows:

$$G = \max_{G} \sum_{i=1}^{n} sc(X_i, Pa(X_i)) = \sum_{i=1}^{n} \max_{P \subseteq \{X_j < X_i\} : |P| \leq d} sc(X_i, P),$$

where $X_j < X_i$ means that X_j appears before X_i in the order of the attributes. This means that we have to maximize the score for every attribute independently the other ones.
In section 4, the order-based version of the algorithm will be used for learning structure of Bayesian networks using multiple data sets.

3. GENERATING INFORMED INITIAL SOLUTIONS

3.1 Feedback Arc Set Problem

3.2 Informed solutions

An important step in algorithm 1 is in the first line, where is created a initial solution for the data set that is based on a permutation of the attributes.In this section, we propose another way to generate an initial solution. Basically, the first line will be changed to another method as is showed in Algorithm 2.

Algorithm 2: Modification of Greedy Search

```
1   L = Informed_solution( X_1 , ... , X_n )
2   For a number of iterations K do
3           current_sol = find_order( L )
4           if score( current_sol ) > score( L ) :
5                   L = current_sol
6   Return L
```

The method for generating an informed solution is as follows:

(1) Find the set of best parents (using the constraint d in previous section) for each attribute
(2) Build a graph using the relations between attributes and their best parents as edges

(3) For each node X in the graph do
   (a) Do a depth-first search (DFS) on the graph using X as root and mark the edges that made cycles (the ones that go to a visited node)
   (b) Change the edge's directions that made cycles to obtain a DAG
   (c) Calculate the network's score
(4) Choose the best network and return it

In step 1, we find the best parents for each attribute X_i performing an exhaustive search bounded by the maximum number of parents parameter d, like follows:

(1) Generate all possible parent sets for X_i with at most size d
(2) For each parent set P calculate $sc(X_i, P)$
(3) Return the set P that maximizes $sc(X_i, P)$

With all the sets calculated, we use that relations as edges to build a graph. Notice that this graph can have cycles and a Bayesian network does not have cycles.
The problem to transform an unweighted directed graph with cycles to a DAG is called Feedback Arc Set Problem (or FASP). This problem is NP-Hard, but there are approximations for solving it. A popular one is similar to step 3, but it only choose a random node and do not compare all possible ways. As we need to obtain the best possible initial solution, a loop over the attributes is done and step 3.c is added for comparison between them. Finally, we return the best network.

This modification of Local Search will be used in next section to learn Bayesian networks with multiple data sets.

4. EXPERIMENTS AND RESULTS

4.1 Configuration

Table I shows the characteristics of each data set used in the experiments.

Name	n (#attributes)	N (#instances)
Census	13	45000
Dataset 2	n_2	N_2
Dataset 3	n_3	N_3
Dataset 4	n_4	N_4

Table I: Data sets

Also, the values for each general parameter are given below.

—Training/test percentage instances: 65% / 35%
—Maximum number of parents (d): 4
—Number of iterations in Local Search (K): 100

4.2 Results

After running both algorithms for each data sets, we compare the cost (Data Log-Likelihood) and the CPU Time. Tables II and III show them respectively.

Name	Random Sol.	Informed Sol.
Census	-165350	-14683
Dataset 2	n_2	N_2
Dataset 3	n_3	N_3
Dataset 4	n_4	N_4

Table II: Data Log-Likelihood using each approach

Name	Random Sol.	Informed Sol.
Census	10.2 / 72.14	93.51 / 48.89
Dataset 2	n_2	N_2
Dataset 3	n_3	N_3
Dataset 4	n_4	N_4

Table III: CPU Time (in seconds) for initializing/searching using each approach

5. CONCLUSIONS

We conclude that:

—Generating an informed initial solution gives best networks that taking a random one
—To generate the informed initial solution takes more CPU time that the one in the first approach
—A higher number of attributes implies more CPU time for the second approach with a significant difference compared to the first approach
—With an informed solution, Local Search algorithm needs less CPU time for finding a better network

REFERENCES

Cover, T. M. and Thomas, J. A. *Elements of Information Theory*. Wiley-Interscience, 1991.

D. Heckerman, D. G. and Chickering, D. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. 20 (MSR-TR-94-09): 197–243, 1995.

David M. Chickering, David Heckerman, C. M. Large-Sample Learning of Bayesian Networks is NP-Hard. 5 (1): 1287–1330, 2004.

Wai Lam, F. B. Learning Bayesian Belief Networks. An approach based on the MDL principle. 10 (4): 31, 1994.