Scientia Iranica D (2013) 20 (3), 682–694

Sharif University of Technology

Scientia Iranica

Transactions D: Computer Science & Engineering and Electrical Engineering

www.sciencedirect.com

Effective pruning strategies for branch and bound Bayesian networks structure learning from data

K. Etminani [a,*], **M. Naghibzadeh** [a], **A.R. Razavi** [b]

[a] *Department of Computer Engineering, Ferdowsi University of Mashhad, Iran*

[b] *Department of Medical Informatics, Mashhad University of Medical Sciences, Iran*

Received 14 March 2012; revised 1 December 2012; accepted 14 January 2013

KEYWORDS

Bayesian networks;
Structure learning;
Discrete variables;
Branch and bound.

Abstract In this paper, designing a Bayesian network structure to maximize a score function based on learning from data strategy is studied. The scoring function is considered to be a decomposable one such as BDeu, BIC, BD, BDe or AIC. Optimal design of such a network is known to be an NP-hard problem and the solution becomes rapidly infeasible as the number of variables (i.e., nodes in the network) increases. Several methods such as hill-climbing, dynamic programming, and branch and bound techniques are proposed to tackle this problem. However, these techniques either produce sub-optimal solutions or the time required to produce an optimal solution is unacceptable. The challenge of the latter solutions is to reduce the computation time necessary for large-size problems.

In this study, a new branch and bound method called PBB (pruned brand and bound) is proposed which is expected to find the globally optimal network structure with respect to a given score function. It is an any-time method, i.e., if it is externally stopped, it gives the best solution found until that time. Several pruning strategies are proposed to reduce the number of nodes created in the branch and bound tree. Practical experiments show the effectiveness of these pruning strategies. The performance of PBB, on several common datasets, is compared with the latest state-of-the-art methods. The results show its superiority in many aspects, especially, in the required running time, and the number of created nodes of the branch and bound tree.

1. Introduction

A Bayesian network or Belief Network (BN) is a probabilistic graphical model that represents conditional independencies between a set of random variables. It is first introduced by Pearl in [1]. It was a directed acyclic graph (DAG) where nodes stand for random variables and edges stand for conditional independencies. The random variables can be discrete, continuous, or both.

Bayesian networks are used for modelling knowledge in computational biology and bio-informatics [2], medicine [3], information retrieval [4], and decision support systems [5], etc. They are used for predicting desired outputs in uncertain environments. One of the advantages of BNs compared to other methods is their support of uncertainty.

Bayesian network structure learning from data has attracted a great deal of research in recent years. Finding the best structure for a Bayesian network is known to be NP-Hard [6,7]. The number of possible structures is $O(n!2^{\binom{n}{2}})$ [8], where n is the number of variables (nodes in the Bayesian Network). Consequently, much of the research has focused on methods that find suboptimal solutions. Generally, there are several approaches to learn a structure. Some methods are based on scoring functions that depend on the data (called scoring-based methods) and some approaches are based on statistical similarities among variables (called constraint-based methods). However, there is some research that makes use of both of these such as [9]. Heckerman et al. [10] compare these two general approaches for learning BNs, and show that the scoring-based methods often have certain advantages over the constraint-based methods.

* Corresponding author. Tel.: +98 511 8763306; fax: +98 511 8763306.
E-mail addresses: etminani@wali.um.ac.ir (K. Etminani), naghibzadeh@um.ac.ir (M. Naghibzadeh), razaviar@mums.ac.ir (A.R. Razavi).

Peer review under responsibility of Sharif University of Technology.

Production and hosting by Elsevier

1026-3098 © 2013 Sharif University of Technology. Production and hosting by Elsevier B.V. All rights reserved.
doi:10.1016/j.scient.2013.02.027

We focus on those methods that are based on scoring functions and try to find the optimal solution with respect to this function.

The rest of the paper is organized as follows: In Section 2, the related work of Bayesian network structure learning is discussed. In Section 3, the Bayesian networks and scoring functions (i.e. problem description) are introduced. The proposed method i.e. PBB, the branching function and the pruning rules, are explained in Section 4. The performance of PBB through experiments is shown in Section 5 and the paper ends with new ideas for future work in Section 6.

2. Related work

Buntine in [11] proposes a hill-climbing method that performs a stochastic local search. It starts with an initial structure and examines possible local modifications (add, delete, reverse or update in the network edges) to a single edge at each phase, evaluates this change and applies it if it maximizes a scoring function. It stops whenever no change in the whole network structure is reported. Although the method is simple and applicable to small networks, since it is an exhaustive method it cannot be applied to networks with a large number of variables due to a large number of possible edge modifications. Besides, it may get trapped in a local maximum. This procedure can be augmented with other methods such as random restarts and the Tabu search [12] to escape local maxima.

Pearl [13] and Peter and Clark [14] have developed algorithms to systematically determine the skeleton of the underlying graph and, then, determine the direction of all arrows by the conditional independencies observed.

Elidan et al. in [15] present a method that is inspired from the notion of residues in regression [16]. They build an "ideal parent profile" of a new hypothetical parent, for each variable. The hypothetical parent is the one that best predicts the variable. Some similarity measures are proposed to compare the ideal parent with the real parent (i.e. the parent that a variable can have from all its 2^{n-1} parent sets). They make use of the ideal parent concept in the hill climbing approach. Instead of blindly choosing the next alterations (add, delete, reverse, or update in the network edges) that maximize the score function, in each phase of hill climbing, for edge addition and replacement, they make use of the ideal parent concept for each variable. They examine all real parents and find k to be the most similar candidate. Edge deletion and reversal are performed separately regardless of the ideal parent concept. Now, a queue of potential operations has been created: edge addition and replacement by applying the ideal parent concept, and edge deletion and reversal without applying the ideal parent concept. Each of these operations is applied to the network if it does not create a directed cycle and leads to a better score. This procedure continues until no change in the network structure is reported. In a network with n nodes and e edges, edge addition is in the order of $O(n^2)$, edge replacement is in $O(n.e)$, edge deletion and reversal are in $O(e)$. The ideal parent concept is applied only to edge addition and replacement to reduce their search space. However, this is a hill climbing approach and at each step, all this work should be performed and no limit for the end of the procedure and no proof for the optimality of the result are presented.

Most exact methods that guarantee to find the optimal structure with respect to a scoring function, are based on dynamic programming [17–19] and branch and bound [20–22] techniques.

Koivisto and Sood in [19] propose the first dynamic programming method for optimal Bayesian network structure learning. They compute the Bayesian posterior probability of structural features by Markov chain Monte Carlo (MCMC) search over orderings. They pre-compute the intermediate terms through the Fast Mobius Transform. Then, compute the equations suggested in [23] as a recursive function within the general framework of dynamic programming. This method runs in time $O(n2^n)$ and has a worst case of $O(n2^n)$ memory requirements.

Every DAG has at least one leaf (sink node). Singh and Moore in [18] and Silander and Myllymaki in [17] use this fact to construct the optimal structure within a dynamic programming technique. The method presented by Silander and Myllymaki has five main steps. In the first step, the local scores for all different parent configurations of each variable are calculated (i.e. all $n2^n$ cases). Using the local scores, in the second step, best parent configurations for all variables from a candidate set of parents are found (i.e. totally $n2^n$ best parents). In the third step, the best sinks for all 2^n variable sets are found. The best ordering of the variables is then found from the whole variables set using the results of the previous step, in the fourth step. And at last in the fifth step, the best network structure is found using the results of the second and fourth steps. Singh and Moore present a very similar method. They try to find the best leaf for each subset of variables and at last find the best leaf for the whole variable set, too. Both of the approaches need to calculate all local scores for each node, which requires both time and memory complexity in the order of $O(n2^n)$. To find the best leaf, in both approaches, the time complexity is $O(n2^n)$ and the space complexity is $O(n2^n)$ for storing the scores of subsets.

Suzuki in [22] presents a branch and bound method for finding the best structure, but this does not guarantee finding the global optimal solution. The time and/or space complexities of the above mentioned methods forbid the application of those approaches to networks with a large number of variables. de Campos et al. in [20,21] propose a branch and bound method that reduces the search space of possible structures, while guaranteeing to obtain the optimal solution. This method begins with the network structure in which each node takes its best parent configuration (the parent set that causes the best local score for that node). The structure is not necessarily a DAG, therefore, the method tries to remove the cycles from the structure to obtain a DAG. It finds a directed cycle and breaks it into sub-cases by forcing some edges to be absent/present. It continues until there is no case to be verified. Each time a DAG is found, it compares it with the best DAG (the DAG with the best score) found so far and updates the current best DAG, if necessary. The method runs at most $\prod_{i=1}^{n} C(i)$ steps, where C(i) is the size of the cache for variable V_i, which is the needed space to store the required local scores for variable V_i. Memory requirement for storing this cache is $\sum_{i=1}^{n} C(i)$ (in the worst case it is $O(n2^n)$).

On the other hand, there are several approaches which are focused on learning special cases of Bayesian networks such as BN classifiers such as [24–27]. There is other research which focus on learning naive Bayes such as [28–30].

We present a new branch and bound method, called PBB (pruned branch and bound), which guarantees to find the global optimal Bayesian network structure in less time with less and memory requirement compared to the best exact previous methods. It uses a decomposable scoring function and drastically reduces the search space for possible structures. The branching method and also bounding rules of the proposed