IEICE TRANS. INF. & SYST., VOL.E90–D, NO.7 JULY 2007

PAPER

Enabling Large-Scale Bayesian Network Learning by Preserving Intercluster Directionality

Sungwon JUNG[†a)], ***Nonmember***, **Kwang Hyung LEE**[††b)], ***Member***, ***and*** **Doheon LEE**[†††c)], ***Nonmember***

SUMMARY We propose a recursive clustering and order restriction (R-CORE) method for learning large-scale Bayesian networks. The proposed method considers a reduced search space for directed acyclic graph (DAG) structures in scoring-based Bayesian network learning. The candidate DAG structures are restricted by clustering variables and determining the intercluster directionality. The proposed method considers cycles on only $c_{\max}(\ll n)$ variables rather than on all n variables for DAG structures. The R-CORE method could be a useful tool in very large problems where only a very small amount of training data is available.

key words: *Bayesian network, clustering, order restriction, search space reduction*

1. Introduction

The Bayesian network model has been used widely to describe probabilistic dependencies between variables. A Bayesian network is a directed acyclic graph (DAG) that includes parameters to describe the conditional probability distribution of the variables. Such conditional dependencies are represented by incoming edges to each variable. Formally, a Bayesian network B can be noted as $B = \langle G, P \rangle$, where G is a DAG that can be stated as $G = \langle \mathbf{V}, \mathbf{E} \rangle$, in which $\mathbf{V}$ is a set of random variables that correspond to the nodes in G and $\mathbf{E}$ is a set of directed edges between these nodes (we use the terms 'node' and 'variable' interchangeably in this paper). P is a joint probability distribution of the random variables in $\mathbf{V}$.

To describe probabilistic dependencies between variables using a Bayesian network model, a Bayesian learning procedure is conducted with the given instance values of the variables. There are two approaches for the learning of Bayesian networks: scoring-based and constraint-based. In scoring-based learning, the problem involves finding an optimal DAG that best fits the given data instances; in constraint-based learning, the presence of edges is determined using statistical dependency measures. Each approach has its own benefits. Scoring-based learning is robust and can handle noisy data, and we can use model averaging techniques to enhance the quality of the result when there is a small amount of training data. Constraint-based learning is generally faster than the scoring-based approach, and gives a trustworthy result if there are sufficient training data. For recent applications of Bayesian networks, there are problems where a large number of variables exist with a very small amount of training data. For example, analyses of biological genetic networks should handle thousands of genes simultaneously when there are only tens or hundreds of observed data instances [9], [11], [18], [20]. Here we focus on such large problems with a small amount of training data, and hence use scoring-based learning.

The scoring-based learning of a Bayesian network B comprises two parts; learning a DAG structure G and learning probabilistic parameters P. In this paper, we focus on learning DAG structures with given data, because this is much more problematic than learning the probabilistic parameters. Learning a DAG structure from given data involves finding an optimal DAG structure that best represents the conditional probabilistic dependencies of the variables. The common process for finding an optimal DAG structure can be stated as follows: given some scoring measure $Score$, such as the BDeu score [14] or the MDL score [12], [23], a DAG structure G_i must be found for which $Score(G_i|\mathbf{D})$ is maximal, where $\mathbf{D}$ is a given set of data instances. Conventional approximate search methods such as greedy hill climbing and genetic algorithms [8] have been used for structure learning because the number of candidate DAG structures is very large even for a small number of variables [21].

However, the efficiency of methods for learning structures needs to be improved for large problems. The number of variables considered here varies from several hundreds to thousands. This scale is much larger than that in previous conventional applications of approximate Bayesian learning, in which networks of only tens of variables have been considered. For example, one of the widely used benchmark Bayesian networks in conventional approximate structure learning is the ALARM network [4], which has 37 nodes. The infeasibility of implementing the learning procedure for networks with more than hundreds of variables using conventional approximate search methods has led to the proposal of various search space reduction approaches [5], [10], [16]. However, these previous methods exhibit limited

Manuscript received July 3, 2006.
Manuscript revised January 9, 2007.

[†]The author is with the Department of Electrical Engineering & Computer Science, KAIST, 373–1 Guseong-dong, Yuseong-gu, Daejeon 305–701, Republic of Korea.
[††]The author is with the Department of BioSystems, AITrc, KAIST, 373–1 Guseong-dong, Yuseong-gu, Daejeon 305–701, Republic of Korea.
[†††]The author is with the Department of BioSystems, KAIST, 373–1 Guseong-dong, Yuseong-gu, Daejeon 305–701, Republic of Korea.
a) E-mail: swjung@biosoft.kaist.ac.kr
b) E-mail: khlee@biosoft.kaist.ac.kr
c) E-mail: dhlee@biosoft.kaist.ac.kr
DOI: 10.1093/ietisy/e90–d.7.1018

scalability because there may be cycles over all n variables, making combinatorial searches on all n variables necessary when searching DAGs. In this paper, we propose a new recursive clustering and order restriction (R-CORE) method for the fast learning of very large Bayesian networks. The proposed method considers cycles on only $c_{\max}(\ll n)$ variables.

This paper is organized as follows. In Sect. 2, we describe previous approaches for the learning of large-scale Bayesian networks. Section 3 outlines and then provides a detailed description of the proposed method. An empirical evaluation is presented in Sect. 4 to show the effectiveness of our approach. Conclusions and further issues are mentioned in Sect. 5.

2. Learning Large-Scale Bayesian Networks

Even though there are only tens of variables in conventional Bayesian networks, the possible number of DAGs still becomes unmanageable since, for n nodes, it is [21]

$$f(n) = \sum_{i=1}^{n} (-1)^{i+1} \left(\begin{array}{c} n \\ i \end{array} \right) 2^{i(n-i)} f(n-i), \quad \text{for } n > 2 \tag{1}$$

$$f(1) = 1$$

$$f(0) = 1$$

For example, the possible number of DAGs is 3, 25, 29,281 and $\simeq 4.2 \times 10^{18}$ for n values of 2, 3, 5 and 10, respectively. Chickering [6] proved that the learning of Bayesian networks is an NP-complete problem. Because of the huge search space for DAGs, common approaches for conventional Bayesian learning have involved the use of approximate search algorithms such as greedy search. However, such algorithms exhibit insufficient scalability for application to cases with more than hundreds of variables. Instead, we need to reduce the search space for DAGs, for which two approaches can be used: assuming the variable order and restricting local structures.

Several Bayesian learning algorithms have been proposed for assuming that the variable order is already given [2], [7], [15], [22]. These algorithms search only DAG candidates that are consistent with the given order. However, they are not practical in real applications because the true order of variables is rarely known.

The local structure restriction approach is based on the sparsity of large networks, whereby each node has a relatively small number of incoming edges. From this observation, Friedman proposed the sparse candidate (SC) method [10] for reducing the search space for DAGs. The SC method reduces the search space by determining $k(\leq n)$ candidate parents for each node. When every other $n-1$ variables are candidate parents of a node, there are $O(2^{n-1})$ possible sets of parents for each node. In the SC method, the restriction to a maximum of k candidate parents results in the number of possible sets of parents being $O(2^k)$, with the actual number possibly being less due to the presence of cycles. It should be noted that there are other local structure restriction approaches [5], [16] that exhibit similar scalability.

Even though the local structure restriction approach is useful for reducing the search space for DAGs, it still has limitations. For the case of the SC method, the number of possible sets of parents for a variable is reduced to $O(2^k)$ from $O(2^n)$, but the DAG search remains a combinatorial search on all n variables (due to cycles existing over all variables). In the application of SC method [11], only hundreds of genes are selected as nodes among several thousands of genes due to the computational cost. Therefore, if we could reduce the number of variables where cycles are considered, we could significantly reduce the search space for DAGs.

3. Proposed Method

3.1 Approach

In this study, we used the modular approach to restrict candidate DAG structures of Bayesian networks on all the variables. The basic idea of the modular approach is based on the assumption that the entire network can be considered to be a 'network of subnetworks'. We restrict the candidate network structures by determining clusters of variables as subnetwork modules and considering the network structure between those clusters as a structure restriction. Our approach comprises the following two main steps:

- Structure restriction
  1. Cluster variables into $c_{\max}(\ll n)$ clusters.
  2. Determine the intercluster directionality with acyclicity.
  3. For large clusters, apply this step recursively.
- Maximization
  1. Find a DAG that best fits the given data while preserving the intercluster directionality determined in the structure restriction step.

Our objective is to restrict the DAG search space by restricting the space of considered cycles. By determining intercluster directionality in an acyclic manner, we can force the maximization step to consider cycles only in each cluster, and not for all the variables. However, a cluster being larger than our preferred $c_{\max}$ will result in the search space being larger than our preferred size. To avoid this situation, we determine directionality by recursively applying the structure restriction step to these large clusters. This approach is illustrated in Fig. 1.

Preserving the determined intercluster directionality in the maximization step restricts the DAG search process to considering cycles on at most $c_{\max}$ variables. In the structure restriction step, every process involved in determining the intercluster directionality is also performed with $c_{\max}$ clusters. Thus cycles are considered for at most $c_{\max}$ entities in the entire process of our approach, which presents a very restricted search space compared to the original.

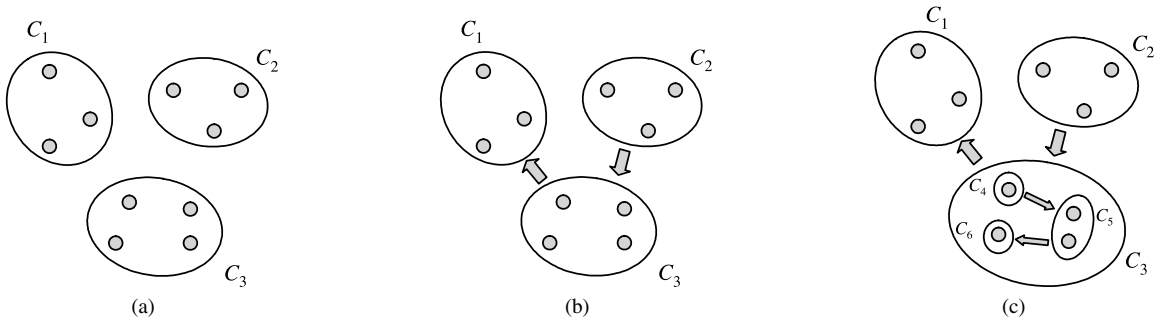


Fig. 1 Outline of the proposed structure restriction approach with $n = 10$ and $c_{\max} = 3$. (a) Clustering variables into three ($c_{\max}$) clusters. (b) Intercluster directionality between clusters. (c) Intercluster directionality in cluster C_3 (where $|C_3| > c_{\max}$).

3.2 Algorithm

3.2.1 Graph-Theoretic Unbalanced Partitioning

The structure restriction step of Sect. 3.1 clusters variables into $c_{\max}$ clusters and determines intercluster directionality between clusters. The determined intercluster directionality is preserved in the process of finding an optimal DAG structure, and thus we allow only one-direction edges between any two clusters according to the determined intercluster directionality. For this reason, any edge with a direction opposite to the given directionality cannot be found by our approach. Thus, our clustering objectives is to minimize the loss of reverse edges between clusters.

Pairwise measures have been used widely for estimating the presence of edges between variables, because of their simplicity. However, estimating edge directions in Bayesian network models with pairwise measures remains an open problem, and hence we loosen the objective of clustering so as to minimize the possible number of edges that can be lost between clusters. The presence of edges in Bayesian networks is estimated using a measure of the mutual dependence of two variables. The mutual information $I(X_i; X_j)$ between two random variables X_i and X_j is computed for a given set of data instances $\mathbf{D}$ according to

$$I(X_i; X_j) = \sum_{a \in X_i} \sum_{b \in X_j} P(a, b) \log_2 \left(\frac{P(a, b)}{P(a)P(b)} \right) \tag{2}$$

where $P(a), P(b)$ and $P(a, b)$ can be computed from given data instances $\mathbf{D}$. We use $w_{(i;j)} = I(X_i; X_j)$ to represent the degree of the presence of an undirected edge between two variables X_i and X_j.

For the set of all variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, our objective is to construct $c_{\max}$ ($\ll n$) clusters $\{C_1, \ldots, C_{c_{\max}}\}$ while minimizing the sum of the degree of undirected edges between clusters. To achieve this goal, we consider an undirected complete graph $G = (\mathbf{V}, \mathbf{E})$ in which each vertex V_i corresponds to the random variable X_i and all vertices are connected with each other (except themselves) with undirected edges. Each edge $e_{(i;j)}$ has a corresponding degree of edges $w_{(i;j)}$. To construct $c_{\max}$ clusters while minimizing the loss of the degree of edges, each $e_{(i;j)}$ is continuously eliminated from $\mathbf{E}$ in increasing order of $w_{(i;j)}$ until G is partitioned into $c_{\max}$ disconnected components $G_1, \ldots, G_{c_{\max}}$. The sets of variables corresponding to the vertices in the G_i components represent the clustering result. This unbalanced partitioning process, which does not consider the sizes of clusters, is presented in Alg. 1.

Algorithm 1 UnbalPartition($\mathbf{X}$, $\mathbf{D}$, $c_{\max}$)

```
1: G := (V, E) such that V_i ∈ V corresponds to X_i ∈ X and E = {e_(i;j) | ∀X_i, X_j ∈ X, i < j}
2: W := {w_(i;j) = I(X_i; X_j) | ∀e_(i;j) ∈ E)}
3:
4: while G has c < c_max disconnected components do
5:    e_min := e_(k;l)(∈ E) such that w_(k;l) is minimum ∀w_(i;j) ∈ W
6:    E := E \ {e_min}
7:    W := W \ {w_(k;l)}
8: end while
9:
10: C := ∅
11:
12: for all disconnected subgraph G_i = (V_i, E_i) in G such that 1 ≤ i ≤ c_max do
13:    C_i := {X_j | V_j ∈ V_i}
14:    C := C ∪ {C_i}
15: end for
16:
17: return C
```

3.2.2 Determining the Intercluster Directionality

To determine the intercluster directionality between clusters, we first define the *intercluster directionality* as in Definition 1.

Definition 1 (Intercluster directionality): *An intercluster directionality* δ for the set of clusters $\mathbf{C}$ is defined with a DAG between clusters $G = (\mathbf{C}, \mathbf{E})$.

An optimal intercluster directionality minimizes the number of edges between variables that violate the directionality between clusters. However, it is still an open problem to define the 'direction of conditional dependency' between two variables. For this reason, we attempt to estimate the intercluster directionality by finding a DAG structure between clusters; we use Bayesian network learning between

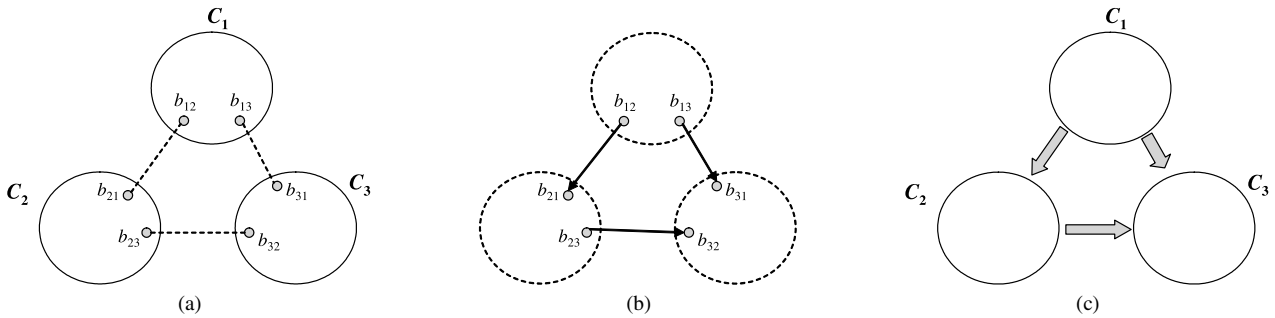


Fig. 2 Outline of the method used to determine the intercluster directionality. (a) Finding boundary variables. (b) Learning a DAG structure between boundary variables with the restriction of cluster acyclicity. (c) Determined intercluster directionality between clusters.

clusters for this task.

Because the variables are clustered with the graph-theoretic approach, each subgraph will contain 'boundary nodes' that connect the subgraph to the other subgraphs. We therefore assume that the conditional dependencies between clusters are determined with the dependencies between such boundary nodes in different clusters. For each cluster C_i, a variable X_i (which is the closest one to cluster C_j, $j \neq i$) is selected as *a representative boundary node* to C_j, and denoted as b_{ij}. When we consider a directed edge e_{ij} from cluster C_i to C_j in the process of Bayesian network learning between clusters, an edge from b_{ij} to b_{ji} is considered instead. Actually this approach is a heuristic and will work only when the target networks can be divided into several subnetworks, with boundary nodes playing a major role in dependency relationships. For other networks which do not have such characteristics, the proposed method may not work correctly. However, we consider many networks in real world may satisfy such characteristics to some extent and our method can be useful for such cases. From this approach, we take the cluster acyclicity restriction defined in Definition 2 in the process of Bayesian network learning for the set of boundary variables **B**.

Definition 2 (Cluster acyclicity for **C**): Let **B** be a set of boundary nodes for the clusters in **C**. Let $G_{\mathbf{B}}$ be a directed graph for **B** and $G_{\mathbf{C}}$ be a directed graph for clusters in **C**. If $G_{\mathbf{B}}$ satisfies the following restrictions:

- For $G_{\mathbf{B}}$, edges are allowed only between b_{ij} and b_{ji}, where $i \neq j$.
- If there is an edge from b_{ij} to b_{ji} in $G_{\mathbf{B}}$, there will also be an edge from C_i to C_j in $G_{\mathbf{C}}$. Then there is no cycle in $G_{\mathbf{C}}$.

we say that **C** exhibits *cluster acyclicity*.

An example of this idea is illustrated in Fig. 2. The algorithm for determining the intercluster directionality by restricting the cluster acyclicity is presented in Alg. 2.

3.2.3 The R-CORE Method

Sections 3.2.1 and 3.2.2 describe the methods used to cluster variables into $c_{\max}$ clusters and to determine the intercluster directionality. The objective of the latter is to reduce the complexity of the combinatorial search for DAGs in the process of Bayesian structure learning. However, the presence of large clusters that leads to a combinatorial search space larger than the desired one diminishes the benefit of this approach. This problem is tackled by using a recursive approach for such large clusters. For clusters with variables that are larger than the given $c_{\max}$ value, Alg. 1 (UnbalPartition) and Alg. 2 (C-Direct) are applied recursively until the

Algorithm 2 C-Direct(**C**, **D**)

```
δ := G such that G = (C, E), E = ∅
B := ∅

for i = 1 to c_max − 1 do
   for j = i + 1 to c_max do
      {X_k(∈ C_i), X_l(∈ C_j)} := argmax_{X_m∈C_i, X_n∈C_j} I(X_m; X_n)
      b_ij := X_k
      b_ji := X_l
      B := B ∪ {b_ij, b_ji}
   end for
end for

Find a G_B = (B, E_B) maximizing Score(G_B; D) with cluster acyclicity
   for C.

for all e^B_ij ∈ E_B do
   E := E ∪ {e_ij}
end for

return δ
```

Algorithm 3 R-Restrict(**X**, **D**, $c_{\max}$)

```
δ := ∅
C := UnbalPartition(X, D, c_max)
δ := C-Direct(C, D)
δ := δ ∪ {δ}

for all C_i such that 1 ≤ i ≤ c_max do
   if |C_i| > c_max then
      δ := δ ∪ R-Restrict(C_i, D, c_max)
   end if
end for

return δ
```

Algorithm 4 R-CORE($\mathbf{X}$, $\mathbf{D}$, $c_{\max}$)

1: /*Structure restriction*/
2: $\boldsymbol{\delta}$:= R-Restrict($\mathbf{X}$, $\mathbf{D}$, $c_{\max}$)
3:
4: /*Maximization*/
5: Find a G maximizing $Score(G; \mathbf{D})$ while preserving all of the intercluster directionality in the $\boldsymbol{\delta}$.
6: **return** G

clusters are smaller than $c_{\max}$. By determining the intercluster directionality in this way, we can restrict the search space of candidate DAG structures for Bayesian structure learning to the preferred size of the combinatorial search space using the parameter $c_{\max}$. This recursive restriction method – which returns the set of intercluster directionalities $\boldsymbol{\delta}$ – is presented in Alg. 3.

We use $\boldsymbol{\delta}$, which is the set of intercluster directionality δ, to learn a Bayesian network structure that best fits the given data while preserving all of the intercluster directionalities therein. A DAG G that preserves the given intercluster directionality δ can be stated as in Definition 3.

Definition 3 (Preserving intercluster directionality): Let $G = (\mathbf{V}, \mathbf{E})$ be a DAG for the corresponding $\mathbf{X}$, $\mathbf{C}$ be the set of clusters for $\mathbf{X}$ and $\delta = G_{\mathbf{C}}(\mathbf{C}, \mathbf{E}_{\mathbf{C}})$ be a given intercluster directionality for $\mathbf{C}$. If there exists a corresponding $e^{\mathbf{C}}_{ij} \in \mathbf{E}_{\mathbf{C}}$ for every $e_{ij} \in \mathbf{E}$, where X_i and X_j are in different clusters, we say that *G preserves the intercluster directionality δ*.

The complete R-CORE algorithm that uses Algs. 1–3 is presented in Alg. 4. This algorithm first determines the set of intercluster directionalities $\boldsymbol{\delta}$, and then finds a G that maximizes the given scoring measure $Score(G; \mathbf{D})$.

The R-CORE algorithm recursively restricts the global structure between clusters to a DAG-shaped one, which significantly reduces the number of candidate DAG structures and thus makes the DAG search process extremely fast. Further, the inclusion of the structure restriction step does not significantly increase the cost, since the total number of clusters considered in this step is $O(n/c_{\max})$. Bayesian structure learning for $O(c_{\max})$ clusters is conducted in each cluster to determine the intercluster directionality. If we denote all the DAGs of n variables (Eq. (1)) as $W(n)$, the number of considered DAG structures in the structure restriction step is only $W(c_{\max}) \cdot O(n/c_{\max})$, which is much lower than $W(n)$.

4. Empirical Evaluation

4.1 Experimental Environment

To show the effectiveness of the R-CORE method, we compare its results with those of SC (which we use as a representative local structure restriction approach). Six known Bayesian networks (Table 1) [1], [3], [4], [13], [17], [24] built for expert systems by human experts are used as benchmarks. The number of nodes was varied from 37 to 724. From each benchmark Bayesian network, 5000 and 10000 data instances were sampled as the training data sets.

Table 1 Benchmark Bayesian networks.

	Nodes	Edges	Mean indegree	Max indegree
ALARM	37	46	1.24	4
HAILFINDER	56	66	1.18	4
WIN95PTS	76	112	1.47	7
PATHFINDER	109	195	1.79	5
DIABETES	413	602	1.46	2
LINK	724	1125	1.55	3

The case of 5000 data instances are closer to our target applications where only small amount of training data is available while another case of 10000 data instances is close to more conventional problems where relatively sufficient amount of data is available. All of those benchmark Bayesian networks include conditional probability tables to describe target distributions and thus we can generate samples from the described distribution for training data. The R-CORE and SC methods were applied to the data instances to learn the target networks. Greedy hill climbing search was used in our experiments to find the DAG structures that maximized the score. The empty graph with no edges was used as the initial graph of the greedy search, which implies that we had no initial knowledge about the dependencies between the variables, and we initially assumed that there were no such dependencies. The scoring measure used in the search procedure was the BDeu score [14] with an equivalent sample size of 10. We selected the equivalent sample size before the experiment, and this was not tuned. The performance of the R-CORE method was evaluated for $c_{\max}$ values of 5, 10, 15, 20, 25, 30, 35 and 40. For the SC method, k values of 5, 10 and 15 were used. However, we were unable to obtain results for the cases of $k = 10$ and 15 for DIABETES and all k values for LINK with the SC method due to their high computational cost.

We first show the scores of the learned results for the R-CORE method for various values of the $c_{\max}$ parameter. We compare the results for the R-CORE and SC methods from two viewpoints: the size of the explored search space (which may reflect the learning speed) and the quality of the results. First, the size of the explored search space is assessed as the number of candidate DAG structures visited during the greedy search. Second, three categories are considered for quality evaluation: edge correctness, overall structural error and the likelihood (BDeu score); which are used widely to evaluate learning methods for Bayesian network structures [19].

4.2 BDeu Score of the R-CORE Method

Figure 3 shows the BDeu scores of the learned results for the R-CORE method to the training data of 5000 instances with different $c_{\max}$ values, where a higher score represents a better learning result. In most of the cases the BDeu score of the learned result increases with the value of $c_{\max}$, which corresponds to determining the intercluster directionality with a higher resolution. For example, the intercluster directionality with a $c_{\max}$ value of 40 is determined with 40 clusters on

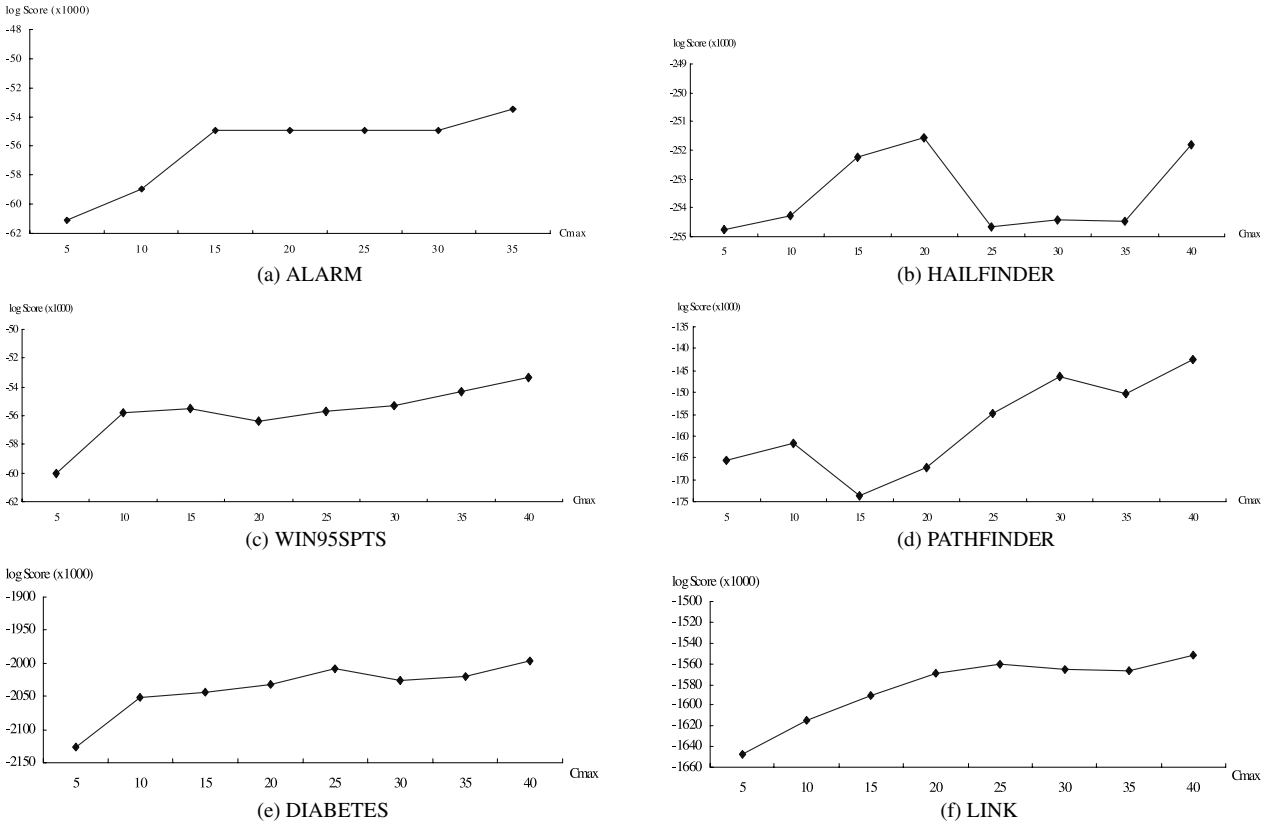


Fig. 3 Logarithm of the BDeu score of networks learned by the R-CORE method for different $c_{\max}$ values. The training data of 5000 instances was used.

each cluster hierarchy level, while a $c_{\max}$ value of 5 is determined with only 5 clusters. Using a lower number of clusters to determine the directionality will increases the number of missing edges with a direction opposite to the directionality in the maximization step. Note that if $c_{\max}$ is equal to the total number of variables, the R-CORE method works as a conventional Bayesian network learning algorithm with no search space restriction. Figure 3 shows only the case of 5000 data instances but another case of 10000 data instances also shows the same tendency of monotonic increasing of the score.

4.3 Evaluation Based on the Explored Search Space Size

For each of six benchmark Bayesian networks, the number of visited candidate DAGs was counted for the R-CORE and SC methods until the searches converged. The numbers of visited candidate DAGs in both methods are illustrated in Fig. 4 for 5000 data instances, which shows the results for only four of the benchmark Bayesian networks because the SC method failed to give learned results within a reasonable time for DIABETES and LINK networks. The results show the proposed R-CORE method learns target networks much faster than the SC method, which is due to the global structure restriction approach of the former considering a much smaller search space than the local structure restriction approach of the latter. In fact, the greedy search algorithm does not consider the entire search space, and hence the results do not represent the exact size of the considered search space for each case. However, the results do indicate the size of the reduced search space approximately when using the R-CORE method. This much faster speed of the R-CORE method was also found from the experiments for another training data of 10000 instances and that result is omitted here.

Table 2 shows the actual running time of both methods in minutes using an Intel Pentium4 3 GHz machine with 2GB RAM. Even though the actual running time may be different according to the implementation method of those algorithms, their relative difference in scale of running time will not change. This actual running time shows the strong benefit of the proposed method. As the number of nodes is increased in target networks, the running time grows super exponentially for the SC method. Moreover, SC did not converge in reasonable time for large networks. The proposed R-CORE method does not show such super exponential growth of running time. Thus R-CORE can be a very useful method for learning large networks in practical point of view.

4.4 Evaluation Based on the Quality of the Result

4.4.1 Precision and Recall of Edges

The R-CORE and SC methods were also compared in terms

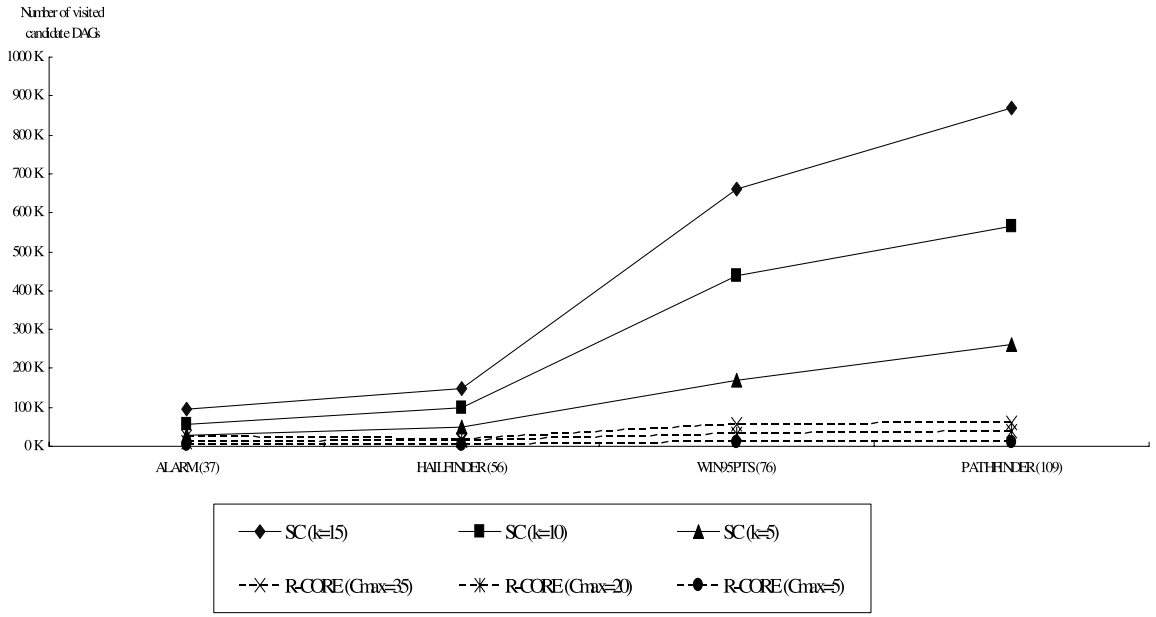


Fig. 4 The number of visited candidate DAGs for the R-CORE and SC methods with 5000 training data instances. The number of nodes in each network is given in the parentheses.

Table 2 Actual running time in minutes for both methods. This is for 5000 data instances.

Benchmark BN	R-CORE			SC		
	$c_{\max} = 5$	$c_{\max} = 20$	$c_{\max} = 35$	$k = 5$	$k = 10$	$k = 15$
ALARM	1	1	1	1	4	5
HAILFINDER	1	1	1	2	10	16
WIN95PTS	1	3	11	7	32	53
PATHFINDER	1	12	15	13	47	98
DIABETES	1	33	58	480	N/A	N/A

of the correctness of edges, which is measured by the precision and recall of edges defined as

$$Precision = \frac{Number\ of\ true\ edges\ in\ the\ learned\ network}{Number\ of\ edges\ in\ the\ learned\ network}$$

$$Recall = \frac{Number\ of\ true\ edges\ in\ the\ learned\ network}{Number\ of\ edges\ in\ the\ original\ network}$$

An edge in the learned network is a true edge if there is a corresponding edge that has the same start and end nodes in the original network. The precision of edges represents how exact the edges are in the learned network, and the recall of edges represents how many of the original edges are retrieved. The results are listed from Tables 3 to 6.

Tables 3 and 4 indicate that R-CORE and SC give comparable precision values, with the R-CORE results even being better in some cases. This implies that it is valid for the proposed R-CORE method to ignore a greater amount of the search space, where there are chances of adding false-positive edges to the learning result. Tables 5 and 6 indicates that the recall value is slightly lower for the R-CORE method than for the SC method, which implies that the former loses more true edges than the latter due to the intercluster directionality restriction. These results together indicate that the search space reduction of the R-CORE method improves the precision of edges but with a minor loss of true edges.

4.4.2 Overall Structural Error

Another criterion used for quality evaluation was the overall structural error. This was increased by 1 if an edge between two nodes in the learned network differed from the edge connection between corresponding nodes in the original network, and hence represents how many erroneous edge connections are in the result. Table 7 shows the best cases of structural error of learned networks with the R-CORE method and the SC method. The best cases of applying the R-CORE method and the SC method for each of ALARM, HAILFINDER, WIN95PTS and PATHFINDER are one of the cases of applying their parameter values $c_{\max}$ and k for each, where the single k value of 5 was applied for DIABETES due to the computational cost of SC. This result indicates that the structural errors of the two methods are comparable. Moreover, there are cases where the structural error is lower for the R-CORE method than for the SC method.

Table 3 Precision of edges for 5000 data instances. Best cases are indicated in boldface.

Benchmark	R-CORE								SC		
BN	$c_{\max}$ =5	10	15	20	25	30	35	40	k =5	10	15
ALARM	**0.474**	0.395	0.286	0.255	0.255	0.298	0.269	·	0.301	0.446	0.466
HAILFINDER	0.328	0.500	0.567	0.548	0.533	0.600	**0.644**	0.639	0.545	0.521	0.436
WIN95PTS	0.208	0.315	0.310	0.203	0.272	0.217	0.313	**0.317**	0.264	0.290	0.241
PATHFINDER	**0.460**	0.363	0.329	0.357	0.375	0.344	0.347	0.377	0.458	0.400	0.387
DIABETES	0.292	0.333	0.365	0.360	0.359	0.350	**0.367**	0.350	0.319	N/A	N/A

Table 4 Precision of edges for 10000 data instances. Best cases are indicated in boldface.

Benchmark	R-CORE								SC		
BN	$c_{\max}$ =5	10	15	20	25	30	35	40	k =5	10	15
ALARM	0.353	0.396	0.356	0.379	0.393	0.388	0.429	·	0.285	0.4	**0.472**
HAILFINDER	0.355	0.348	0.417	0.394	0.462	0.485	0.515	0.414	0.492	**0.569**	0.486
WIN95PTS	0.269	0.221	0.200	0.263	0.293	0.267	**0.320**	0.209	0.263	0.258	0.286
PATHFINDER	0.428	0.448	0.366	0.425	0.412	0.456	0.459	0.388	**0.461**	0.413	0.451
DIABETES	**0.320**	**0.320**	0.292	0.294	0.312	0.302	0.307	0.319	0.319	N/A	N/A

Table 5 Recall of edges for 5000 data instances. Best cases are indicated in boldface.

Benchmark	R-CORE								SC		
BN	$c_{\max}$ =5	10	15	20	25	30	35	40	k =5	10	15
ALARM	0.391	0.326	0.304	0.283	0.283	0.304	0.304	·	0.347	0.543	**0.608**
HAILFINDER	0.288	0.439	0.515	0.515	0.485	0.545	0.576	**0.591**	0.545	0.545	0.469
WIN95PTS	0.268	0.366	0.393	0.250	0.330	0.295	0.411	0.393	0.401	**0.562**	0.491
PATHFINDER	0.323	0.272	0.262	0.287	0.308	0.318	0.303	0.323	**0.400**	0.389	0.389
DIABETES	0.244	0.281	0.312	0.314	0.317	0.311	**0.322**	0.314	0.302	N/A	N/A

Table 6 Recall of edges for 10000 data instances. Best cases are indicated in boldface.

Benchmark	R-CORE								SC		
BN	$c_{\max}$ =5	10	15	20	25	30	35	40	k =5	10	15
ALARM	0.261	0.413	0.457	0.478	0.478	0.413	0.522	·	0.347	0.478	**0.565**
HAILFINDER	0.333	0.364	0.455	0.424	0.455	0.485	0.530	0.439	0.515	**0.621**	0.545
WIN95PTS	0.259	0.304	0.321	0.411	0.455	0.393	0.518	0.375	0.392	0.473	**0.544**
PATHFINDER	0.303	0.379	0.349	0.395	0.395	0.395	0.400	0.390	0.405	0.405	**0.451**
DIABETES	0.282	0.307	0.311	0.316	0.341	0.322	0.324	**0.346**	0.307	N/A	N/A

Table 7 Structural errors of best cases for both methods.

Benchmark	5000 data		10000 data	
BN	R-CORE	SC	R-CORE	SC
ALARM	38	33	34	30
HAILFINDER	39	53	55	46
WIN95PTS	132	150	138	142
PATHFINDER	191	192	189	188
DIABETES	587	613	612	579

Table 8 Likelihood of best cases for both methods. The shown values are logarithm of BDeu scores divided by 1000.

Benchmark	5000 data		10000 data	
BN	R-CORE	SC	R-CORE	SC
ALARM	−53.4	−51.4	−102.9	−102.3
HAILFINDER	−251.5	−250.6	−497.6	−497.4
WIN95PTS	−53.3	−49.0	−103.2	−96.6
PATHFINDER	−142.5	−136.4	−265.5	−261.3
DIABETES	−1996.9	−1991.8	−3444.8	−3874.6

For the ALARM case, there was no case where the R-CORE method outperforms the SC method for any parameter value and for both sets of data instances. The reason for this may be the target network is too small for R-CORE to be effective. Because the R-CORE method restricts the DAG search space much more than the SC method, it always have the possibility to give worse results than those given by the SC method. This weakness can be more significant for small networks like ALARM, where the entire DAG search space is relatively smaller than larger networks and the SC method may fall into local minima with less possibility.

When we consider those two data sets of 5000 and 10000, the result clearly shows that the proposed R-CORE method is more efficient when there are relatively small amount of training data. The R-CORE method also shows comparable result for the larger training data set of 10000. Thus the proposed method can be a better approach for the large problems where limited training data is available.

4.4.3 Likelihood

The likelihood of the learned Bayesian network structure is represented by the BDeu score, which implies $P(G|\mathbf{D})$. Table 8 shows the BDeu score of the best cases for both methods. The R-CORE method shows slightly less likelihood for most of the cases except for the case of DIABETES with 10000 data instances. This result was already expected be-

cause the R-CORE method restricts much more DAG search spaces than the SC method and thus has more chance of losing DAG candidates of high scores during the search. However, we can find that this weakness in likelihood comparison does not always correspond to the result of structural quality as shown in Sect. 4.4.2.

4.5 Summary of the Evaluation

The proposed R-CORE method reduces the search space for DAGs in Bayesian network learning from a combinatorial search of considering cycles on all n variables to that for only $c_{\max}(\ll n)$ entities. This reduction in the search space results in the R-CORE method learning target networks much faster than the widely used SC method. Moreover, the R-CORE method can learn large networks that SC method cannot learn. If we increase the $c_{\max}$ value of R-CORE, the score of the learned result generally still increases due to a larger search space being covered. From the structural viewpoint, R-CORE retrieves slightly fewer true edges than does SC in most cases. However, the precision values are comparable for R-CORE and SC, with the former showing superior values in some cases. This results in R-CORE showing comparable structural quality with SC because it effectively ignores (worthless) search spaces that may increase the chance of adding false-positive edges to the answer. Our empirical evaluation has therefore shown that the proposed R-CORE method achieves much faster learning than the SC method without a loss of learning quality. Thus the proposed method can be a comparable tool to the SC method while learning target networks much faster. Moreover, we can expect even better structural quality than the SC method for the large problems where limited amount of training data is available.

5. Conclusions

In this paper, we propose the R-CORE method for reducing the search space for the DAGs of large-scale Bayesian network learning. We assume that target networks can be divided into several subnetworks, with boundary nodes playing a major role in dependency relationships. To reduce the search space for candidate DAG structures, we recursively cluster the nodes and determine intercluster directionality on each level of the cluster hierarchy. The size of the search space is significantly reduced by restricting every search procedure for DAGs to the combinatorial search of considering cycles on at most $c_{\max}(\ll n)$ variables or clusters by preserving the determined intercluster directionalities. Our empirical evaluation with benchmark Bayesian networks shows that our proposed method learns target networks much faster than the widely used SC method. Further, even though the proposed R-CORE method considers much smaller search space than the SC method, the learned results for the two methods are of comparable quality. The proposed method can be more efficient for the problems where limited amount of training data is available. We used greedy hill climbing to explore DAG candidates in the experiments described here, but more accurate (and slower) learning methods could also be used to explore DAG candidates because the search space is much smaller than that for the SC method. Additionally, iterative or evolutionary learning search techniques could be applied.

There are several ways in which this research could be extended. We clustered the nodes in target networks with an unbalanced graph partitioning algorithm. We could validate the clusters chosen using appropriate cluster validation methods based on cluster sizes and the partitioning errors. Such an approach may yield guidelines for the preferred speed and quality of the complete algorithm. Further, we could apply this global structure restriction approach to the learning of a dynamic Bayesian network, for which the search space is much larger than that of conventional Bayesian networks.

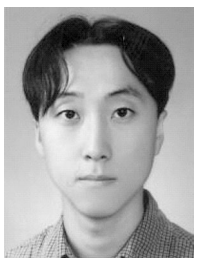
Acknowledgements

This work was supported by the Korean Systems Biology Research Grant (2005-00343) from the Ministry of Science and Technology, National Research Laboratory Grant (2005-01450) from the Ministry of Science and Technology. We would also like to thank CHUNG Moon Soul Center for BioInformation and BioElectronics and the IBM SUR program for providing research and computing facilities.

References

[1] B. Abramson, J. Brown, W. Edwards, A. Murphy, and R.L. Winkler, "Hailfinder: A Bayesian system for forecasting severe weather," Int. J. Forecasting, vol.12, no.1, pp.57–71, 1996.
[2] S. Acid and L.M. Campos, "BENEDICT: An algorithm for learning probabilistic belief networks," Proc. 6th International Conference IPMU'96, Granada, 1996.
[3] S. Andreassen, R. Hovorka, J. Benn, K.G. Olesen, and E.R. Carson, "A model-based approach to insulin adjustment," Proc. Third Conference on Artificial Intelligence in Medicine, pp.239–248, 1991.
[4] I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and G.F. Cooper, "The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks," Proc. Second European Conference on Artificial Intelligence in Medicine, pp.689–693, London, 1989.
[5] L.E. Brown, I. Tsamardinos, and C.F. Aliferis, "A novel algorithm for scalable and accurate Bayesian network learning," MEDINFO, 2004.
[6] D.M. Chickering, "Learning Bayesian networks is NP-complete," AI & STAT V, 1996.
[7] G.F. Cooper and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data," Mach. Learn., vol.9, pp.309–347, 1992.
[8] R. Etxeberria, P. Larrañaga, and J.M. Picaza, "Analysis of the behaviour of genetic algorithms when learning Bayesian network structure from data," Pattern Recognit. Lett., vol.18, pp.1269–1273, 1997.
[9] V. Filkov, S. Skiena, and J. Zhi, "Identifying gene regulatory networks from experimental data," Proc. RECOMB, 2001.
[10] N. Friedman, I. Nachman, and D. Pe'er, "Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm," Proc. Fifteenth Conference on Uncertainty in Artificial Intelligence, pp.206–215, 1999.

[11] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian networks to analyze expression data," J. Computational Biology, vol.7, pp.601–620, 2000.
[12] P. Grunwald, "A tutorial introduction to the minimum description length principle," in Advances in Mimimum Description Length: Theory and Applications, MIT Press, 2004.
[13] D.E. Heckerman, E.J. Horvitz, and B.N. Nathwani, "Toward normative expert systems: Part I the pathfinder project," Methods of Information in Medicine, vol.31, pp.90–105, 1992.
[14] D. Heckerman, D. Gerger, and D.M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," Mach. Learn., vol.20, pp.197–243, 1995.
[15] E. Herskovits and G. Cooper, "Kutato: An entropy-driven system for construction of probabilistic expert systems from databases," Proc. 6th International Conference on Uncertainty in Artificial Intelligence, pp.117–128, Cambridge, MA, 1990.
[16] K. Hwang, J. Lee, S. Chung, and B. Zhang, "Construction of large-scale Bayesian networks by local to global search," PRICAI 2002, pp.375–384, 2002.
[17] C.S. Jensen and A. Kong, "Blocking Gibbs sampling for linkage analysis in large pedigrees with many loops," Research Report, 1996.
[18] A.T. Kwon, H.H. Hoos, and R. Ng, "Inference of transcription regulation relationships from gene expression data," Bioinformatics, vol.19, no.8, pp.905–912, 2003.
[19] R.E. Neapolitan, Learning Bayesian Networks, Pearson Prentice Hall, 2004.
[20] P.H. Lee and D. Lee, "Modularized learning of genetic interaction networks from biological annotations and mRNA expression data," Bioinformatics, vol.21, no.11, pp.2739–2747, 2005.
[21] R.W. Robinson, "Counting labeled acyclic digraphs," in New Directions in the Theory of Graphs, pp.239–273, Academic Press, New York, 1973.
[22] J. Suzuki, "Learning Bayesian belief networks based on the MDL principle: An efficient algorithm using the branch and bound technique," IEICE Trans. Inf. & Syst., vol.E82-D, no.2, pp.356–367, Feb. 1999.
[23] J. Suzuki, "Learning Bayesian belief networks based on the minimim description length principle: Basic properties," IEICE Trans. Fundamentals, vol.E82-A, no.10, pp.2237–2245, Oct. 1999.
[24] http://www.cs.huji.ac.il/labs/compbio/Repository/Datasets/win95pts/win95pts.htm



Sungwon Jung received the B.S., M.S. and Ph.D. degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST), Korea in 1998, 2000 and 2007, respectively. Currently, he is a post-doctoral fellow at IBM-KAIST Bio-Computing Research Center, Department of BioSystems, KAIST. His research interests include artificial intelligence, bioinformatics, machine learning and medical informatics.

Kwang Hyung Lee received D.E.A and Dr.Ing. degrees from the Department of Computer Science, INSA de Lyon University, France, in 1982 and 1985, respectively, and the Dr.Etat degree from the Department of Computer Science, INSA de Lyon University, France, in 1988. He is a professor in Department of BioSystems and Department of Computer Science, and a chair professor of Mirae Corporation and the dean of academic affairs at KAIST. His research interests include fuzzy systems, artificial intelligence and bioinformatics.

Doheon Lee received the B.S., M.S., and Ph.D. degrees in computer science from KAIST, Daejeon, Korea, in 1990, 1992 and 1995, respectively. He has conducted visiting researches in University of Texas, Austin, and National Institutes of Health, Bethesda, in 1999 and 2002, respectively. He is now an Associate Professor in Department of BioSystems, KAIST. His research interests include bio-data mining, biosystem modeling and bioinformatics. Dr. Lee is and Associate Editor for ACM Transactions on Internet Technology and Computers in Biology and Medicine.