DEMONSTRATIO MATHEMATICA
Vol. XXXV No 3 2002

Mieczysław A. Kłopotek

A NEW SPACE-SAVING BAYESIAN TREE CONSTRUCTION METHOD FOR HIGH DIMENSIONAL DATA

Abstract. Bayesian networks have many practical applications due to their capability to represent joint probability distribution in many variables in a compact way. There exist efficient reasoning methods for Bayesian networks. Many algorithms for learning Bayesian networks from empirical data have been developed. A well-known problem with Bayesian networks is the practical limitation for the number of variables for which a Bayesian network can be learned in reasonable time. A remarkable exception here is the Chow/Liu algorithm learning tree-like Bayesian networks. However, also this algorithm has an important limitation, related to space consumption. The space required is quadratic in the number of variables. The paper presents a novel algorithm overcoming this limitation for the tree-like class of Bayesian networks. The new algorithm space consumption grows linearly with the number of variables while the execution time is comparable with the Chow/Liu algorithm. This opens new perspectives in construction of Bayesian networks from data containing thousands and more variables, e.g. in automatic text categorization.

1. Introduction

Currently, Bayesian networks [14] appear to be quite a popular method of representation of uncertain knowledge. They can represent concisely a joint multivariate discrete probability distribution exploiting properties of conditional independence. A Bayesian network is an acyclic directed graph (dag) nodes of which are labeled with variables and conditional probability tables of the node variable given its parents in the graph. The joint probability distribution is then expressed by the formula:

$$P(x_1, \ldots, x_n) = \prod_{i=1\ldots n} P(x_i|\pi(x_i))$$

where $\pi(x_i)$ is the set of parents of the variable (node) $X_{i.}$.

1991 *Mathematics Subject Classification*: 68T30.
Key words and phrases: machine learning, Bayesian networks, Chow/Liu algorithm, spatial complexity.

On the one hand, Bayesian networks allow for efficient reasoning, and on the other many algorithms for learning Bayesian networks from empirical data have been developed [8].

A well-known problem with Bayesian networks is the practical limitation for the number of variables for which a Bayesian network can be learned in reasonable time. A remarkable exception here is the Chow/Liu [4, 5] algorithm learning tree-like Bayesian networks. However, also this algorithm has an important limitation, related to space consumption. The space required is quadratic in the number of variables.

The paper presents a novel algorithm overcoming this limitation for the tree-like class of Bayesian networks. The new algorithm space consumption grows linearly with the number of variables while the execution time is comparable with the Chow/Liu algorithm. This opens new perspectives in construction of Bayesian networks from data containing thousands and more variables, e.g. in automatic text categorization.

Section 2 presents a brief introduction to the Chow/Liu algorithm. In Section 3 the new algorithm is proposed. In Section 4, the behavior of the Chow/Liu algorithm and of the new algorithm for tree-like underlying distributions are investigated. In Section 5, behavior of both algorithms for general type probability distributions is studied. Section 6 summarizes experiments with the Chow/Liu and the new algorithm. Section 7 contains some concluding remarks.

2. The Chow/Liu Algorithm

By a tree-like Bayesian network we understand a quadruple $(\mathbf{X}, \mathbf{E}, P_{\mathbf{X}}, P_{\mathbf{E}})$ where $\mathbf{E}$ is a set of edges constituting a tree over the set of nodes $\mathbf{X}$, $P_{\mathbf{X}}$ is a set of marginal probability distributions for elements of $\mathbf{X}$, and $P_{\mathbf{E}}$ is a set of probability distributions for edges from $\mathbf{E}$ such that for each edge $XY = \{X, Y\}$ $P_{\mathbf{E}}(XY)$ is marginally consistent with $P_{\mathbf{X}}(X)$ and $P_{\mathbf{X}}(Y)$.

For any partial order $\prec$ of nodes such that for each edge $\{X, Y\}$ either $X \prec Y$ or $Y \prec X$ and for no two edges $\{X, Y\}$, $\{X, Z\}$ both $Z \prec X$ and $Y \prec X$ hold, and there exists $X0$ being a node such that for no $Y \in \mathbf{X}$ $Y \prec X0$ holds, the joint probability distribution represented by the Bayesian network, is given by:

$$P(\mathbf{X}) = P_{\mathbf{X}}(X0) \cdot \prod_{\{X,Y\} \in \mathbf{E}, X \prec Y} P_{\mathbf{E}}(\{X, Y\}) / P_{\mathbf{X}}(X).$$

The best known algorithm for construction of tree-like Bayesian networks from data seems to be the Chow/Liu algorithm [4, 5]. For probability distributions described by tree-like Bayesian networks it recovers robustly the underlying tree structure and for general type probability distributions

it recovers the closest tree-like Bayesian network [16, 17]. It exploits the idea of maximum weight spanning tree, with dependence measure $DEP(X,Y)$ between variables X, Y equal to

$$DEP(X,Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x) \cdot P(y)}$$

where x, y run through the domains of X and Y respectively. $P(x,y)$ is the probability of co-occurrence of the events $X = x$ and $Y = y$, in practice it is calculated as relative frequency from some database.

The basic outline of the algorithm is as follows:

Algorithm CL(D,X)

(**D** is a probability distribution over a set of variables including the set of variables **X**)

1. Let **X** be the set of (discrete) variables. Find $X_1, X_2 \in \mathbf{X}$ such that $DEP(X_1, X_2) \geq DEP(Y_1, Y_2)$ for any $Y_1, Y_2 \in \mathbf{X}$.
2. Form two sets of nodes **T**, **N**, and the set of edges **E**, and initialize **T**= $\{X_1, X_2\}$, **N**=**X**-**T**, **E**= $\{(X_1, X_2)\}$.
3. If **N** is empty, then **STOP**.
4. Otherwise find $X_1 \in \mathbf{T}, X_2 \in \mathbf{N}$ such that $DEP(X_1, X_2) \geq DEP(Y_1, Y_2)$ for any $Y_1 \in \mathbf{T}, Y_2 \in \mathbf{N}$.
5. Update $\mathbf{E} := \mathbf{E} \cup \{(X_1, X_2)\}$, $\mathbf{T} = \mathbf{T} \cup \{X_2\}$, $\mathbf{N} = \mathbf{N} - \{X_2\}$.
6. Go to step 3.

End of Algorithm

As a result $\mathbf{Tr} = (\mathbf{X}, \mathbf{E})$ is the tree being the backbone (the acyclic graph) of the resulting tree-like Bayesian network.

Notice that the algorithm of Chow/Liu relies on the following property of the $DEP()$: If in the intrinsic Bayesian network the node Z lies on the path from node X to node Y, then $DEP(X,Z) > DEP(X,Y) < DEP(Y,Z)$.

The most time-consuming step of the algorithm is the calculation of $DEP(X,Y)$, because it is connected to calculations involving all records from the database. In step 1 $DEP(X,Y)$ is accessed $(card(X) - 1) \times card(X)/2$ times and upon each execution of step 4 it is accessed $(card(\mathbf{T}) - 1) \cdot card(\mathbf{N})$ times. If $card(\mathbf{X}) = n$, then the total amounts to $(n \cdot (n-1)/2) + n \cdot n \cdot (n-1)/2 - (2n-1) \cdot (n-1) \cdot n/6$ which grows with n^3 for large n. It is easily seen that for a given pair of variables X, Y, $DEP(X,Y)$ is accessed many (up to n) times.

For purposes of time saving the practical implementations create a table $TDEP[X,Y]$ for storing the values of $DEP(X,Y)$ so that we need to calcu-

---

T means set of "treated" nodes and **N** - set of "not treated" nodes

late the $DEP()$ function only $(n \cdot (n-1)/2)$ times. For large n the number of times the whole database is searched through is proportional to n^2 only. The algorithm changes then to:

Algorithm CL1(D,X)

(**D** is a probability distribution over a set of variables including the set of variables **X**)

1. Let **X** be the set of (discrete) variables. For each $X_1, X_2 \in \mathbf{X}$ calculate $TDEP[X_1, X_2] = DEP(X_1, X_2)$.
2. Find $X_1, X_2 \in \mathbf{X}$ such that $TDEP[X_1, X_2] \geq TDEP[Y_1, Y_2]$ for any $Y_1, Y_2 \in \mathbf{X}$.
3. Form two sets of nodes **T**, **N**, and the set of edges **E**, and initialize $\mathbf{T} = \{X_1, X_2\}$, $\mathbf{N} = \mathbf{X} - \mathbf{T}$, $\mathbf{E} = \{(X_1, X_2)\}$.
4. If **N** is empty, then **STOP**.
5. Otherwise find $X_1 \in \mathbf{T}, X_2 \in \mathbf{N}$ such that $TDEP[X_1, X_2] \geq TDEP[Y_1, Y_2]$ for any $Y_1 \in \mathbf{T}, Y_2 \in \mathbf{N}$.
6. Update $\mathbf{E} := \mathbf{E} \cup \{(X_1, X_2)\}, \mathbf{T} = \mathbf{T} \cup \{X_2\}, \mathbf{N} = \mathbf{N} - \{X_2\}$.
7. Go to step 3.

End of Algorithm

Further reductions in time consumption are possible (see e.g. [11, 12, 13]).

Though in $TDEP[]$ we do not need the diagonal elements and the table is symmetric ($DEP(X, Y) = DEP(Y, X)$), it requires still $n(n-1)$ cells, which may be prohibitive even for moderate size $n = 10,000$ which may be required in free text applications. The goal of this paper is to propose a new algorithm for building the tree-like Bayesian networks with memory consumption proportional to n and with time complexity not exceeding the CL1 algorithm.

The performance improvements are important due to many applications of this algorithm, e.g. as starting phases of other Bayesian network learning algorithms [3, 15], in Bayesian classifiers of TAN-type [6, 1] etc. In particular, applications in domains requiring usage of large Bayesian networks (with thousands of nodes) like intelligent genetic algorithms for feature selection [7] or free text classification [12], the space consumption of main memory may be a critical factor (disk access would slow down the process beyond any acceptable limits).

3. The Description of the New Algorithm

The new algorithm relies on the following paradigm:

Imagine that for the set $\mathbf{X}$ you define a series of sets $\mathbf{X}_2 \subset \mathbf{X}_3 \subset \mathbf{X}_{n-1} \subset \mathbf{X}_n = \mathbf{X}$ with $card(\mathbf{X}_i) = i$. Let $\mathbf{Tr}_i = (\mathbf{X}_i, \mathbf{E}_i)$ be a tree constructed by the algorithm CL1 for a given set of nodes $\mathbf{X}_i$ and the background database. Ei be the set of triples $(X,Y,DEP(X,Y))$ with $X, Y \in \mathbf{X}_i$ By the way, we can consider the problem of building $\mathbf{Tr}_i$ as a problem of building a Bayesian network from data with hidden (or latent) variables $\mathbf{X} - \mathbf{X}_i$ We claim now that we can construct $\mathbf{Tr}_i$ from $\mathbf{Tr}_{i-1}$ and the set of dependences $DEP(X, X_i)$ with $X \in \mathbf{X}_{i-1}$ and X_i being the only element from $\mathbf{X}_i - \mathbf{X}_{i-1}$.

Below we present the new algorithm, followed by the proof of its correctness.

Algorithm IT(D,X)

($\mathbf{D}$ is a probability distribution over a set of variables including the set of variables $\mathbf{X}$)

1. Define the sequence of sets $\mathbf{X}_2 \subset \mathbf{X}_3 \subset \mathbf{X}_{n-1} \subset \mathbf{X}_n = \mathbf{X}$ with $\mathbf{X}_i = \{X_1, X_2, \ldots, X_i\}$ for $i = 2, \ldots, n$.
2. Initialize $\mathbf{Tr}$ as $\mathbf{Tr}$=($\mathbf{T}$=$\{X_1, X_2,\}$,$\mathbf{E}$=$\{(\ X_1, X_2, DEP(X_1, X_2))\}$ and $i := 2$.
3. $i := i + 1$.
4. if $i > n$ **STOP**.
5. Create the set of edges $\mathbf{E}' = \{(X, X_i, DEP(X, X_i)) | X \in \mathbf{T}\}$.
6. $\mathbf{E}^x = \mathbf{E} \cup \mathbf{E}'$.
7. Find the edge $e = (X, Y, DEP(X, Y))$ from $\mathbf{E}^x$ such that for any edge $e' = (X', Y', DEP(X', Y'))$ from $\mathbf{E}^x$ $DEP(X, Y) \geq DEP(X', Y')$ holds.
8. Initialize the sets $\mathbf{T}'' = \{X, Y\}$, $\mathbf{E}'' = \{e\}$, $\mathbf{N}'' = \mathbf{T} - \mathbf{T}''$.
9. $\mathbf{E}^x = \mathbf{E}^x - \{e\}$.
10. If $\mathbf{N}''$ is empty, then $\mathbf{E} := \mathbf{E}''$, $\mathbf{T} := \mathbf{T}''$, go to step 3.
11. In $\mathbf{E}^x$ find an edge $e = (X, Y, DEP(X, Y))$ with $X \in \mathbf{T}''$, $Y \in \mathbf{N}''$, such that for any edge $e' = (X', Y', DEP(X', Y'))$ from $\mathbf{E}^x$. with $X' \in \mathbf{T}''$, $Y' \in \mathbf{N}''$, $DEP(X, Y) \geq DEP(X', Y')$ holds.
12. For that edge e set: $\mathbf{T}'' = \mathbf{T}'' \cup \{Y\}$, $\mathbf{E}'' = \mathbf{E}'' \cup \{e\}$, $\mathbf{N}'' = \mathbf{N}'' - \{Y\}$.
13. Go to step 10.

End of Algorithm

It is obvious that the result of IT is a tree-like Baysian network.

Before analyzing theoretical properties of IT, let us stress here that this algorithm is space-saving. Instead of about $n^2/2$ cells required by CL1, it needs at most $2(n-1)$ cells for storing DEP: $(n-1)$ cells for storing distances

On complexities resulting from existence of hidden variabkles for learning Bayesian networks from data consult e.g. [10, 9].

between nodes in the current tree and $(n-1)$ cells for storing distances of the ingoing node to the nodes currently in the tree.

4. Claims about the New Algorithm

We would like to assume subsequently that the algorithm is applied to data stemming from a tree-like distribution. We demonstrate the rationality of the new algorithm.

Let us denote by CL(**Tr**(**X**,**E**), **X'**) the tree grown for the set $\mathbf{X}' \subseteq \mathbf{X}$ of nodes given the intrinsic distribution is based on the tree **Tr**. Let CLS(**Tr**,**X'**) be the construction sequence of the tree CL(**Tr**,**X'**) (that is the sequence by which CL included the edges into the tree).

Algorithm RCL(AB,**Tr**,**X**)

Let us define the reduced Chow/Liu algotrithm (RCL(AB,**Tr**,**X**)) with A, B being nodes from **X**, in such a way that in the algorithm CL(**Tr**,**X**) we replace step 1 with the following:

1. set X_1 to A, and X_2 to B.

End of Algorithm

By RCLS(AB, **Tr**,**X**) let us denote the construction sequence of this algorithm.

Subsequently we show that many assumptions of the Chow/Liu algorithm may be weakened. For example, the step 1 of CL, seeking the node pair with maximum DEP, requires calculation of DEP for all pairs of nodes. We do not need to look for such a pair at the very offset of the algorithm, as the following Proposition demonstrates.

PROPOSITION 1. *In the Chow/Liu algorithm, the initial edge can be any edge of which we know it is a true edge. That is if AB is an edge in* **E** *in* $\mathbf{Tr}(\mathbf{X}, \mathbf{E})$, *then* $CL(\mathbf{Tr}(\mathbf{X}, \mathbf{E}), \mathbf{X})$ *yields the same result as* $RCL(AB, \mathbf{Tr}, \mathbf{X})$.

Proof. Consider any subgraph **T** being a tree of the intrinsic tree **Tr** underlying the distribution. If we start step 4 of CL with this tree **T**, then only a node neighboring in **Tr** with a node in **T** has a chance to be attached to the tree **T** because for any other node Y from outside of **T** there exists a node Z outside of **T** on the path from Y to any node X in **T** so that $DEP(X,Z) > DEP(X,Y)$ and so Y has no chance to be selected. The node Y from outside of **T** neighboring in **Tr** with the node Z inside of **T** can only be connected to Z as for any other node X in **T** Z is on the path from Y to X so that $DEP(Y,Z) > DEP(X,Y)$ and hence X has no chance to be connected with Y.

---

the pair of nodes maximizing DEP is just such an edge.

Therefore step 4 and subsequent ones of CL will transform any tree-like subgraph of **Tr** into a tree-like subgraph of **Tr**. Any edge in **Tr** is by itself a tree-like subgraph of **Tr**, hence starting from it we will reach finally the graph **Tr**. ∎

Potentially, we do not need even to investigate all the pairs of nodes subsequently, as the following Proposition demonstrates.

PROPOSITION 2. *If the removal of an edge AB from a tree* **Tr** *splits the tree* **Tr** *into parts Pa (containing A) and Pb (containing B), both being themselves trees, then we can construct the tree* **Tr** *first by growing a tree* **Tr***a from nodes of Pa plus the node B using CL, then by growing a tree* **Tr***b from nodes of Pb plus the node A using CL, and then joining the sets of edges from* **Tr***a and* **Tr***b.*

Proof. From Proposition 1 it is obvious that **Tr**a is identical with Pa plus edge AB, **Tr**b is identical with Pb plus edge AB. Hence their sum is just **Tr**. ∎

However, these insights are insufficient to claim properties of the algorithm IT. We need to know the fate of edges and missing edges in the tree from one set of nodes to the other (X_i, X_{i+1}) to show that comparisons of distances ignored by IT algorithm do not affect decisions made by CL algorithm.

PROPOSITION 3. *Let* $\mathbf{Tr}(\mathbf{X}, \mathbf{E})$ *be a tree. Let* $CL(\mathbf{Tr}, \mathbf{X}')$ *be obtained for* $\mathbf{X}' \subseteq \mathbf{X}$. *Let Z be a node from* $\mathbf{X} - \mathbf{X}'$. *Let A, B be in* $\mathbf{X}'$. *Let Z be on path from A to B in* **Tr**. *Then AB cannot be in* $CL(\mathbf{Tr}, \mathbf{X}' \cup \{Z\})$.

Proof. Notice that

$$DEP(A,B) < DEP(Z,A) \quad \text{and} \quad DEP(A,B) < DEP(Z,B).$$

Let us consider the construction sequence $\mathrm{CLS}(\mathbf{Tr}, \mathbf{X}' \cup \{Z\})$, in particular the point when one of the two nodes A, B is included into the tree grown so far, and the other isn't. Assume A is already included (the case B is symmetric). Now either Z is already included or is not. Assume first that Z is included. Then the edge AB cannot be included because AZ is a competing candidate with greater DEP ($DEP(A,B) < DEP(Z,A)$). Now assume Z is already included. Then the edge AB cannot be included because BZ is a competing candidate with greater DEP ($DEP(A,B) < DEP(Z,B)$). We conclude that AB will never be included into the tree $CL(\mathbf{Tr}, \mathbf{X}' \cup \{Z\})$. ∎

The Proposition 3 demonstrated that IT behaves rationally upon inclusion of a new node: the d-separation property (for definition of d-separations see e.g. [8]) is imposed. The next Proposition shows that this property is kept at next inclusions of IT.

PROPOSITION 4. *Let* $\mathbf{Tr}(\mathbf{X},\mathbf{E})$ *be a tree. Let* $CL(\mathbf{Tr},\mathbf{X}')$ *be obtained for* $\mathbf{X}' \subseteq \mathbf{X}$. *Let* Z *be a node from* $\mathbf{X}\text{-}\mathbf{X}'$. *Let* A, B *be in* $\mathbf{X}'$. *Let there be no edge* AB *in* $CL(\mathbf{Tr},\mathbf{X}')$ *and let* C *be a node in* $\mathbf{X}'$ *such that* C *is on the path between* A *and* B *in* $\mathbf{Tr}$. *Then* AB *cannot be in* $CL(\mathbf{Tr},\mathbf{X}' \cup \{Z\})$.

Proof. Notice that

$$DEP(A,B) < DEP(C,A) \quad \text{and} \quad DEP(A,B) < DEP(C,B).$$

Let us consider the construction sequence $\mathrm{CLS}(\mathbf{Tr},\mathbf{X}' \cup \{Z\})$, in particular the point when one of the two nodes A, B is included into the tree grown so far, and the other isn't. Assume A is already included (the case B is symmetric). Now either C is already included or is not. Assume first that C is included. Then the edge AB cannot be included because AC is a competing candidate with greater DEP ($DEP(A,B) < DEP(C,A)$). Now assume C is already included. Then the edge AB cannot be included because BC is a competing candidate with greater DEP ($DEP(A,B) < DEP(C,B)$). We conclude that AB will never be included into the tree $\mathrm{CL}(\mathbf{Tr}, \mathbf{X}' \cup \{Z\})$. ∎

The next Proposition shows that IT establishes true edges as soon as the ends of the edges are available.

PROPOSITION 5. *Let* $\mathbf{Tr}(\mathbf{X},\mathbf{E})$ *be a tree. Let* $CL(\mathbf{Tr},\mathbf{X}')$ *be obtained for* $\mathbf{X}' \subseteq \mathbf{X}$. *Let* Z *be a node from* $\mathbf{X}\text{-}\mathbf{X}'$. *Let* A *be in* $\mathbf{X}'$. *Let* Z *be a direct neighbor of* A *in* $\mathbf{Tr}$. *Then* AZ *will be an edge in* $CL(\mathbf{Tr},\mathbf{X}' \cup \{Z\})$.

Proof. In $\mathbf{Tr}$ removal of AZ would split $\mathbf{Tr}$ into two parts: Pa containing A and Pz containing Z. As the preceding Propositions indicate, no node from $X' \cap Pa$ can be connected with any node from $\mathbf{X}' \cap Pz$ except for nodes A and Z. But $\mathbf{X}' = (\mathbf{X}' \cap Pa) \cup (\mathbf{X}' \cap Pz)$ and in order to obtain a tree containing all nodes, AZ must be included. ∎

The next Proposition shows that IT keeps true edges as soon as it has established them.

PROPOSITION 6. *Let* $\mathbf{Tr}(\mathbf{X},\mathbf{E})$ *be a tree. Let* $CL(\mathbf{Tr},\mathbf{X}')$ *be obtained for* $\mathbf{X}' \subseteq \mathbf{X}$. *Let* Z *be a node from* $\mathbf{X}\text{-}\mathbf{X}'$. *Let* A, C *be in* $\mathbf{X}'$ *and let* C *be a direct neighbor of* A *in* $\mathbf{Tr}$. *Then* AC *will be an edge in* $CL(\mathbf{Tr},\mathbf{X}' \cup \{Z\})$.

Proof. In $\mathbf{Tr}$ removal of AC would split $\mathbf{Tr}$ into two parts: Pa containing A and Pc containing C. As the preceding Propositions indicate, no node from $\mathbf{X}' \cap Pa$ can be connected with any node from $\mathbf{X}' \cap Pc$ except for nodes A and C. But $\mathbf{X}' = (\mathbf{X}' \cap Pa) \cup (\mathbf{X}' \cap Pc)$ and in order to obtain a tree containing all nodes, AZ must be included. ∎

Now we come to the fundamental theorems of this section that show correctness of the IT algorithm, that is that IT recovers the intrinsic tree of

the tree-like Bayesian network underlying the probability distribution. Theorem 1 shows the correctness of an algorithm with modified DEP measures and then Theorem 2 shows that IT is identical with this algorithm with modified DEP measures.

THEOREM 1. *Consider the sequence of sets* $\mathbf{X}_2 \subset \mathbf{X}_3 \subset \mathbf{X}_{n-1} \subset \mathbf{X}_n = \mathbf{X}$ *with* $\mathbf{X}_i = \{X_1, X_2, \ldots, X_i\}$ *for* $i = 2, \ldots, n$. *Consider a modified Chow/Liu algorithm CL' such that CL'(*$\mathbf{Tr}$, $\mathbf{X}_2$*) is identical with CL'(*$\mathbf{Tr}$, $\mathbf{X}_2$*), and for* $i = 3, .., n$ *CL'(*$\mathbf{Tr}$, $\mathbf{X}_i$*) is identical with CL(*$\mathbf{Tr}$, $\mathbf{X}_i$*) except that the dependence* $DEP(A, B)$ *is zero if* AB *is not an edge in CL'(*$\mathbf{Tr}$, $\mathbf{X}_{i-1}$*) and neither* A *nor* B *is identical with* X_i. *Then CL'(*$\mathbf{Tr}$, $\mathbf{X}$*) yields identical result with CL(*$\mathbf{Tr}$, $\mathbf{X}$*).*

Proof. We have to demonstrate the following: If two nodes X_j, X_k with $j < k$ are direct neighbors, then they will be connected in CL($\mathbf{Tr}$, $\mathbf{X}_k$), hence also in CL'($\mathbf{Tr}$, $\mathbf{X}_k$) because the dependence $DEP(X_j, X_k)$ will be identical in CL($\mathbf{Tr}$, $\mathbf{X}_k$) and CL'($\mathbf{Tr}$, $\mathbf{X}_k$), and all the other dependences (also those competing with connection X_jX_k) will have values in CL'($\mathbf{Tr}$, $\mathbf{X}_k$) not greater than in CL($\mathbf{Tr}$, $\mathbf{X}_k$). Later on, for $l > k$, by induction we can show that in $DEP(X_j, X_k)$ will be identical in CL($\mathbf{Tr}$, $\mathbf{X}_l$) and CL'($\mathbf{Tr}$, $\mathbf{X}_l$), and all the other dependences (also those competing with connection X_jX_k) will have values in CL'($\mathbf{Tr}$, $\mathbf{X}_l$) not greater than in CL($\mathbf{Tr}$, $\mathbf{X}_l$). This is because X_jX_k has been included in CL'($\mathbf{Tr}$, $\mathbf{X}_{l-1}$) and thus will be included in CL'($\mathbf{Tr}$, $\mathbf{X}_l$).

Hence any edge contained in $\mathbf{Tr}$ will also appear in CL'($\mathbf{Tr}$, $\mathbf{X}$). As CL'($\mathbf{Tr}$, $\mathbf{X}$) is a tree over the same set of nodes as $\mathbf{Tr}$, it must be identical with $\mathbf{Tr}$. ∎

THEOREM 2. *The algorithm IT yields the same tree as CL.*

Proof. The previous theorem describes an algorithm CL' which is essentially identical with IT because IT simply ignores edges not present in the tree built in the previous stage and this is identical with setting respective dependence to zero because the dependences DEP are non-negative by definition. So IT yields the same as CL because CL' does as claimed in Theorem 1. ∎

Notice also that

PROPOSITION 7. *Let* $\mathbf{Tr}$ *be the tree of the underlying distribution, and* AB *an edge in it. Let Pa and Pb be sub-trees obtained from* $\mathbf{Tr}$ *upon removal of* AB *(Pa containing* A*, Pb containing* B*). Then for any node* X *in Pa* $DEP(X, B) < DEP(A, B)$ *and for any node* Y *from Pb* $DEP(A, Y) < DEP(A, B)$.

Proof. We need only to consider the case of node X, as the case with Y is symmetric. Obviously A is on the path from X to B. Hence it is a straight forward conclusion that $DEP(X,B) < DEP(A,B)$. ∎

5. Beyond the underlying tree-like distributions

The Chow/Liu algorithm is known to reconstruct robustly the Bayesian network underlying a probability distribution given the network is tree-like [4, 5]. However, it can be applied to sets of variables of any probability distribution yielding then a best approximating tree. We would like to be able to use the IT algorithm also in this context. The Chow/Liu algorithm is known for its optimal behavior in this context. Does IT behave that good also? To answer this question let us study some properties of Chow/Liu algorithm (CL) in the context of a general type distribution.

By CL(**D**,**X**) let us denote the tree yielded by CL, and by CLS(**D**,**X**) the construction sequence (the sequence in which edges are added) for an underlying distribution D.

Propositions 8–10 establish a brand of rationality behind IT in that for any node the DEP to its direct neighbor is higher than to the nodes "behind" the neighbor.

PROPOSITION 8. *Let AB be any edge in the tree CL(**D**,**X**). Let Pa and Pb be two trees we obtain from CL(**D**,**X**) after removing the edge AB, with Pa containing A and Pb containing B. Let A be included into CLS(**D**,**X**) before B has been included. Then $DEP(A,B) \geq DEP(A,V)$ for any V in Pb.*

Proof. At the moment, that CL(**D**,**X**) was deciding to include B, only a subset **S** of nodes from Pa (including A) was in the current **T** tree and none of the nodes from Pb. That is all nodes from Pb were in **N**. The nature of the step 4 of CL algorithm implies that $DEP(A,B) \geq DEP(A,V)$ for any node V from **N**, and thus for all nodes from Pb. ∎

PROPOSITION 9. *Let AB be any edge in the tree CL(**D**,**X**). Let Pa and Pb be two trees we obtain from CL(**D**,**X**) after removing the edge AB, with Pa containing A and Pb containing B. Let A be included into CLS(**D**,**X**) before B has been included. Then $DEP(A,B) \geq DEP(V,B)$ for any V in Pa.*

Proof. At the moment, that CL(**D**,**X**) was deciding to include B, only a subset **S** of nodes from Pa (including A) was in the current **T** tree and none of the nodes from Pb. The nature of the step 4 implies that $DEP(A,B) \geq DEP(Y,B)$ for any node Y from **S**. It also implies that for any node Y from **S** and for any node Z from Pa but not from **S**, $DEP(A,B) \geq DEP(Y,Z)$.

Now consider the inclusion of the first node Z_1 from Pa but not from **S**. It is attached to a node Y_1 from **S**. Obviously $DEP(Y_1,Z1) \geq DEP(Z1,B)$.

But we had $DEP(A,B) \geq DEP(Y_1,Z1)$, hence also $DEP(A,B) \geq DEP(Z1,B)$. But we had also that for any Z from Pa but not from **S** different from Z_1 $DEP(Y_1,Z1) \geq DEP(Z,B)$ and due to $DEP(A,B) \geq DEP(Y_1,Z1)$, we have also $DEP(A,B) \geq DEP(Z,B)$. Hence $DEP(A,B) \geq DEP(V,B)$ for any V from Pa. ∎

PROPOSITION 10. *Let AB be any edge in the tree CL(**D**,**X**). Let Pa and Pb be two trees we obtain from CL(**D**,**X**) after removing the edge AB, with Pa containing A and Pb containing B. Then $DEP(A,B) \geq DEP(A,V)$ for any V in Pb.*

Proof. The conclusion of this Proposition follows directly from Propositions 8 and 9. ∎

The following states that the IT algorithm keeps edges removed in the tree construction process.

PROPOSITION 11. *Let A,B be not connected in the tree CL(**D**,**X**'). Let Z be a node from **X** but not from **X**'. Then A,B will not be connected in CL(**D**,**X**$' \cup \{Z\}$).*

Proof. If A and B are not connected in CL(**D**,**X**'), then there is a path $A - A_n - \ldots - A_3 - A_2 - A_1 - D - B_1 - B_2 - \ldots - B_m - B$ connecting them where D is the "oldest" node on the path that is the one that was included as the first among the nodes mentioned. Let the nodes A_1 to A_r be the next ones included. Let then the node B_1 be included. By the property of the algorithm, $DEP(D,B_1)$ nust be smaller than $DEP(D,A_1)$, $DEP(A_1,A2)$, ..., $DEP(A_k,A_{k-1})$. Let then nodes B_2 to B_r be included. And then the node A_{k+1}. Obviously, $DEP(A_{k+1},A_k)$ must be smaller than $DEP(D,B_1)$, $DEP(B_1,B_2)$ ..., and hence also smaller than $DEP(D,A_1)$, $DEP(A_1,A_2)$.... Thus we can demonstrate that $DEP(A,B)$ is smaller than any dependence of any two neighboring nodes on the path $A - A_n - \ldots - A_3 - A_2 - A_1 - D - B_1 - B_2 - \ldots - B_m - B$.

Assume now that in the tree construction sequence CLS(**D**,**X**$' \cup \{Z\}$) A was included before B. (if B is included first, the argument is similar). Assume A_n is not included. Then AB cannot be included because AA_n has a higher DEP. After A_n is included but A_{n-1} not. AB cannot be included because A_nA_{n-1} has a higher DEP. Etc. When Bm is included and B not, then AB cannot be included because B_mB has a higher DEP. ∎

The previous Proposition permits to conclude that the rationality established in Proposition 10 may be extended one step more.

PROPOSITION 12. *Let AB be any edge in the tree CL(**D**,**X**). Let Pa and Pb be two trees we obtain from CL(**D**,**X**) after removing the edge AB, with Pa*

containing A and Pb containing B. Then $DEP(A,B) \geq DEP(W,V)$ *for any V in Pa and any W in Pb.*

Proof. The conclusion of this Proposition is provable analogously to that of Proposition 11. ∎

Now let us present the crucial theorem of this paper stating that the results of IT are identical to those of CL.

THEOREM 3. *The algorithm IT yields the same tree as CL for arbitrary distribution.*

Proof. The previous Propositions show that in an incremental process of building trees for larger and larger sets, once a pair of nodes went apart, it will never be considered for merging. Hence their dependence does not need to be considered. So IT yields the same as CL. ∎

6. Experiments

To verify the Propositions raised in this paper, experimental implementations of CL1 and IT were tested on identical artificial data sets generated from tree-like Bayesian networks with binary variables. Networks with 100 up to 2,000 nodes were considered. Conditional probabilities of success on success and failure on failure of the variables were varied from 0.6 to 0.9. Branching factors of the underlying trees were chosen in the range from 2 to 8. Sample sizes ranged from the number of variables to the tenfold of the number of variables. The sequence of variable inclusions was randomized.

The experiments confirmed the otherwise known high robustness of the Chow/Liu algorithm: The number of errors in inserting edges rarely was reaching 1 %. (In over 90 % of all experiments no errors occurred at all).

The IT algorithm proposed in this paper behaved exactly in the same way as the Chow/Liu algorithm (perfect reconstruction of the original tree-like Bayesian network with Chow/Liu was paralleled by perfect reconstruction when using IT, also all errors of Chow/Liu were followed by IT).

The IT algorithm exhibited consistently a slight execution time advantage over CL1 algorithm. This may be attributed to the fact that IT requires much less memory and probably it avoids therefore some additional calculations in memory management.

7. Conclusions

This study has demonstrated the possibility of reducing space consumption when constructing tree like Bayesian network from data from quadratic in the number of variables by the Chow/Liu algorithm to a linear one without worsening the time efficiency. A new algorithm achieving this goal has

been proposed and it was demonstrated that the resulting tree will have properties no worse than the one delivered by the Chow/Liu algorithm.

Out of this fact new application possibilities are open. Bayesian network construction for applications with 10,000 and more nodes like those needed in free text classifications will be possible. The new approach is independent of other approaches to improvements of efficiency of the Chow/Liu algorithm. For example the sparse data time saving algorithms proposed in [11, 12, 13] may still be applied in the context of the new algorithm.

The success in reducing considerably space consumption without worsening time complexity may be considered as an encouragement for further research into possibilities of time saving without increase of space complexity for Chow/Liu like algorithms.

References

[1] J. Cerquides, *Applying General Bayesian Techniques to Improve TAN Induction*, Knowledge Discovery and Data Mining, 1999, pp. 292–296, http://citeseer.nj.nec.com/cerquides99applying.html.

[2] J. Cheng, D. A. Bell, W. Liu, *An algorithm for Bayesian belief network construction from data*, Proceedings of AI & STAT'97, Ft. Lauderdale, Florida, 1997.

[3] J. Cheng, D. A. Bell, W. Liu, *Learning belief networks from data: an information theory based approach*, Proceedings of the Sixth ACM International Conference on Information and Knowledge Management, 1997.

[4] C. K. Chow, C. N. Liu, *Approximating discrete probability distributions with dependence trees*, IEEE Transactions on Information Theory, IT-14, No. 3, 1968, pp. 462–467.

[5] C. K. Chou, T. J. Wagner, *Consistency of an estimate of tree-dependent probability distribution*, IEEE Transactions on Information Theory, IT-19, 1973, 369–371.

[6] N. Friedman, D. Geiger, M. Goldszmidt, *Bayesian Network Classifiers*, Machine Learning vol. 29, 1997, pp. 131. http://citeseer.nj.nec.com/friedman97bayesian.html

[7] N. Inza, M. Merino, P. Larranaga, J. Quiroga, B. Sierra, M. Girala, *Feature Subset Selection by Genetic Algorithms and Estimation of Distribution Algorithms. A Case Study in the Survival of Cirrhotic Patients Treated with TIPS.*, Artificial Intelligence in Medicine (in press). http://citeseer.nj.nec.com/345995.html

[8] M. A. Kłopotek, S. T. Wierzchoń, M. Michalewicz, M. Bednarczyk, W. Pawłowski, A. Wąsowski, *Bayesian Network Mining System*, Proc. X International Symposium on Intelligent Information Systems, Zakopane, 18–22 June, 2001, Springer-Verlag, New York 2001. ISBN-3-7908-1407-5. pp. 97–110.

[9] M. A. Kłopotek, *On a Deficiency of the FCI Algorithm Learning Bayesian Networks from Data*, Demonstratio Math. 33 (2000), 181–194.

[10] M. A. Kłopotek, *Methods of Identification and Interpretations of Belief Distributions in the Dempster-Shafer Theory*, (in Polish). Publisher: Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland, 1998, ISBN 83-900820-8-x.

[11] M. Meila, *An Accelerated Chow and Liu Algorithm: Fitting Tree Distributions to High-Dimensional Sparse Data*, http://citeseer. nj.nec.com/285522.html

[12] M. Meila, *An accelerated Chow and Liu algorithm: fitting tree distributions to high-dimensional sparse data*, http://citeseer. nj.nec.com/363584.html

[13] M. Meila, M. Jordan, *Learning with mixtures of trees*, J. Machine Learning Research 1 (2000), http://citeseer.nj.nec.com/ meila00learning.html, http://www.ai.mit.edu/projects/jmlr/papers/ volume1/meila00a/html/meila00a.html

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

[15] J. Suzuki, *Learning Bayesian Belief Networks based on the Minimum Descripion Length Principle: Basic Properties*, IEICE Trans. on Foundamentals, Vol. E82-A, Oct. 1999, and the conference paper in UAI-93.
http://www.math.sci.osaka-u.ac.jp/ suzuki/net.html

[16] R. S. Valiveti, B. J. Oommen, *The use of chi-squared statistics in determining dependence tree*, Technical Report SCS-TR-153, School of Computer Science,Carleton University, Ottawa, Ontario, March 1989.

[17] R. S. Valiveti, B. J. Oommen, *On using the chi-squared statistics for determining statistic dependence*, Pattern Recognition Vol. 25 No. 11 (1992), 1389–1400.

INSTITUTE OF COMPUTER SCIENCE
POLISH ACADEMY OF SCIENCES
ul. Ordona 21
01-237 WARSZAWA, POLAND
E-mail: klopotek@ipipan.waw.pl

FACULTY OF MATHEMATICS AND INFORMATION SCIENCE
WARSAW UNIVERSITY OF TECHNOLOGY
Plac Politechniki 1
00-661 WARSZAWA, POLAND

Received May 16, 2001; revised version February 17, 2001.