Bayesian Belief Networks for Data Mining

Harald Steck [*]**and Volker Tresp**
Siemens AG, Corporate Technology
Information and Communications
81730 Munich, Germany
{Harald.Steck, Volker.Tresp}@mchp.siemens.de

Abstract

In this paper we present a novel constraint based structural learning algorithm for causal networks. A set of conditional independence and dependence statements (CIDS) is derived from the data which describes the relationships among the variables. Although we implicitly assume that there exists a perfect map for the true, yet unknown, distribution, there does not need to be a perfect map for the CIDSs derived from the limited data. The reason is that the distribution of limited data might differ from the true probability distribution due to sampling noise. We derive a necessary condition for the existence of a perfect map given a set of CIDSs and utilize it to check for inconsistencies. If an inconsistency is detected, the algorithm finds all Bayesian networks with a minimum number of edges such that a maximum number of CIDSs is represented in each of the multiple solutions. The advantages of our approach are illustrated using the alarm network data set.

1 INTRODUCTION

A Bayesian belief network represents conditional independences in the underlying probability distribution of the data in the form of a directed acyclic graph (DAG). The DAG: $A \to B \leftarrow C$ states, for example, that A and C are independent if B is not known, but A and C become dependent as soon as the state of B is fixed. If all conditional independences in the probability distribution are represented in the DAG and vice versa, then the Bayesian network is said to be a perfect map of the probability distribution, and is called a causal network [Pearl 1988]. Typically a Bayesian network is constructed from expert knowledge although recently there has been an increasing interest in learning the structure of Bayesian networks from data for data mining applications. The basic idea is to display probabilistic dependences and independences derived from the data in a concise way by a Bayesian network which has the potential of providing much more information about a domain than visualizations solely based on correlations and distance measures. Bayesian networks can be used to display causal models which can be understood by humans more intuitively than models with undirected edges like Markov networks.

Model uncertainty is a serious problem in structural learning of Bayesian network models and comes into play if there does not exist a perfect map of the probability distribution of the data. If a *limited* data set is given, its probability distribution might differ from the *true* one due to sampling noise. For structural learning we assume that the true probability distribution is such that there exists a perfect map.

It is important to present to the user truefully the structural uncertainties since otherwise he or she might draw incorrect conclusions from the presented structure. In a Bayesian approach, structural uncertainty can be represented to the user by presenting multiple solutions which have obtained a high score. The disadvantage is that one can never be sure that one has found all solutions with a high score. Furthermore, it is very difficult for a user to study multiple solutions visually and to draw meaningful conclusions. In addition, the computationally cost of finding the K-best solutions is even higher than for finding only the network with the best score. Therefore we have focused on an approximate way of structural learning without losing model uncertainty in the result.

In the next section, we give a short summary of structural learning. Then we present a proposition which serves as a necessary condition for the existence of a perfect map given a set of conditional independence

[*]also with Technical University of Munich, Department of Computer Science, 80290 Munich, Germany

and dependence statements (CIDS). This proposition will be used to check for inconsistencies among the estimated CIDSs which then might lead to multiple solutions. Then we present the algorithm utilizing this proposition when checking the CIDSs for consistency and comment on its complexity. We close with results of the computer experiments.

2 STRUCTURAL LEARNING

There are two main approaches to structural learning of Bayesian belief networks. In the Bayesian approach [Cooper and Herskovits 1992, Heckerman *et al.* 1994, Heckerman 1995] a *global* cost function – the posterior probability of the (entire) network – is maximized. This approach requires an involved search for the best structure and is therefore computationally very expensive and not directly applicable to data mining applications.

In this paper we pursue a constraint based approach similar to the ones in [Wermuth and Lauritzen 1983, Fung and Crawford 1990, Spirtes *et al.* 1993, Suzuki 1996, Cheng *et al.* 1997]. The basic idea of those algorithms is to derive a set of CIDSs from the data without taking into account the Bayesian network structure. The Bayesian network is then constructed from the CIDSs in a later step. This is done in standard algorithms by removing an edge in the Bayesian network whenever the corresponding pair of variables is found conditional independent. There can, however, occur inconsistencies among the CIDSs derived from *limited* data when reconstructing the entire network (up to equivalence), i.e. not all CIDSs can be represented in a perfect map simultaneously. These inconsistencies reflect model uncertainty, i.e. there might exist more than one Bayesian network model describing the probability distribution of the data well (according to some measure). This indicates how reliable one particular structure learned from the database is.

These multiple solutions have usually many edges in common and differ in the presence of a few edges only which we call *inconsistent edges*, since they are related to inconsistencies among the CIDSs. The set of inconsistent edges can further be divided into subsets such that the presence or absence of edges belonging to different subsets is independent of each other in the multiple solutions. Such a subset we call an *ambiguous region*. Hence, we visualize the set of solutions in a single graph (cf. Figure 3), in which the edges common to all networks are sketched by solid lines, whereas the edges belonging to the same ambiguous region are depicted by dashed lines of the same style. The possible structures in each of the ambiguous regions cannot be depicted in such a graph. They are hence stated in the figure caption. The directions of the edges are specified in a later stage by this kind of constraint based approach.

These possibly multiple solutions for the network structures do not belong to the same equivalence class, since they differ in the presence or absence of certain edges. So the multiple solutions have to be distinguished from networks belonging to the same equivalence class.

The advantage of our constraint based approach is that it can systematically construct the set of all Bayesian networks. Furthermore, it can take advantage of the fact that the multiple solutions typically have many edges in common and differ only in a few *inconsistent* edges which can be partitioned into independent *ambiguous regions.*

Regarding the derivation of the CIDSs from the data set, the computation time of this approach relative to the global Bayesian approach is appreciably shorter for sparse network structures, i.e. when the probability distribution exhibits many conditional independences. For dense networks, this may not hold, but Bayesian network models might not be the most suitable model in such a situation, anyway. The computation time can additionally be reduced by carrying it out in parallel. This can be realized in a simple master and slave scheme.

3 NECESSARY PATH CONDITION

The algorithm we depict in this paper makes use of the following proposition as a necessary condition for the existence of a perfect map given a set of CIDSs derived from the data. Essentially, the proposition requires that if two variables[1] a, b only become independent by conditioning on an additional variable c, then there must be a path connecting them via c in the graph. Otherwise the dependence without conditioning on c would be unexplained. If there were no necessary condition at all, then one would arrive at the SGS or PC algorithm [Spirtes *et al.* 1993].

Proposition: *Let V be the set of variables and P a probability distribution with the perfect map G: If two variables $a, b \in V$ are conditionally independent in P given a set of variables $S \subseteq V \backslash \{a, b\}$ and they are dependent given any (proper) subset $S' \subset S$, then there is no edge between a and b in G and there exist (undirected) paths between each $s \in S$ and a not crossing b as well as between each $s \in S$ and b not crossing a.*

[1] For brevity, we use *nodes* and *variables* synonymously and denote both with small letters, since there is a one-to-one correspondence.

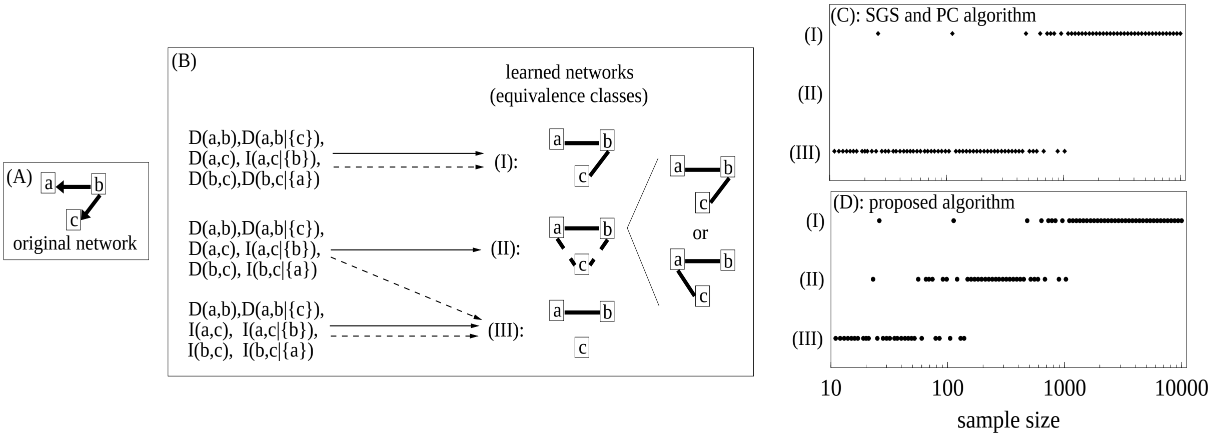

Figure 1: From the original network (A) we randomly generate data sets of various sizes (sample size between 10 and 10000). The distribution of the variables is given by $b = \varepsilon_b$, $a = m_{ab}b + \varepsilon_a$ and $c = m_{cb}b + \varepsilon_c$ with the parameters $m_{ab} = 3$, $m_{cb} = 0.3$ and with Gaussian distributed noise ε_i for $i \in \{a, b, c\}$ of unit variance and zero mean. For comparison with the SGS and PC algorithms we used the test on vanishing partial correlations with a significance level of 0.01 in this example in order to derive the conditional independence and dependence statements (CIDSs) from the data sets. For details see section (3).

Proof: This proposition follows immediately from the definitions of the perfect map and of the d-separation (see for instance [Pearl 1988]). Given two variables a and b and a set $S \subseteq V\backslash\{a, b\}$, assume that there is a variable $v \in S$ such that there is no (undirected) path between a and v not crossing b in the perfect map G. Then, if a and b are d-separated given S, they are also d-separated given $S\backslash\{v\}$. Hence, considering the probability distribution P, if a and b are independent conditional on S, then they are also independent conditional on $S\backslash\{v\}$. *QED.*

Necessary Path Condition:

In general one cannot expect that an arbitrary set of CIDSs can be represented in a perfect map, since a perfect map implies certain relations among the CIDSs to hold. Therefore it is desirable to have necessary and sufficient conditions at hand for efficiently checking the CIDSs for consistency, i.e. if all the CIDSs can simultaneously be displayed in one perfect map.

Since we like to retain the property of the learning algorithm to sequentially learn the skeleton of the Bayesian network, i.e. first the presence of its edges, then the orientations of the edges and finally the parameters of the model, a sufficient condition for the existence of a perfect map when constructing the skeleton cannot be available. The above proposition provides, however, a necessary condition for the existence of a perfect map which means that it can be used to check for inconsistencies among the CIDS in the sense that if an inconsistency is found there cannot exist a perfect map representing all the CIDSs.

In networks involving a large number of variables, there might be several estimated conditional independence statements (CIS) for each pair of variables a, b. Therefore we propose the following *Necessary Path Condition* to be used to check for inconsistencies in the algorithm: For each absent edge $[a, b]$ in the network there has to be represented at least one – not all – CIS $I(a, b|S)$ – with the corresponding $D(a, b|S')$ for all $S' \subset S$ – in the perfect map according to the above proposition. This is less strict than the proposition itself, but makes the algorithm more robust, since an inconsistency is only detected if there is an edges for which no CIDSs can be found consistent.

Example (cf. Figure 1):

In Figure 1 it is shown in a toy model of three variables a, b and c that it is essential to check the set of CIDSs derived from *limited* data on *consistency* with the Bayesian network model, since each conditional independence statement (CIS) or conditional dependence statement (CDS) is derived from the data without taking into account the model at all. In (B) we compare the resulting networks constructed from (possibly inconsistent) sets of CIDSs by two different algorithms: First, the SGS and PC algorithms (dashed arrows) which assumes an edge absent in the model whenever a conditional independence is derived from the data without checking for consistency. Second, the proposed algorithm (solid arrows) checking for consistency.

In the first scenario (cf. network (I) in (B)), assume that the only conditional independence statement derived from the data is $I(a,c|\{b\})$ (together with the (conditional) dependence statements $D(a,b)$, $D(a,c)$, $D(b,c)$, $D(a,b|\{c\})$, $D(b,c|\{a\})$). According to the proposition this requires that there must exist a path between a and b as well as between b and c, whereas the edge $[a,c]^2$ must be absent. Apparently, there exists a perfect map, namely comprising the edges $[a,b]$ and $[b,c]$, i.e. the set of CIDSs is consistent.

In the second scenario (cf. networks (II) in (B)), assume the two CISs $I(a,c|\{b\})$ and $I(b,c|\{a\})$ – and the CDSs $D(a,b)$, $D(a,c)$, $D(b,c)$, $D(a,b|\{c\})$ – are found from the data. The edge $[a,b]$ is required by these CIDSs, but the CIS $I(a,c|\{b\})$ together with $D(a,c)$ additionally requires the presence of the edge $[b,c]$ and the absence of the edge $[a,c]$, whereas the other CIS requires just the contrary. Hence, there cannot exist a perfect map fulfilling all the CIDSs simultaneously. For example, the set of CIDSs corresponding to a perfect map containing only the edge $[a,b]$ (as found by the SGS or PC algorithm) comprises $I(a,c)$ and $I(b,c)$ which violates both $D(a,c)$ and $D(b,c)$. If one assumes now that the inconsistencies among the CIDSs are solely due to sampling noise in the *limited* data set, it makes sense to search for all possible Bayesian networks with a minimum number of edges each of which constructed from a maximal *consistent* subset of the CIDSs derived from the data. In this example, it is apparent that exactly one of the two CISs cannot be represented in a perfect map. Consequently, one finds two possible network structures: the edge $[a,b]$ is common to both, and they contain edge $[a,c]$ or $[b,c]$, respectively. Neither of the two networks represents all the CISs which were derived from the data. Note that the correct causal network, i.e. the perfect map of the – in general unknown – *true* set of CIDSs is among the multiple solutions in this example.

Finally, the set of CIDSs comprising the unconditional CISs $I(a,c)$ and $I(b,c)$ are consistent and lead to a network with only the edge $[a,b]$ being present (cf. network (III) in (B)), i.e. the original network structure could not be recovered from these CIDSs which tend to be derived from small data sets.

In (C) and (D) the resulting network structures depending on the sample size is shown. While our approach yields multiple solutions (cf. (II) in (D)) for "medium sized" data sets (sample size between ca. 70 and ca. 700), the SGS or PC algorithms find the network comprising the single edge $[a,b]$ (cf. (C)).

4 ALGORITHM

Before we go into details of the proposed algorithm, we give a short overview of its main steps. After the set of conditional independence and dependence statements (CIDS) has been derived from the data the proposed algorithm applies the Necessary Path Condition presented above in order to check, if a perfect map can exist given the set of CIDSs. If inconsistencies are detected, the algorithm searches for all possible Bayesian networks as described in detail in section 4.3. For each of the multiple solutions, the directions of the edges are fixed in the last step. All these networks contain the same number of edges. Among those are all the edge resulting from the the SGS algorithm[3]. Each of the minimal solutions found by the algorithm represents an equivalence class, and is a candidate for being a perfect map of the (unknown) *true* probability distribution of the data set.

4.1 DERIVING THE CIDSs

The algorithm builds up the set of conditional independence and dependence statements from a given data set in the first step. This can efficiently be done by relying on asymptotic results which appear reasonably accurate in our experiments, for example by calculating the Bayesian Information Criterion or a test statistic for each pair of variables given a subset of the remaining variables.

Considering independence tests it is apparent that CDSs derived from the data are more reliable than the CISs. This is because the null hypothesis of independence can only by falsified but not verified by a test. This indicates that network structures learned by a constraint based approach tend to remove too many edges, in particular given small data sets. As will be shown below, this tendency of removing too many edges can largely be reduced by checking the CIDSs for consistency.

For convenience we do not denote the conditional *dependence* statements, i.e. if a statement is not in the set of CISs (and cannot be inferred from it by combining some of the CISs according to the Faithfulness and Markov Conditions as described

[2] In this notation for undirected edges, the order of the variables does not matter, i.e. $[a,b]=[b,a]$.

[3] In this paper, we focus on how to construct the skeleton of a Bayesian network from the CIDSs rather than how to derive the CIDSs themselves. The CIDS derived by the SGS and PC algorithms can differ, since the latter facilitates heuristics. Here, we compare our algorithm with the SGS algorithm, because the experiments carried out here are based on the CIDSs derived by the SGS algorithm. Of course, our algorithm can also be applied to the CIDSs derived by the PC algorithm, and the same statements apply as for the SGS algorithm.

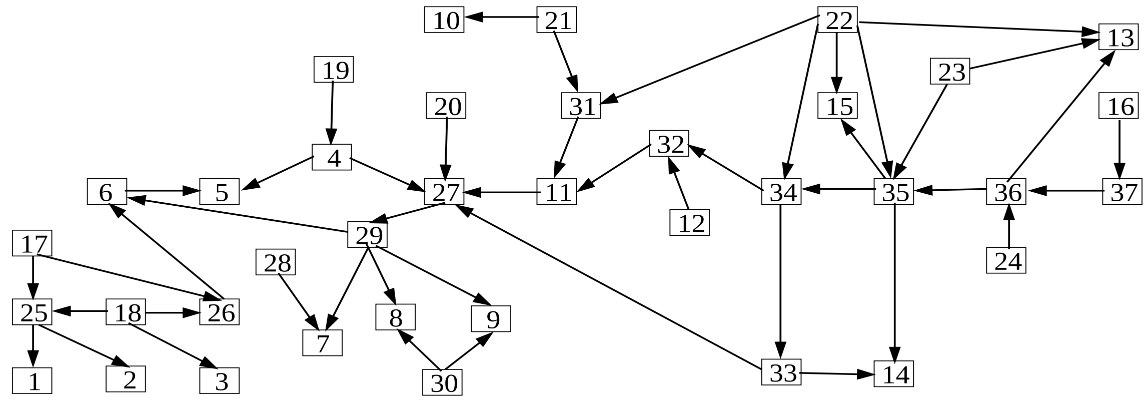

Figure 2: The alarm network contains 37 variables and 46 edges. The numbering of the variables is chosen as in [Cheng *et al.* 1997].

in [Spirtes *et al.* 1993]), it is understood that this implies a (conditional) dependence.

4.2 RULES

For each given CIS of the form $I(a, b|S)$ with $D(a, b|S')$ $\forall S' \subset S$ the proposed Necessary Path Condition requires the absence of the edge $[a, b]$ and the presence of the paths between a and each variable $s \in S$ as well as between b and each variable $s \in S$. We represent this constraint on the absence or presence of certain edges by a *rule*. Such a rule is of the form $X \prec \mathbf{Y}$, where X denotes an edge and $\mathbf{Y}$ is a (possibly empty) set of edges. According to the Necessary Path Condition, the above CIS with the CDSs is translated into a rule as follows: $X = [a, b]$ and $\mathbf{Y} = \bigcup_{s \in S}\{[a, s], [b, s]\}$. It can be interpreted in the way that edge X can only be absent in the Bayesian network, if the edges in the set $\mathbf{Y}$ are present. Since the above proposition requires certain paths rather than certain edges to be present, this is absorbed in the fact that new rules can be generated by substituting rules into each other as follows: Given two rules $X \prec \mathbf{Y}$ and $W \prec \mathbf{Z}$ then a new rule can be generated for edge X, if the edges $W \in \mathbf{Y}$ and $X \notin \mathbf{Z}$, namely $X \prec (\mathbf{Z} \cup \mathbf{Y} \backslash \{W\})$. Therefore an edge can be absent, if there is at least one rule fulfilled (cf. Necessary Path Condition).

Once the set of rules is derived from the set of CIDSs, the associated perfect map can be constructed. If there is a Bayesian network such that for all edges a rule is fulfilled, then there might exist a perfect map associated with the estimated probability distribution. If there are some edges for which none of the given rules can be fulfilled by a Bayesian network, we call the set of rules and the set of CIDSs *inconsistent*. In this case, there does not exist a perfect map associated with the estimated probability distribution.

4.3 MULTIPLE SOLUTIONS

Our algorithm finds multiple solutions, when there does not exist a perfect map of the estimated CIDSs according to the Necessary Path Condition. If inconsistencies among the CIDS are found by the algorithm, it makes sense to consider these inconsistencies to be present due to sampling noise in the limited data set and to retain the assumption that there exists a perfect map associated with the (unknown) set of (*true*) CIDSs. Therefore the algorithm searches for all possible minimum subsets of the set of CIDSs which are consistent in the above sense, i.e. the algorithm searches for all possible network structures with a minimum number of edges such that for a maximum number of edges a rule is fulfilled. Each of these possible networks is a candidate for being the perfect map of the (unknown) set of *true* CIDSs.

It turns out that the edges of the resulting networks can be divided into three main groups: If there is no rule $X \prec \mathbf{Y}$ for an edge X, then no estimated CIS was found, and hence this edge is present in the network. If there can be rendered a rule $X \prec \mathbf{Y}$ for edge X such that $\mathbf{Y}$ is empty or contains only edges which are present in the perfect map, then we call it a *consistent* edge which is absent. An edge for which no such rule can be generated indicates that there does not exist a perfect map given the set of CIDSs. Such edges we call *inconsistent* and they might be present or absent in some of the multiple solutions. The multiple solutions differ only in the presences of the the latter edges.

As we will see from the experiments (in section 5), the inconsistent edges can usually be further subdivided: They can be partitioned into sets of edges whose presences depend on each other, whereas edges belonging

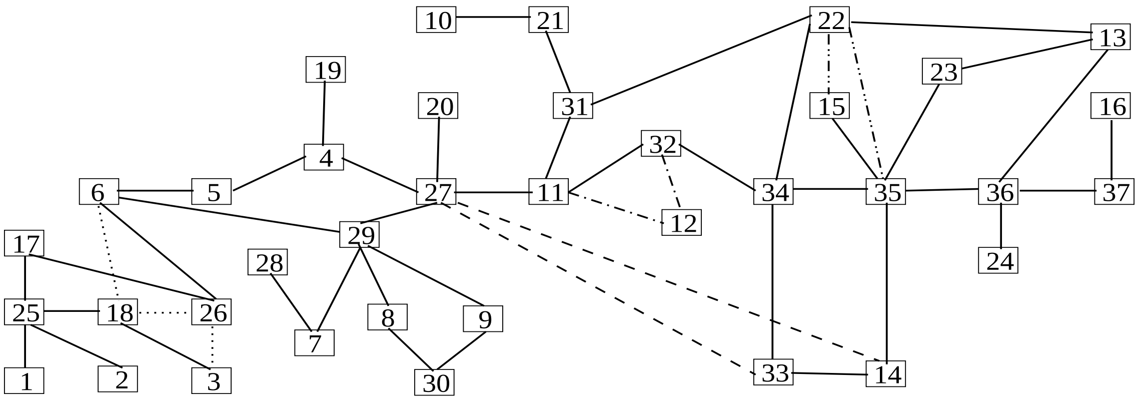

Figure 3: This graph sketches the multiple solutions learned from a data set of size 10000 with the significance level 0.01 before the directions of the edges are added. Solid lines denote edges which are present in all the possible network structures. The multiple solutions differ in the edges belonging to the 4 *ambiguous regions* which are depicted in different line styles. The possible structures for each region are as follows: In region (A) either edge $[11, 12]$ or edge $[12, 32]$ is present. In region (B) either $[14, 27]$ or $[27, 33]$ is present. In region (C) either $[15, 22]$ or $[22, 35]$ is present. In region (D) either the single edge $[18, 26]$ or the two edges $[6, 18]$ and $[3, 26]$ are present; here, the only minimum structure is the one comprising the single edge $[18, 26]$. Hence, there are two minimum structures in each of the regions (A), (B) and (C), and one minimum structure in region (D). Therefore, the overall number of multiple solutions is 8. 41 edges have correctly been identified which are present in all the multiple solutions. Additional 4 edges have been found due to the Necessary Path Condition implemented in our algorithm. The networks among the multiple solutions which are closest to the original one contain 45 (out of 46) correct edges, and only one edge is missing, namely either $[15, 22]$ or $[22, 35]$. The resulting network of the SGS algorithm[3] contains the same 41 edges, which are common to all multiple solutions of our algorithm, so that 5 edges are missing.

to different sets do not depend on each other. Each of such a set we call an *ambiguous region*. There might be several of such regions.

Technically speaking, the algorithm finds all the edges belonging to the same ambiguous region in the following way: Only the rules for inconsistent edges are considered. If for two inconsistent edges X and W there can be rendered rules $X \prec \mathbf{Y}$ and $W \prec \mathbf{Z}$ such that $X \in \mathbf{Z}$ and $W \in \mathbf{Y}$, then they are grouped into the same ambiguous region, because it might not be possible to fulfill both of those rules simultaneously in a Bayesian network. This can also be seen as searching for cycles in a directed acyclic graph (DAG) which is generated from the rules: Each node of that graph represents an inconsistent edge of the Bayesian network, and for each rule $X \prec \mathbf{Y}$ edges are present pointing from each node $Y \in \mathbf{Y}$ to node X in that DAG.

Our algorithm takes advantage of the fact that the multiple solutions differ only in the *ambiguous regions* and that they are independent of each other. Since each of such a region usually contains only a few edges, searching for all possible structures such that the number of consistent CISs is maximum can be done very efficiently: Simply by carrying out an exhaustive search in each ambiguous region separately. The number of different network structures is then given by the product of the number of different structures in each ambiguous region.

4.4 FINDING DIRECTIONS OF EDGES

Constraint based algorithms of this kind have the property that they can be split up into several steps. First, the algorithm finds the (undirected) edges which are present in the Bayesian network. We have focused on that part in this paper. In the second step, directions are added to those edges which can be derived from the data so that the equivalence classes are identified. This can be done like in [Spirtes *et al.* 1993], for example.

The fact that a data set is of limited size might, however, give rise to additional inconsistencies among the estimated CIDSs regarding the directions of the edges. This increases additionally the number of multiple solutions which might differ in the directions of their edges, although they have the same edges in common. We do not present any details on that issue in this paper.

4.5 COMPLEXITY

Calculating the Bayesian information criterion or a test statistic for each pair of variables given every subset of the remaining variables is intractable for large numbers of variables. Since not all of those computations are usually necessary for constructing the Bayesian network from data, the complexity of the problem can be reduced by applying heuristics (see for instance [Spirtes *et al.* 1993]).

Carrying out the computations for all pairs of variables in ascending size of the conditioning set and up to a certain maximum order only, reduces the complexity greatly, i.e. it becomes polynomial in $|V|$. For CISs of high orders which are not computed from the data set it is assumed that exactly those are true which can be inferred from the CISs of lower orders according to the Faithfulness and Markov Conditions as described in [Spirtes *et al.* 1993]. Calculations relying on asymptotic results might yield more unreliable results for higher orders, anyway. Conditioning only on neighbors of the pair of variables (in the undirected graph) is an additional heuristic to speed up the derivation of the CIDSs.

These heuristics require the algorithm of keeping track of the network structure when deriving the CIDSs. After finishing all calculations of a certain order of the conditioning set, a new intermediate version of the network can be built up based on the results available so far. Then, this version can be used to decide, if carrying out computations on higher orders is necessary, and if so, what the neighbors of each node are.

Finding all possible structures of the network from the set of rules is, in principal, intractable for a large number of variables, too. As we found in our experiments, however, there are edges which are present in all the multiple solutions and only a few edges which belong to ambiguous regions. Since the structures in different ambiguous regions are independent of each other, all the possible structures can be found for each ambiguous region separately, which speeds up calculation very much. Hence, the problem is exponential in the number of rules involved in the largest ambiguous region of the network. In the experiments we found that the size of the ambiguous regions is much smaller than the total number of edges. Therefore, finding all the possibilities in each ambiguous region becomes tractable. For example, in our alarm network experiments, it took less than a minute on a Sun UltraSPARC-II to generate all the possible structures given the set of CIDSs.

It turned out in our experiments that almost all the calculation time is consumed for deriving the CIDSs from the data and only a small fraction is needed for constructing all the multiple solutions.

5 EXPERIMENTS

The alarm network [Beinlich *et al.* 1989] has evolved as kind of a benchmark for structural learning of Bayesian belief networks. We used the alarm network from Norsis Corp. [Netica Alarm Network] to generate randomly data sets of various size. The variables are discrete and their number of states ranges from two to four. The size of the sample data was varied between 50000 and 1000, which is much smaller than the number of configurations in the joint state space.

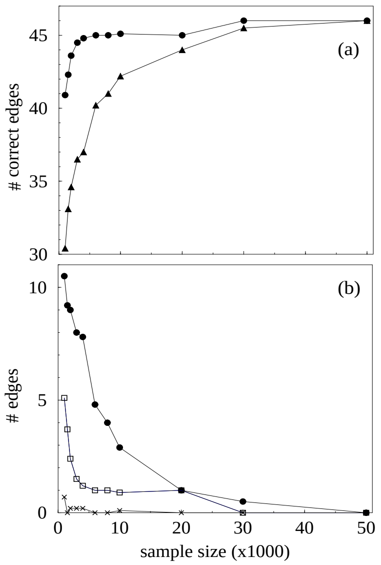

Figure 4: **(a)** The number of correct edges learned from data depends on the sample size. The networks among the multiple solutions which have the most edges in common with the original network contain almost all the correct edges even for small sample sizes (dots). The number of those edges is significantly closer to the correct number of 46 edges than is the number of edges which are common to *all* the learned multiple solutions (triangles). The latter edges are identical with the edges found by the SGS algorithm^3. **(b)** The difference of the two curves in (a) is the number of edges which are simultaneously present in the ambiguous regions of any one of the multiple solutions (dots). The number of edges missing in all the multiple solutions rises for smaller sample sizes (squares), but stays at smaller values than the number of edges being present in the ambiguous regions due to the proposed Necessary Path Condition. In contrast, the *sum* of those edges (dots + squares) is missing in the network resulting from the SGS algorithm^3. The number of erroneously added edges stays at small values (crosses) due to a small significance level of 0.01. Each point represents the mean of 5 experiments of the alarm network.

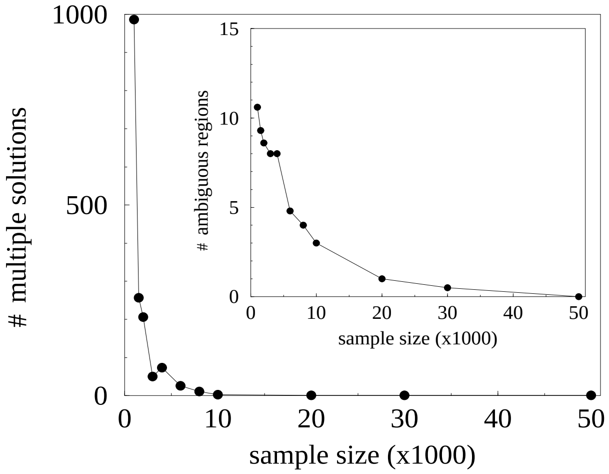

Figure 5: As the size of the data set decreases, the number of multiple solutions increases and so does the number of ambiguous regions (cf. inset). Each point represents the mean of 5 experiments of the alarm network with a significance level of 0.01.

In order to derive the conditional independences, we used the likelihood ratio test with an asymptotic χ^2 distribution as described in [Spirtes *et al.* 1993] for discrete variables.

A typical result of structural learning from a data set of limited size is sketched in Figure 3. For this data set of size 10000 the algorithm detected that there does not exist a perfect map such that all the CIDSs derived from the data can be fulfilled according to the Necessary Path Condition. The multiple solutions differ in the presence of the edges belonging to 4 ambiguous regions, for each of which the possible structures are depicted in the caption. The networks among the multiple solutions of our algorithm which are closest to the original network contain 45 correct edges (out of 46) whereas the SGS algorithm[3] yields a network with only 41 correct edges.

In particular for fairly small data sets the differences in the solutions of those two algorithms increase. In our approach, the number of missing edges grows much more slowly with the sample size decreasing than they do in the resulting network of the SGS algorithm, because the number of edges present in the ambiguous regions increases (cf. Figure 4). The number of edges erroneously present stays particularly small due to the fact that our algorithm allows to use a small significance level when applying conditional independence tests.

The number of multiple solutions depends on the size of the data set (cf. Figure 5). From a frequentist point of view, the *estimated* probability distribution of a larger data set is expected to be more similar to the (unknown) *true* distribution so that the structure of the causal network can uniquely be determined. As the size of the data set shrinks, a unique causal network cannot be derived any more and the number of possible networks of the data increases. Hence, the number of multiple solutions found by the algorithm indicates in a way, if the size of the data set is sufficiently large for learning the network structure with certainty.

As the size of the data set decreases, not only increases the number of multiple solutions, but so does also the number of ambiguous regions, whereas the number of edges involved in each ambiguous region increases only slowly.

6 CONCLUSIONS

Structural learning of causal networks based on this kind of constraint based approach can be split up into several steps which are carried out sequentially. First, it is learned which (undirected) edges are present in the network, then their directions are fixed and eventually the values of the parameters are adapted to the data set. In this paper we focus on the first step, deciding which edges are present and absent in the situation that only a limited amount of data is available.

The proposition presented here serves as a necessary condition for the existence of a perfect map given a set of conditional independence and dependence statements (CIDS). It essentially states that if an edge is absent in the perfect map, certain other paths are required to be present.

The proposed algorithm checks the set of CIDSs on consistency with the Bayesian network model according to the Necessary Path Condition. If inconsistencies are found, it is assumed that they are solely due to sampling noise in the *limited* data set and that there nevertheless exists a perfect map of the true, yet unknown, probability distribution. Therefore, the algorithm searches for all network structures which contain a minimum number of edges and represent a maximum number of consistent CIDSs. This results in multiple solutions.

It turned out in our experiments that all the multiple solutions have many edges in common. There are also some edges (which we call inconsistent edges) in which the structures of the multiple solutions differ from each other. Usually, they can be grouped together in what we call an ambiguous region. In each of which the possibly multiple structures can be found efficiently, since the ambiguous regions are independent of each other and usually involve only a small number of edges. The overall number of multiple solutions is given by the product of the number of different minimum structures in each of the ambiguous regions. We found in the experiments that the number of multiple solutions increases when the size of the data set decreases. Furthermore, also the number of ambiguous regions rises with a decreasing number of data sets.

The multiple solutions of our algorithm contain all the edges which are present in the network found by the SGS algorithm[3] [Spirtes *et al.* 1993] as well as some additional edges, since an estimated CIS does not necessarily lead to the absence of the corresponding edge in the Bayesian network. When the size of the data set decreases, the overall number of correct edges in the resulting networks of our algorithm drops much more gradually than it does in the network found by the SGS algorithm. Depending on the properties (e.g. size) of the data set, we found in our experiments that the number of edges present in the networks learned by our approach can be larger than in the result of the SGS or PC algorithms by $0 - 30\%$.

In Bayesian approaches like [Cooper and Herskovits 1992, Heckerman *et al.* 1994, Heckerman 1995], a cost function for the *entire* network is evaluated. This can therefore be called a *global* approach. Constraint based algorithms like [Spirtes *et al.* 1993, Suzuki 1996, Cheng *et al.* 1997] which remove all edges for which a conditional independence statement can be derived from the data do not take into account the structure of the Bayesian network at all and can therefore be considered *local.* The algorithm presented here is also constraint based, but checks the set of CIDSs for consistency by requiring the presence of certain paths for each edge being absent in the network. Therefore, our approach is not completely local, but takes into account the *neighborhood* of each edge. The size of such a *neighborhood* can vary as it depends on the number and lengths of the required paths, for example.

From an application point of view we might state that a sufficient number of data for arriving at a unique solution is rarely ever available. If the number of data is very small, one cannot really expect too much to start with. In the intermediate region however, our new approach finds out that this is structure which could not be uniquely identified and provides a clear statement about its uncertainty. Without displaying this uncertain structure much is lost in the interpretation of the data.

In conclusion, we believe that Bayesian networks will play an increasing role in data mining applications where they are capable of displaying efficiently the important dependences in a domain. The constraint based approach is superior the global Bayesian approach in terms of learning speed. We have extended the constraint based approach to truefully display structural ambiguities which we feel is an important step towards gaining the acceptance of the user.

Acknowledgements

We would like to thank Reimar Hofmann for fruitful discussions. This work has been supported by an Ernst-von-Siemens grant and is part of the PRONEL project (PRObabilistic NEtworks and Learning) funded by the EU (Esprit Project number 28932, partners: Siemens, HUGIN, Aalborg University, TIGA).

References

[Pearl 1988] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*, Morgan and Kaufman, San Mateo, 1988.

[Wermuth and Lauritzen 1983]
N. Wermuth and S. Lauritzen. Graphical and recursive models for contingency tables, *Biometrika* **72**, pp. 537-552, 1983.

[Cooper and Herskovits 1992] G. F. Cooper and E. Herskovits. A Bayesian Method for the Induction of Probabilistic Network from Data, *Machine Learning*, Vol. **9**, pp. 309-347, July 1992.

[Heckerman 1995] D. Heckerman. A Tutorial on Learning Bayesian Networks, *Technical Report MSR-TR-95-06*, Microsoft Research, 1994.

[Heckerman *et al.* 1994] D. Heckerman, D. Geiger and D. M. Chickering. Learning Bayesian networks: the combination of knowledge and statistical data, *Technical Report MSR-TR-94-09*, Microsoft Research, 1994.

[Spirtes *et al.* 1993] P. Spirtes, C. Glymour and R. Scheines. *Causation, Prediction, and Search*, Springer, Lecture Notes in Statistics 81, 1993; *http://hss.cmu.edu/html/departments/philosophy/ TETRAD.BOOK/book.html*

[Cheng *et al.* 1997] J. Cheng, D. A. Bell, W. Liu. An Algorithm for Bayesian Belief Network Construction from Data, *AI & STAT*, 1997; Learning Belief Networks from Data: An Information Theory Based Approach, *CIKM*, 1997.

[Fung and Crawford 1990]
R. M. Fung and S. L. Crawford. Constructor: A System for the Induction of Probabilistic Models, *AAAI-90 Proceedings. Eighth National Conference*

on Artificial Intelligence, MIT Press, 1990, Vol. **2**, pp. 762-769.

[Suzuki 1996] J. Suzuki. Learning Bayesian Belief Networks Based on the Minimum Description Length Principle: An Efficient Algorithm Using the B & B Technique, *Proceedings of the international conference on machine learning*, Bally, Italy, 1996.

[Beinlich *et al.* 1989] 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, *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pp. 247-256, London,UK, 1989.

[Netica Alarm Network] Alarm Network from the Network Library of Norsys Software Corp., *http://www.norsys.com/networklibrary.html*