An Algorithm to Handle Structural Uncertainties in Learning Bayesian Network

Cleuber Moreira Fernandes

Politec Informática Ltda
SIG quadra 4 lote 173, Zip Code 70.610-440, Brasília, DF, Brazil
cleuber@compeduc.net

Wagner Teixeira da Silva, Marcelo Ladeira

University of Brasília (UnB), Computer Science Department,
Brasília, Brazil, P.O. Box 4466 – 70.919-970
{wagner,mladeira}@cic.unb.br

Abstract. Bayesian network is a graphical model appropriated to represent and to analyze uncertainty, knowledge and beliefs contained implicitly in the data. In this paper we propose the XPC algorithm for structural learning in Bayesian networks using decomposable metrics in families (a variable and its parents) in order to obtain the maximum-score network. The concept of conditional independence, based on Pearl's d-separation, is used to identify conflicting regions, where the existence of some edges depends on the non-existence of others. Hence, the user is required to choose which edges are relevant in the structure. The comparative experiments using well-know benchmarks show XPC produces better results than other algorithms mainly when the amount of data is small. A heuristic for optimizes the independence tests are also proposed. Keywords: Bayesian Networks, Structural Learning, Machine Learning, Uncertainty, Structural Ambiguities.

1 Introduction

Bayesian network (BN) is a graphical model that combines the main features of Graph Theory and Probability Theory to represent uncertainty in expert systems [1,7]. A BN is a directed acyclic graph (DAG), consisting of:

1. nodes representing random variables;
2. arcs between nodes representing dependencies and;
3. a conditional probability table attached to each node that depends on the states of adjacent predecessor nodes.

In summary, a BN represents through its structure probabilistic relationships among variables of interest [12]. This approach has proved both a powerful and a very useful tool for representing and reasoning under uncertainty conditions.

To build a BN means to identify the structure (DAG) and the numeric parameters (conditional probabilities tables). This can be made in three forms: a) by an expert domain, b) automatically from the data and c) combining the two previous forms.

There are two approaches of algorithms for structural learning in Bayesian networks given a dataset:

- Search and scoring based algorithms [12,13] and
- Dependence analysis based algorithms [2,7,10].

In this paper an algorithm for structural learning in BN given a dataset using dependence analysis is presented.

The structural learning in BN aims to generate the DAG G that it is more adherent to the joint probability distribution (JPD) P induced from a dataset D. With the dependence analysis method, it tries to explore the dependence relationships among the variables, using the d-separation criterion to obtain a structure that best represents the relationships of dependences and independence observed in D. This method uses a major property of BN, known as assertive of conditional independence (ACI) that is very efficient in structural learning in sparse BN [1]. The necessary statistical tests to identify the ACI can lead to a combinatorial explosion, what justifies the needing of using efficient heuristic techniques to reduce the search space.

When all ACI verified in a JPD P are present in the DAG G and vice versa, G and P are said be faithful each other, and G is a perfect map (P-map) of the JPD P [7].

Traditional learning algorithms based on dependence analysis produce provably correct structures under the assumptions of infinite datasets, perfect tests, and Pearl's DAG faithfulness. However, in practice this assumption doesn't necessarily happen, mainly because not always the amount of data is enough to guarantee the correct statistical tests. In this case, can appear mistakes in the conditional independence tests (CIT) and the resulting graph can not be a P-map of the probability distribution P. Therefore, structural ambiguities can be generate. These ambiguities represent the uncertainty associated to the existence of some edges among variables with weak dependence relationship.

We propose the XPC (eXtended PC) algorithm as a satisfactory solution for the problem of structural uncertainty in learning in Bayesian networks. This hybrid algorithm combines part of the techniques of the PC algorithm of Peter Spirtes and Clark Glymour [10] with some concepts of the necessary path condition, introduced by Harald Steck [11]. A heuristic to reduce the effort computational of the independence tests is also proposed in the XPC algorithm.

The XPC learns Bayesian networks very close of the real, even when not have large datasets, which is very common in several areas. That is possible because XPC doesn't eliminate edges prematurely with weak dependence; it identifies regions ambiguous and allows a domain expert to solve the uncertain associated to the presence or absence of these edges. An empirical evaluation of XPC with known benchmarks presented satisfactory results in comparison with the real networks and other algorithms of learning Bayesian network.

The remainder of the paper is organized as follows. Section 2 presents the definitions of marginal and conditional independence, Minimum Description Length (MDL), statistical test G^2 and concepts of the necessary path conditional. Section 3 describes the PC algorithm. Section 4 presents the four main phases of the XPC

algorithm and the way it works with an example. Section 5 shows the experimental results and finally, section 6 addresses some conclusions.

2 Definitions and Concepts

This section introduces some definitions and necessary concepts for the understanding of the remaining of the paper.

2.1 Marginal and Conditional Independence

Let X, Y, Z be discrete random variables with marginal probability functions P(x), P(y) and P(z) respectively and with joint probability functions P(x,y), P(y,z) and P(x,z) respectively.

1. X is (marginally) independent of Y if and only if:
   P(x|y) = P(x) whenever P(y) > 0, denoted by ($X \perp Y$).
2. X is conditionally independent of Y given Z if and only if:
   P(x|y,z) = P(x|z) whenever P(y,z) > 0, denoted by ($X \perp Y|Z$).

To check whether or not the marginal or conditional independence hypothesis holds among variables given a dataset, it is necessary to carry out statistical tests. The MDL and G^2 can be used for this purpose.

The reliability of the statistical tests depends directly on the amount of available data. When this amount is not enough can happen two types of mistakes: type (1) rejected hypothesis when it should be accepted and type (2) hypothesis accepts when it should be rejected.

2.2 Minimum Description Length

Although MDL has your origin in the information theory it has the same mathematical formula as BIC (Bayesian Information Criterion) that measures the quality of the adjustment of the network to the data using maximum likelihood [1,12].

$$MDL(X_i, pa_i) = \left(\sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} \right) - \frac{1}{2} q_i (r_i - 1).\log N \qquad (1)$$

Where N_{ijk} is the frequency of the k-th state of the variable X_i, conditioned to j-th state of its parents; N_{ij} is the marginal frequency of the variable X_i conditioned to j-th state of its parents; r_i is the cardinality of the states set of the variable X_i; q_i is the cardinality of the joint states set of the parents of X_i ($\boldsymbol{pa_i}$), and N is the number of cases in the dataset.

MDL can be used to test marginal and conditional independence among variables, as:

i) If $MDL(X,Y) - MDL(X,\phi) < \gamma$ then X is marginally independent of Y, i.e., $(X \perp Y)_P$.

ii) If $MDL(X,\{Y,\boldsymbol{S}\})-MDL(X,\boldsymbol{S}) < \gamma$ then X is conditionally independent of Y given $\boldsymbol{S}$, i.e., $(X \perp Y|\boldsymbol{S})_P$.

The computational effort of computing $MDL(X\{Y,\boldsymbol{S}\})$ is exponential in the size of $\boldsymbol{S}$ – i.e., requires time proportional to the product of the sizes of the domains of X, Y, and of all of the nodes of $\boldsymbol{S}$. The same claim is also hold for G^2 measure. The heuristic proposal in the subsection 4.2 minimizes this problem.

2.3 Statistics G^2

The statistical test G^2 possesses asymptotically the chi-square distribution (χ^2) when the independence hypothesis is true [10,14].

$$G^2(X_i, X_j, S) = 2\left(\sum_{k\in D_S}\sum_{j\in D_{X_j}}\sum_{i\in D_{X_i}} N_{ijk}\log\left(\frac{N_{ijk}}{\hat{m}_{ijk}}\right)\right) - \frac{1}{2}\chi^2_{1-\alpha,(df)} \quad (2)$$

Where N_{ijk} is the observed frequency, $\hat{m}_{ijk}$ is the expected frequency, and $\frac{1}{2}\chi^2_{1-\alpha,(df)}$ is the penalty term of complex structures.

To test marginal and conditional independence using G^2 the result of the equation above needs to be compared with the value obtained in the table of percentiles of the χ^2 distribution for the corresponding significance level α and the degree of freedom (df):

- If $G^2(X,Y,\phi) < \chi^2_{1-\alpha,(df)}$ then X is marginally independent of Y, i.e., $(X \perp Y)_P$.
- If $G^2(X,Y,S) < \chi^2_{1-\alpha,(df)}$ then X is conditionally independent of Y given $\boldsymbol{S}$, $(X \perp Y|\boldsymbol{S})_P$.

The Fig. 1 shows the equivalence between γ parameter and the significance level α; both are used to determine the intensity in the dependence relationships among the variables [5].

α	γ	**Intensity**
0.08	0...1	Weak
0.05	1...3	Medium
0.01	3...5	Strong
0.001	> 5	Very Strong

Fig. 1. Equivalence among γ and α parameter

If the marginal or conditional independence hypothesis holds, then arcs between a pair of variables are removed or not added.

2.4 Assertive of Conditional Independence and d-separation Criterion

The Assertive of conditional independence (ACI) about a joint probability distribution P, is the set of statements about all the independence relationships implicit in P.

When the joint probability distribution P(***X***) is represented by a BN on ***X***, the assertive of conditional independence on P can be verified in the structure G of that net using the d-separation criterion. The structure G of a BN is a DAG. Taking the subsets ***W, Y*** and ***S*** of ***X***, ***W*** and ***Y*** are independent, given ***S***, if ***S*** d-separates ***W*** of ***Y*** in G, according to equation (3) [1].

$$(W \perp Y \mid S)_G \Leftrightarrow S \text{ d - separate W and Y on DAG G} \tag{3}$$

A DAG G is said to be an independence map (I-map) of a JPD P if $(\boldsymbol{X}\perp\boldsymbol{Y}|\boldsymbol{S})_G \Rightarrow (\boldsymbol{X}\perp\boldsymbol{Y}|\boldsymbol{S})_P$, i.e., if all the derived ACI of G are also present in P. A graph G is said to be a minimum I-map of a JPD P if it is an I-map of P, but it stops being if some edge be removed.

2.5 Important Definitions

A directed acyclic graph G is considered optimal if it maximizes the global quality of a BN, i.e., if it is the directed acyclic graph most probabilistically adherent to P.

The edges of interest are those that if inserted in a DAG G increase the quality of the network. Therefore, the presence of such edges is a necessary condition for a DAG to be optimal. These are called certainly-present edges. Inversely, certainly-absent edges can exist. When they are eliminated of G, the quality of the network increase.

An edge $X\sim Y$ is certainly-present in a DAG G (locally optimal) or in your correspondent skeleton $\overline{G}$ if $\forall \boldsymbol{S}\subseteq \boldsymbol{V}\setminus\{X,Y\}$: $(X \not\perp Y|\boldsymbol{S})_P$ and $(Y \not\perp X|\boldsymbol{S})_P$. Inversely, an edge $X\sim Y$ is certainly-absent if $\forall \boldsymbol{S} \subseteq \boldsymbol{V}\setminus\{X,Y\}$: $(X\perp Y|\boldsymbol{S})_P$ and $(Y\perp X|\boldsymbol{S})_P$, where $\boldsymbol{V}$ is the set of variables in dataset D.

Besides the edges certainly-present and certainly-absent, ambiguous edges can also exist. An edge $X\text{--}Y$ or $Y\text{--}X$ is considered ambiguous if $(X \not\perp Y|\boldsymbol{S})_P$ given some subset $\boldsymbol{S}\subseteq\boldsymbol{V}\setminus\{X,Y\}$, and $(X\perp Y|\boldsymbol{S'})_P$ given some other subset $\boldsymbol{S'}\subseteq\boldsymbol{V}\setminus\{X,Y\}$ with $\boldsymbol{S'}\neq\boldsymbol{S}$.

2.6 Necessary Path Conditional

The necessary path condition can be seen as a local approach for the learning task, once this is conceived with the edges and present paths in the neighborhood of an absent edge. If $(X\perp Y)_p$, then should exist a such $\boldsymbol{S}\subseteq\boldsymbol{V}\setminus\{X,Y\}$ in a path between X and Y that d-separates them.

If $X\perp Y$, then should exist a $\boldsymbol{S}\subseteq\boldsymbol{V}\setminus\{X,Y\}$ in a path between X and Y that it d-separates them. The complete recursive path (sc-path) and incomplete recursive path (si-path) represent this concept in the necessary path condition.

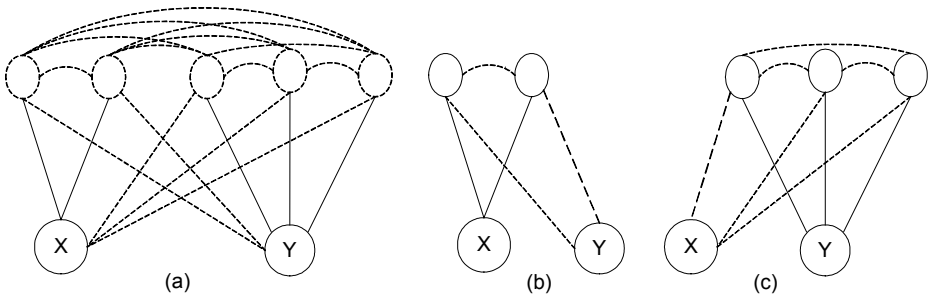

Fig. 2. Neighborhood of an absent edge X~Y, characterized by necessary path condition

In the Fig. 2, small dashed lines represent si-paths, the solid lines represent the edges between a node and its parent-candidates and long dashed lines symbolize sc-paths.

According to the Fig. 2, for an edge $X \sim Y$ to be absent in a skeleton, it is necessary to observe the following statements:

i) If doesn't exist sc-path between X and Y, Fig. 2(a), then an si-path needs to be present:
   - Between X and the parent-candidates of Y, and
   - Between Y and the parent-candidates of X, and
   - Among the parent-candidates of X and Y.

ii) If exist a sc-path between X and Y, Fig. 2(b) and 2(c), then a si-path needs to be present:
   - Between Y and each parent-candidate of X, and
   - Among the parent-candidates of X, or
   - Between X and each parent-candidate of Y, and
   - Among the parent-candidates of Y.

According with the necessary path condition, if a skeleton is locally optimal, should exist a si-path among all pair of variables $X,Y \in \boldsymbol{V}$. Inversely, if exist a pair of variables $X,Y \in \boldsymbol{V}$ so that not exist a si-path among them, the skeleton cannot be optimal. For more details see [11].

2.7 Minimal Skeletons

The minimal skeletons (MS) assist the necessary path condition and they don't possess any certainly-absent edge. They are minimal in the sense that cannot remove any edge without violating the necessary path condition. In the set of all the skeletons that satisfy the necessary path condition are the minimal skeletons and those with some edges incorrectly inserted.

All the optimal skeletons are among the graphs in conformity with necessary path condition. However, nor every optimal skeleton has to be minimal. Therefore, each MS is optimal or it is contained in an optimal skeleton. Thus, a MS is a sub graph of an optimal skeleton.

3 PC Algorithm

The PC algorithm is based on dependence analysis, and it produces good results given large datasets. This algorithm is called stepwise backward, because it begins with a complete undirected graph and with subsets S_{XY} of Adjacent(X)\{Y} of cardinality zero, after cardinality one, and so on; edges (X,Y) are removed recursively of the complete graph when a marginal or conditional independence is found, using the statistical test G^2.

The main problem of the PC algorithm is that it doesn't possess a mechanism to consider and to treat mistakes in the independence tests. Therefore, edges with weak dependence relationship can be eliminated or maintained incorrectly in the final graph, as Fig. 7(a).

4 XPC Algorithm

The XPC algorithm seeks to repair the limitation of the PC algorithm treating possible mistakes in the conditional independence tests. The solution proposal is based on an approach introduced by Steck [11] called necessary path condition and a heuristic for optimizes the independence tests (subsection 4.2). The notion of ambiguous regions allows identifying sets of uncertain and inter-dependent edges. These structural ambiguities can be resolved for interaction with an expert domain.

The necessary path condition is implemented in the XPC algorithm by rules. These rules can be simplified from way to reduce the uncertainty related to the presence of ambiguous edges. The XPC is constituted by four main phases, as below:

XPC Algorithm
1. Applies marginal and conditional independence tests;
2. Generates the rules of necessary path condition;
3. Simplifies the rules;
4. Resolves the ambiguities.

In the subsections below, the phases of XPC algorithm will be concisely presented. This paper does not present all details of the XPC algorithm. More information can be found in [5].

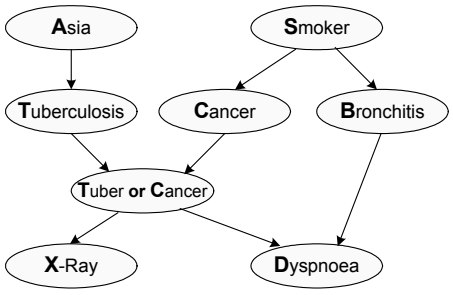

Fig. 3. Real Asia Bayesian network

An example using Asia Bayesian network (Fig. 3) shows the performance in each phase of the XPC algorithm. Asia is a well known fictitious medical example about whether a patient has tuberculosis, lung cancer or bronchitis, related to their X-ray, dyspnea, visit-to-Asia, and smoking status. This BN was introduced by Lauritzen and Spiegelhalter [16] and can be found in Norsys Website [15].

4.1 Independence Tests

The tests of marginal and conditional independence are accomplished in the first phase of the XPC algorithm. Initially XPC tests the marginal independence among all the pairs of variables $X,Y \in \boldsymbol{V}$, using MDL metric. If $(X \perp\!\!\!\perp Y|\phi)_P$ then inserts an edge between X and Y in the skeleton. Fig. 4(a) shows the initial graph produced by this procedure.

After XPC tests the conditional independence among all the pairs of variables $X,Y \in \boldsymbol{V}$ adjacent in $\overline{G}$, conditioned to subsets $\boldsymbol{S} \neq \phi$, where $\boldsymbol{S} \subseteq$ Adjacents$(X)\backslash\{Y\}$. If $(X \perp\!\!\!\perp Y|\boldsymbol{S})_P$, the edge between $X \sim Y$ is maintained, otherwise is considered ambiguous $(X\text{--}Y)$ and it will be manipulated by the necessary path condition. Fig. 4(b) shows the preliminary graph generated at the end of the phase one.

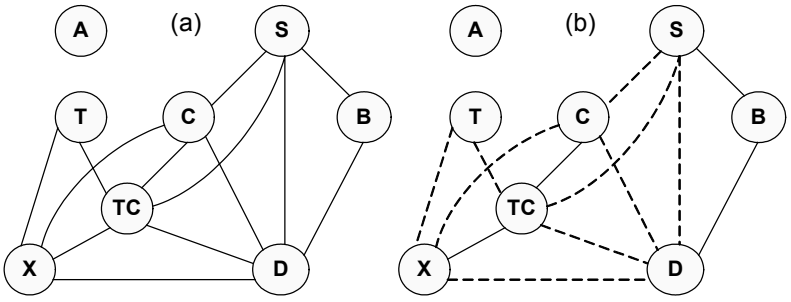

Fig. 4. Initial Graph (a) and Preliminary Graph (b)

The main difference between PC and XPC algorithm is that PC eliminates an edge of the graph immediately when finding a marginal or conditional independence, while XPC uses the necessary path condition as heuristic to consider the uncertainty (mistakes) in the statistical tests.

Therefore, edges that would be eliminated or inserted erroneously will be considered by the algorithm XPC by rules that will try to reinforce the independence hypothesis using the criterion d-separation.

4.2 Heuristic for Optimizes Independence Tests

The XPC algorithm uses the measure MDL as test of marginal and conditional independence among two variables. As presented in the sub section 2.2, the equation

1 uses frequency count of the configurations observed in the data. Configuration is a joint state of the variables *X*, *Y* and of the variables of the condition set ***S***.

In real domains, as prevention of frauds in credit cards, the amount of used variables is very big and the number of states of each one surpasses a dozen [4]. Therefore, same having a considerable sample of data, several configurations are not present. This fact does with that the matrix of frequency calculated of the data is sparse, i.e. it contends a lot of cells with zero, what turns its representation in costly memory. Exemplifying with the practical application mentioned above, when the algorithm tried to accomplish a test of conditional independence between variable *X* and *Y* given a conditional set contends five variables, it exceeded a gigabyte of available memory.

We propose a satisfactory solution for this problem, just storing in a matrix of frequencies the present configurations in the sample of data and using a B-Tree to index this matrix. In this way, the cells with zero won't be considered, reducing the necessary space for storage.

Besides, the calculations with measured MDL already accomplished are stored in a hash table and recovered when necessary, reducing the number of calculations.

4.3 Generates the Rules

The XPC algorithm implements the necessary path condition by the rules c-rule and i-rule. A c-rule is *CR(Xr,CA,CC,CI)* and an i-rule is *IR(Xr,CA,CI)* where *Xr=[X,Y]* is a pair of variables $X,Y \in \boldsymbol{V}$; *CA*, *CC*, and *CI* are sets of pairs of variables, called condition sets.

The *CA* set represents the need that a certain edge exists, the *CC* set represents the sc-path condition and the *CI* set represents the si-path condition. Together, these sets represent the conditions on the ones, which the edge *Xr* can be absent in a skeleton. It is considered that a certain rule was satisfied when all the conditions *CA*, *CC* and *CI* are found, in another words, when these sets are empty. A rule is generated for each ambiguous edge identified by the second phase of the algorithm. The example below shows the rules generated for the ambiguous edges S--C and S--TC, respectively, as Fig. 4(b).

- *CR([S,C],{[S,TC]},{[C,TC]},{[C,TC]})*
- *CR([S,TC],{[S,C]},{[C,TC]},{[C,TC]})*

4.4 Rules Simplification

After all the rules have been generated in the second phase of the algorithm, these can be simplified from way to reduce the uncertainty on the presence or absence of the ambiguous edges. In this phase some ambiguous edges can be eliminated. For a specify ambiguous edge *X*--*Y* to be absent it is necessary that at least a rule will be satisfied for this pair of variables. In other words, all the sets conditions *CA*, *CC* and *CI* should be empty after the simplification process. The phase of simplification of the rules is composed by three procedures: *remove satisfied conditions*, *condition graph* and *reduction of the number of rules* [5]

After the rules simplification phase is possible to identify the inter-dependence among the ambiguous edges. An ambiguous region consists of a set of inter-dependent uncertain edges. The edge (C,TC) was eliminated of the rules set *CC* and *CI* because it is an edge certainly-present. The Fig. 5 exemplifies the identification of the inter-dependences based on the rules set.

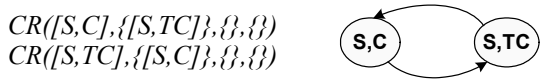

Fig. 5. Conditional Graph representing inter-dependence

4.5 Resolves the Ambiguities

In this phase of the XPC algorithm the identified ambiguous regions are exhibited graphically in an only summary graph, which allows to identify and to solve the structural uncertainties visually, as Fig. 6(a).

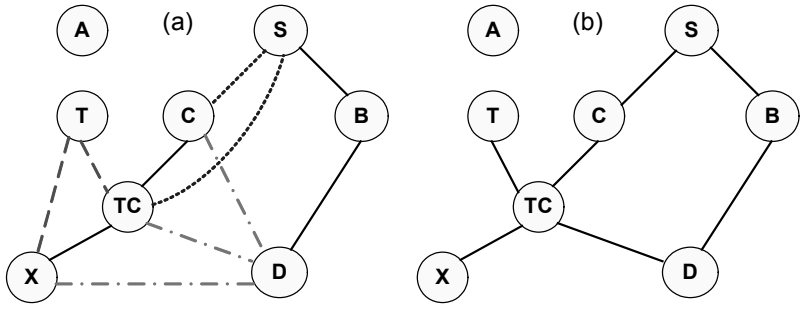

Fig. 6. Summary graph representing ambiguous regions

Thus, the domain expert can to select and to confirm the existence of an ambiguous edge and the algorithm automatically eliminates the other inter-dependent edges. Inversely, when the expert eliminates an ambiguous edge the algorithm automatically confirms the existence of the other inter-dependent edges. The Fig. 6(b) shows the graph generated after resolving the structural uncertainties.

5 Empirical Evaluation

The intention of the next example is to show more explicitly the comparison of the performance of PC and XPC algorithms. The graph shown by the Fig. 7(a) was generated by PC implemented by the software Tetrad IV [10] and the graph shown by the Fig. 7(b) was produced by XPC implemented by us in the software UnBBayes [6].

Comparing the graph of the Fig. 6(a) with the Fig. 7(a) it can be observed that the edges inserted erroneously by the PC were considered by XPC as ambiguous and handled by the necessary path condition. This positive characteristic of the algorithm

here proposed gets the attention of the domain expert for structural uncertainties that would be arbitrarily inconsiderate by PC algorithm. Thus, it is evidenced that XPC produces better results than the PC algorithm.

Both algorithms use the procedures proposed by Pearl to orient the edges [7].

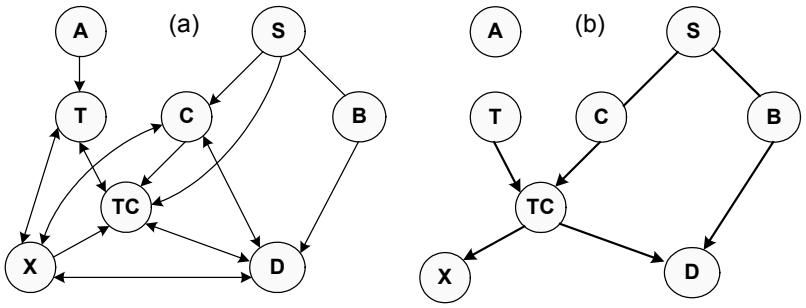

Fig. 7. Final result produced respectively by the PC and XPC

Several experiments were accomplished using known benchmarks, such as Asia, Fire, Mendel Genetics and Cancer in order to compare the structures induced with the originals in a qualitative form. Most of these files containing the description of the conditional probability tables and the BN structures were taken from the Norsys Website [15]. The structures generated by the XPC algorithm were also compared with those produced by the algorithms TPDA de Cheng, Bell and Liu [2], PC [10].

The comparison criterion used was the number of structural errors that is the sum of the number of extra edges plus absent edges in relation to the structures real of the Bayesian networks cited above. Different sizes of samples were generated from the real Bayesian networks using the procedure of cases generation of Norsys Netica [15].

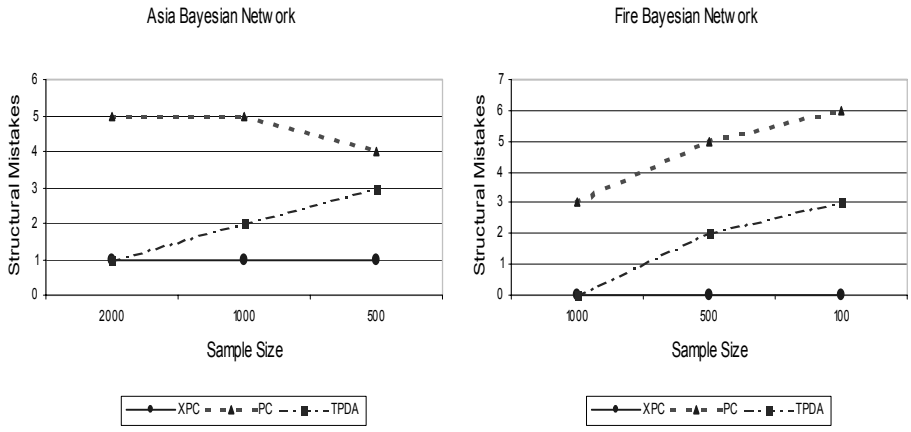

Fig. 8. Results of the accomplished experiments with Asia and Fire

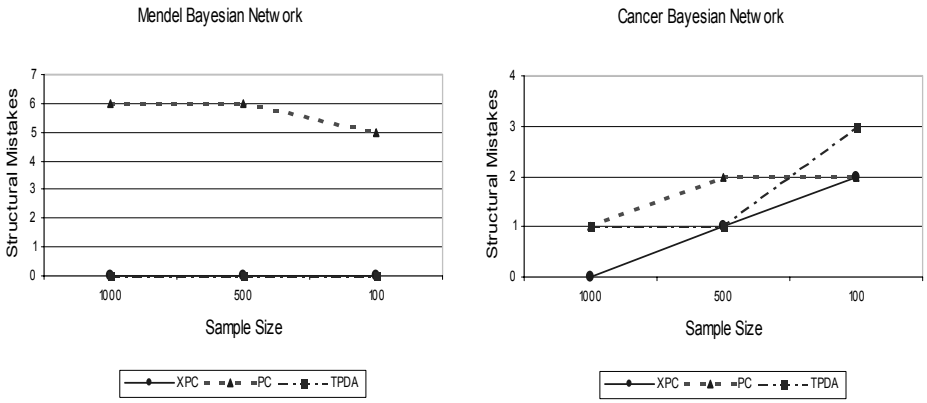

Fig. 9. Results of the accomplished experiments with Mendel and Cancer

According with the graphics of Fig. 8 and 9, the XPC algorithm had smaller variation than others when the size of the datasets was reduced. This happens because the XPC considers the uncertainty in the presence of some edges among variables with weak dependence in the joint probability distribution. This evidences the characteristic of the XPC of producing good results even when the volume of data is small.

6 Conclusions

This paper explored the structural learning in Bayesian networks, considering the neighborhood of each edge when deciding on presence or absence of this. That is done using the necessary path condition that was derived from the properties of optimal DAG. This condition allows an edge to be only absent if other edges or path are present in its neighborhood. Thus, it identifies inter-dependence among the edges, producing alternative structures.

Compared to the other algorithms based on dependence analysis, the XPC algorithm reduces the number of structural errors. That happens because the XPC algorithm does not eliminate an edge of the graph immediately when meeting a correspondent conditional independence. These uncertain edges are considered by the rules that implement the necessary path conditional, identifying structural ambiguities. These ambiguous structures can be shown to the domain expert in an summary graph, what facilitates the interpretation of multiple solutions. This fact increases the reliability and acceptance of the generated models.

The XPC algorithm and the heuristic for optimizing the independence tests are the main contributions of this research. It produces good results in structural learning in Bayesian networks even when it is not had a large dataset. That algorithm is available in the framework UnBBayes [6] that is a GNU GPL license open source software.

Josep Roure [9] proposed a heuristic approach for incremental learning i.e. to update the model with new distribution of the data. In this way, the computation effort is reduced once it will not be necessary to execute the process of learning of the

beginning. Roure assumes that is possible to store in memory whole statistical sufficiency. However, that representation has quadratic complexity in the number of variables and in the number of states of them, what became infeasible when we have so many variables. As a future work, the same heuristic for optimizing the independence tests proposed by this paper can be adapt to optimize the representation of the statistical sufficiency requested by the incremental algorithm of Roure.

References

[1] Castillo, E., Gutiérrez, J. M. and Hadi, A. S.: Expert Systems and Probabilistic Network Models. Monographs in Computer Science, Springer-Verlag, New York (1997).

[2] Cheng, J. et al.: Learning Belief Networks from Data: An Information-theory Approach. The Artificial Intelligence Journal, v.137, (2002) p.43-90.

[3] da Silva, W. T., Ladeira, M.: Data Mining in Bayesian Networks. –In: Annals of the XXII Brazilian Congress of Computation (XII JAI). Florianópolis: SBC: Edited by Ingrid Jansch Porto, (2002), v.2, p.235-286 (Portuguese).

[4] Fernandes, C.: Based solution in Bayesian Networks for Prevention of Frauds in Credit Cards. In Brazilian Symposium of Information Systems, Porto Alegre (2004), http://www.compeduc.net/professor/cleuber/publicacoes, October (Portuguese).

[5] Fernandes, C. Structural Learning in Bayesian Networks Using Small Data sets. M.Sc. thesis, Dept. of Computer Science, University of Brasília, Brazil, (2003). http://www.compeduc.net/professor/cleuber/publicacoes, October (Portuguese).

[6] Ladeira, M., da Silva, D. C., Vieira, M. H. P., Onishi, M. S., Carvalho, R. N., da Silva, W. T.: A Open and Independent of Platform Tool for Probabilistic Networks. In: National Meeting of Artificial Intelligence, (2003) (Portuguese).

[7] Pearl, J.: Causality: Models, Reasoning and Inference. Los Angeles, Cambridge University Press, (2000).

[8] Rissanen, J.: Stochastic Complexity (with discussion). Journal of Royal Statistical Society, Series B, (1987), 49:223-239 and 253-265.

[9] Roure J.: An Incremental Algorithm for Tree-shaped Bayesian Network Learning. In: 15th European Conference of Artificial Intelligence 2002. Van Harmelen ed. IOS Press.

[10] Spirtes, P., Glymour, C. and Scheines, R.: Causation, Prediction, and Search. 2nd edition. The MIT Press, Cambridge, MA, (2000). ISBN 0-262-19440-6.

[11] Steck, H.: Constraint-Based Structural Learning in Bayesian Networks using Finite Data Sets. Doctoral Thesis, Institut für Informatik der Technischen Universität München, Germany, (2001). http://tumb1.biblio.tu-muenchen.de/publ/diss/in/2001/steck.pdf. May (2001).

[12] Heckerman, D.: A Tutorial on Learning Bayesian Networks. In: Jordan, M.I. (Ed.), Learning in Graphical Models. MIT Press edition, 301-354, Cambridge, MA (1999).

[13] Buntine, W.: A Guide to the Literature on Learning Probabilistic Networks from Data. IEEE Transactions on Knowledge and Data Engineering, (1996), 8 (2), 195-210.

[14] Lindgren, B. W.: Statistical Theory. 3rd edition. Macmillan Publishing, New York (1976). ISBN 0-02-370830-1.

[15] Norsys Software Corp. Library of Belief Networks. Vancouver, BC, Canada. http://www.norsys.com/networklibrary.html, October (2004).

[16] Lauritzen, S.L., Spiegelhalter, D.J.: Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion). Journal Royal Statistical Society, [S.l.], Series B, v.50, n.2, p.425-448, 1988