Fast Conditional Independence-based Bayesian Classifier

Estevam R. Hruschka Jr. and Sebastian D. C. de O. Galvao
Federal University of Sao Carlos
estevam @dc.ufscar.br
sebastian.david@gmail.com

Machine Learning (ML) has become very popular within Data Mining (KDD) and Artificial Intelligence (AI) research and their applications. In the ML and KDD contexts, two main approaches can be used for inducing a Bayesian Network (BN) from data, namely, Conditional Independence (CI) and the Heuristic Search (HS). When a BN is induced for classification purposes (Bayesian Classifier - BC), it is possible to impose some specific constraints aiming at increasing the computational efficiency. In this paper a new CI based approach to induce BCs from data is proposed and two algorithms are presented. Such approach is based on the Markov Blanket concept in order to impose some constraints and optimize the traditional PC learning algorithm. Experiments performed with the ALARM, as well as other six UCI and three artificial domains revealed that the proposed approach tends to execute fewer comparison tests than the traditional PC. The experiments also show that the proposed algorithms produce competitive classification rates when compared with both, PC and Naive Bayes.

Categories and Subject Descriptors: Software and Applications [**Data Mining**]:

General Terms: Bayesian Networks, Supervised Learning

Additional Key Words and Phrases: Bayesian Classifiers, Markov Blanket, Conditional Independence

1. INTRODUCTION

Machine Learning (ML) [Mitchell 1997] is a field which has become very popular within Data Mining (KDD - Knowledge Discovery from Databases) and Artificial Intelligence (AI) research and their applications. ML development brought a great variety of learning algorithms in terms of the nature of induction (supervised or unsupervised learning), the knowledge representation formalism, the interaction with the environment and other issues. In the last years, Bayesian Networks (BNs) have been applied in many supervised and unsupervised learning successful applications. Therefore, many new BNs learning algorithms have been proposed [Neapolitan

Journal of Computing Science and Engineering, Vol. 1, No. 2, December 2007, Pages 162-176.

2003]. There are two main approaches, described in the literature, to the BN learning problem. The first one is based on dependency analysis and is called Conditional Independence (CI), while the second approach searches for good network structures according to a heuristic (or metric) and is called Heuristic Search (HS). The search space for learning a BN from data, however, has an exponential dimension, thus, this is a difficult problem. In this sense, approximate models can be induced and consequently, the search space of this process is likely to be reduced. To do so, some restrictions are usually imposed and often the algorithms obtain good results with acceptable computational effort.

When a supervised learning task is conducted, the BN is usually called Bayesian Classifier (BC). Considering that a BC has a known target class variable, additional constraints can be imposed to the supervised learning of Bayesian Networks. Therefore, extra computational efficiency gains can be derived. For instance, the Naive Bayes (NB) classifier [Duda and Hart 1973] can be seen as a particular Bayesian Network in which every attribute has its corresponding node connected only to the node representing the class. The traditional NB has provided good results in many domains [Friedman et al. 1997] and, consequently, it is widely used in many applications.

In the literature, the Markov Blanket (MB) concept [Pearl 1988] has been applied in conjunction with both, CI [Cheng et al. 2002] and HS [Hruschka Jr. et al. 2004] learning methods as a feature selection strategy in order to reduce the number of variables of a BC, as well as, reducing its complexity. In this sense, after inducing a BN from data, the relevant variables to a specific query (class variable in a BC) can be identified, thus, the model can be pruned.

In this work, the MB is also applied trying to reduce the model complexity. Instead of using it after the complete BN induction (as done in [Cheng et al. 2002] and [Hruschka Jr. et al. 2004]), however, the algorithms proposed in this paper, named MarkovPC and MarkovianPC, explore the MB of the class variable during the BN induction process. Therefore, it is possible to simplify the model structure (BC) while building it, and consequently, to reduce the learning algorithm complexity. To do so, our method requires a supervised learning. In other words, both MarkovPC and MarkovianPC are BNs learning algorithm designed to classification problems. Thus, it is possible to define that MarkovPC and MarkovianPC are Bayesian Classifiers (BCs) learning algorithms based on the CI approach.

Some preliminary experimental results of MarkovPC algorithm were reported in a conference paper [Galvao and Hruschka Jr. 2007] at DaWaK'2007 (9th International Conference on Data Warehousing and Knowledge Discovery). The extended version here presented provides detailed discussions of the main ideas, presents extended theoretical analysis and an extended version of MarkovPC, named MarkovianPC. The conducted experiments revealed that MarkovianPC can reduce the computational effort needed to learn a BC while maintaining the accuracy (correct classification rates) obtained when using MarkovPC.

The remainder of this work is structured as following; the next section gives an overview of Bayesian Networks and Bayesian Classifiers foundations. Section 3 presents a classic CI BN learning algorithm named PC. Section 4 introduces the proposed MarkovPC and MarkovianPC algorithms which can be seen as extensions

of the PC algorithm. Section 5 shows the performed experiments and discusses the obtained results. Finally, Section 6 describes the conclusions and points out to some future works.

2. BAYESIAN NETWORKS AND BAYESIAN CLASSIFIERS

A Bayesian Network (BN) [Pearl 1988] G has a directed acyclic graph (DAG) structure. Each node in the graph corresponds to a discrete random variable in the domain. An edge, Y → X in the graph describes a parent child relation, where Y is the parent and X is the child. All parents of X constitute the parent set of X, which is denoted by π_X. Each node of the BN structure is associated to a conditional probability table (CPT) specifying the probability of each possible state of the node, given each possible combination of states of its parents. If a node has no parents, its CPT gives the marginal probabilities of the variable it represents. In this work we propose a BN supervised learning algorithm which is based on Definition 2.1 and Theorem 2.1 taken from [Neapolitan 2003]:

Definition 2.1. Let V be a set of random variables, P be their joint probability distribution, and X ∈ V. Then a Markov Blanket MB(X) of X is any set of variables such that X is conditionally independent of all the other variables given MB(X).

THEOREM 2.1. *Suppose a Bayesian Network G (G satisfies the Markov condition given in [Neapolitan 2003]). Then for each variable X, the set of all parents of X, children of X, and parents of children of X is a Markov Blanket of X (MB(X)).*

The proof of Theorem 2.1 is straightforward and is given in [Neapolitan 2003], so we will not replicate it here. Definition 2.1 and Theorem 2.1 motivates Corollary 2.1.

COROLLARY 2.1. *Given a Bayesian Network G, the only nodes in G that have influence on the conditional distribution of a given node X (given the state of all remaining nodes) are the nodes that form the Markov Blanket of X (MB(X)).*

Corollary 2.1 is a simple consequence of Definition 2.1 and Theorem 2.1. It is proven in [Neapolitan 2003] and will not be proved here.

Definition 2.2. Instead of encoding a joint probability distribution over a set of random variables, as done by a BN, a Bayesian Classifier (BC) is special case of Bayesian Network that aims at correctly predicting the value of a discrete class variable, given the value of a vector of attribute variables (predictors).

Taking into account Corollary 2.1 and Definition 2.2, it is possible to notice that, in a BC, the Markov Blanket (Figure 1 shows a graphical example of a BN and MB(X)) of the class attribute can be used as a criterion for selecting a subset of relevant attributes for classification purposes. As it will be detailed in the sequel, this concept plays an important role in our proposed method. Meanwhile, let us address how Bayesian networks can be built.

BNs can be induced directly from domain knowledge or they can be automatically learned from data. It is also possible to combine both strategies. Learning BNs from data became an active research topic in the last decade [Neapolitan 2003], and there are two main classes of methods to perform this task: methods based on

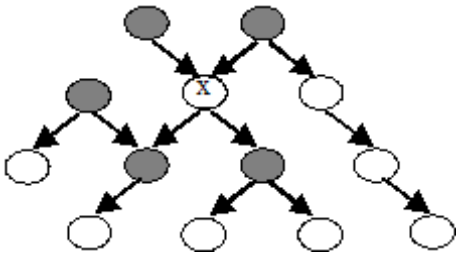

Figure 1. The shadowed nodes represent the Markov Blanket of X.

heuristic search and methods based on conditional independence tests. Bayesian Network learning methods may be used to induce a BC and this is done in this work. The BN learning algorithm used in the experiments described in Section 5 is based on the PC algorithm [Spirtes et al. 2001], which builds a BN from data based on Conditional Independence tests as addressed in the next section.

3. THE PC ALGORITHM

The PC algorithm [Spirtes et al. 2001] (summarized in Figure 3) is based upon statistical conditional independence tests. It works looking for a Bayesian Network that represents the independence relationship among variables in a dataset. This is done based on the conditional independence criteria I(X,Y|A) defined in [Pearl 1988] where A is a subset of variables, X and Y are variables. If I(X,Y|A) is true, variable X is conditionally independent of Y given A (d-separation criterion).

To verify whether X and Y are conditionally independent given A, we compute the cross entropy CE(X,Y|A) where the probabilities are their maximum likelihood estimators extracted from the data (i.e. relative frequencies). Other measures can also be used and the most common are based on statistical tests such as Chi-Squared and InfoGain [Chickering and Meek 2006; Spirtes et al. 2001], for instance.

Having as input a list with all the independencies (I(X,Y|A)) and adjacencies of each node (ADJ_X), PC first finds the graph skeleton (undirected graph) that best represents the d-separations expressed by I(X,Y|A). Afterwards, it starts establishing the orientation of the edges.

As stated in [Spirtes and Meek 1995], "if the population, from which the sample input was drawn perfectly fits a DAG C all of whose variables have been measured, and the population distribution P contains no conditional independence except those entailed by the factorization of P according to C, then in the large sample limit the PC algorithm produces the true pattern" present in the data.

4. MARKOVPC AND MARKOVIANPC

MarkovPC can be seen as an extension of the traditional PC algorithm (described in section 3). It is designed to explore the Class' Markov Blanket (CMB) trying to build more accurate and simpler classifiers. The main idea is excluding from possible structures (DAGs) those having attributes out of the CMB.

Definition 4.1. Consider G a Bayesian Classifier, all the nodes in G present in the Markov Blanket of the Class node form the Class' Markov Blanket CMB.

PC Algorithm
Problem: Given a set IND of d-separations, determine the DAG pattern faithful to IND if there is one.
Inputs: a set V of nodes and a set IND of d-separations among subsets of the nodes.
Outputs: If IND admits a faithful DAG representation, the DAG pattern gp containing the d-separations in this set.

```
1  begin
2  A.) form the complete undirected graph gp over V;
3  B.)
4  i = 0;
5  steps = 0;            //helps to measure the algorithm effort
6  repeat
7  for (each X ∈ V) {
8    for (each Y ∈ ADJX) {
9      determine if there is a set S ⊆ ADJX? {Y} such
10     that |S| = i and I({X},{Y}|S) ∈ IND; //|S|returns
11     steps = steps + 1;              //the size of set S
12     if such a set S is found {
13       SXY = S;
14       remove the edge X ? Y from gp;
15     }
16   }
17 }
18 i = i + 1;
19 until (|ADJX| < i for all X ∈ V);
20 C.)for each triple of vertices X, Y, Z such that the pair
21   X, Y and the pair Y, Z are each adjacent in C but the pair
22   X, Z are not adjacent in C, orient X–Y–Z as X->Y<-Z if
and
23   only if Y is not in SXY
24 D.) repeat
25     If A -> B, B and C are adjacent, A and C are not
26     adjacent, and there is no arrowhead at B, then
27     orient B–C as B ->C.
28     If there is a directed path from A to B, and edge
29     between A and B, then orient A-B as A->B.
30   until no more edges can be oriented.
31 end.
```

Figure 2. PC algorithm adapted from [Neapolitan 2003; Spirtes et al. 2001].

The use of the CMB allows the reduction of the computational effort needed to build the classifier structure (DAG), as well as, the simplification of its structure. Thus, MarkovPC can minimize the effort needed to both, build the classifier and to use it as a class prediction tool. In addition, having a simpler classifier makes easier the data visualization and understanding, reduce the measurement and storage requirements, reduce training and utilization times, and defy the curse of dimensionality to improve prediction performance. Figure 3 summarizes MarkovPC in an algorithmic fashion based on the PC algorithm description given by [Neapolitan 2003; Spirtes et al. 2001].

Observing figures 2 and 3 it is possible to verify that the main difference between PC and MarkovPC are in lines 8 and 9 of Figure 3. These lines define the CMB' test. In other words, the UNDIRECTED_MB(CLASS) procedure represents a set of nodes that may be contained in the CMB. It is important to say that the CMB' created by the MarkovPC algorithm is an approximation of the CMB identified by the PC algorithm. It happens because, at this point (lines 8 and 9 of the algorithm), the graph is not oriented and, consequently, it is not possible identify the parents and children of each node. Therefore, the exact CMB can not be defined yet. In this sense, UNDIRECTED_MB(CLASS) is regardless of graph orientation and it means that this set contains all the nodes that can be reached from the Class node in at most 2 edges. These nodes are eligible candidates to be part of the CMB when the graph is directed (steps C and D of Figure 3).

Definition 4.2. Consider G a Bayesian Classifier having C as a Class node, if X and Y are nodes in G, the distance from X to Y, written d(X,Y), is the minimum length of any undirected path from X to Y. All the nodes X in G for which d(C,X) < 3 form the extended Class' Markov Blanket CMB'.

THEOREM 4.1. *Consider G a Bayesian Classifier having C as a Class node, if X is a node in G and X* $\in$ *CMB, then X* $\in$ *CMB'.*

PROOF. According to Theorem 2.1, the set of all parents of a class node C (π_C), children of C (λ_C), and parents of children of C (λ_{π_C}) is a Markov Blanket of C (MB(C)). Therefore, following Definition 4.1, MB(C)=CMB. Considering Definition 4.2 and considering also that $\forall$ X $\in \pi_C$, d(C,X))=1; $\forall$ Y $\in \lambda_C$, d(C,Y))=1 and $\forall$ Z $\in \lambda_{\pi_C}$, d(C,Z))=2, it is possible to conclude that $\forall$ W $\in$ CMB, W $\in$ CMB' □

Following this strategy of defining an extended CMB' without considering the arc direction, all nodes identified by the PC algorithm as part of the real CMB will also be present in the CMB' defined by MarkovPC. Some nodes present in the CMB' defined by MarkovPC, however, may not be present in the CMB defined by the PC algorithm.

The resulting CMB' induced by means of the strategy applied by the MarkovPC algorithm can be summarized as follows: considering a consistent list of independencies (IND), the CMB' induced by the MarkovPC algorithm is not the minimal CMB, but it contains at least all the relevant nodes (present in the CMB induced by the traditional PC) to the Class variable.

When concerning the induced classifier complexity, as the MarkovPC algorithm selects the most relevant variables, it tends to induce classifiers containing a reduced number of variables. It is important to notice that, the variable selection is based

MarkovPC Algorithm
Inputs: a set V of nodes, a set IND of d-separations among subsets of the nodes and a CLASS class-node ∈ V.
Outputs: If IND admits a faithful DAG representation, the DAG pattern gp containing the d-separations in this set optimized as a Bayesian classifier.

```
1  begin
2  A.) form the complete undirected graph gp over V;
3  B.)
4  i = 0;
5  steps = 0;            //helps to measure the algorithm effort
6  repeat
7  for (each X ∈ V) {       //first X should be the CLASS node
8    if UNDIRECTED_MB(CLASS) ⊄ X {
9      remove X from gp;
10   }
11   else {
12     for (each Y ∈ ADJX) {
13       determine if there is a set S ⊆ ADJX? {Y} such
14       that |S| = i and I({X},{Y}|S) ∈ IND;
15       steps = steps + 1;
16       if such a set S is found {
17         SXY = S;
18         remove the edge X ? Y from gp;
19       }
20     }
21   }
22   steps = steps + 1;
23 }
24 i = i + 1;
25 until (|ADJX| < i for all X ∈ V OR i<=|S|);
26 C.)for each triple of vertices X, Y, Z such that the pair
27   X, Y and the pair Y, Z are each adjacent in C but the pair
28   X, Z are not adjacent in C, orient X – Y – Z as X -> Y <- Z
29   if and only if Y is not in SXY
30 D.)repeat
31     If A -> B, B and C are adjacent, A and C are not
32     adjacent, and there is no arrowhead at B, then
33     orient B–C as B ->C.
34     If there is a directed path from A to B, and edge
35     between A and B, then orient A-B as A->B.
36   until no more edges can be oriented.
37 end.
```

Figure 3. MarkovPC algorithm.

on an approximate CMB (CMB'), thus the number of selected variables depends upon the CMB' size. Considering the characteristics described in the MarkovPC algorithm, on one hand it is possible to say that, when working with domains having a high number of features and a small CMB', the MarkovPC effort tends to be smaller than the one done by the traditional PC. On the other hand, in a situation where a domain has a CMB' containing a high number of variables, MarkovPC tends to lead to no variable elimination, and, thus, its effort can be higher than the one needed by the traditional PC.

Considering that testing whether a variable is present in the CMB' has the same complexity present in line 10 of PC algorithm (Figure 2), it is possible to analyze the above situation in an extreme case, in which a domain with N variables (all present in the CMB') and a list of independencies sets (IND) having cardinalities from 0 to M (see [Spirtes et al. 2001] for a more detailed description of independency cardinality) are given. In such a situation, the MarkovPC algorithm will need (N * (M+1)) tests more that the Traditional PC algorithm. It happens because, for each independency cardinality set, the MarkovPC will test whether each variable is present in the CMB' or not. In most of the domains, however, this aforementioned extreme situation does not happen, thus, the MarkovPC tends to need less effort than the traditional PC.

Another difference between Figure 2 and Figure 3 is present in the "until" clause (line 19 in Figure 2 and line 25 in Figure 3). This clause in MarkovPC has an "or" operator which is not present in the original PC algorithm. The motivation for inserting this "or" operator is to eliminate unnecessary tests. This modification is not necessary to the Markov Blanket strategy proposed in MarkovPC, but it is implemented trying to reduce the computational complexity of the algorithm. It is important to observe that this modification (inserting the "or" operator) do not change the algorithm behavior in terms of the classifier to be induced.

Taking into account the Average Correct Classification Rates (ACCRs), the MarkovPC algorithm should produce results consistent to the ones produced by the traditional PC.

Based on the algorithm depicted in Figure 3 (MarkovPC algorithm) another implementation is proposed in this work. This new implementation is named MarkovianPC and uses the same main ideas already presented in MarkovPC. The only difference between the MarkovPC and the MarkovianPC algorithms is related to the UNDIRECTED_MB(CLASS) function. In the MarkovianPC algorithm, instead of using the extended CMB' proposed in Definition 5, the UNDIRECTED_MB(CLASS) works with the Markov Blanket of a Markov Network. The following definitions help in the MarkovianPC algorithm understanding.

Definition 4.3. Consider an undirected graph G = (V,E). If G is a Markov Network, then each node X $\in$ V represents a random variable in a set of random variables and each edge (X,Y) $\in$ E represents a dependency between the random variables X and Y.

Definition 4.4. Consider G a Markov Network, if X and Y are nodes in G, the Markov Blanket of X (here called Markovian Markov Blanket of X MMB(X)) is defined to be formed by all Y such that d(X,Y)=1.

As mentioned before, the UNDIRECTED_MB(CLASS) function is called in the MarkovPC algorithm in line 9. At this point, the algorithm is working with an undirected graph (UG) structure. In this sense, the UG can be seen as a Markov Network. Therefore, the extended Class' Markov Blanket (CMB') can be defined based on Definition 4.4 as showed in Definition 4.5.

Definition 4.5. Consider G a Markov Network having C as a Class node, all the nodes in G present in the Markov Blanket of the Class node form the Class' Markovian Markov Blanket CMMB.

Following along these lines, it is possible to verify that the only difference, between the MarkovPC and the MarkovianPC algorithms, is that the first uses the CMB' and the later uses the CMMB to define the output of the UNDIRECTED_MB(CLASS) function. This single difference, however, may have great impact in the induced classifier structure. It happens because when using CMMB, the obtained Markov Blanket tends to be smaller than the one obtained using CMB'. Therefore, the MarkovianPC algorithm tends to induce simpler and faster BCs.

It is important to notice that, as happens with the MarkovPC algorithm, the output of the MarkovianPC algorithm is a Bayesian Classifier (special case of Bayesian Network) and not a Markov Network. The CMMB concept, however, is defined for Markov Networks, thus, using the CMMB in the MarkovianPC algorithm is a heuristic solution employed to reduce the computational effort needed by the learning algorithm. In this sense, it is not possible to prove that all $X \in CMB$ is also present in CMMB. Next section describes the conducted experiments and analyzes the obtained results.

5. EXPERIMENTS AND RESULTS

Trying to verify the soundness of the proposed MarkovPC and MarkovianPC algorithms, when compared to the traditional PC algorithm, a number of empirical classification experiments were conducted. The main aspects to be considered when concerning the MarkovPC, as well as, the MarkovianPC behaviors are twofold: the Average Correct Classification Rates (ACCRs) and the classifier (structure) complexity. The remaining of this section initially describes the knowledge domains used in the experiments as well as the experimental methodology adopted. The results from the experiments are then presented and analyzed.

Ten domains were used in our simulations. A well-known Bayesian Network domain, namely ALARM [Beinlich et al. 1988]; six benchmark problems from the U. C. Irvine repository [Asuncion and Newman 2007], namely, Car, kr-vs-kp, Lung Cancer (Lung), Postoperative-Patient-Data (Patient), Solar-flare 1 (Flare 1) and Solar-flare2 (Flare 2); and finally, three synthetic domains namely, Synth1, Synth2 and Synth3. Table I summarizes datasets characteristics.

The ALARM network has 8 prediction variables and, in this work, all these variables are used as classes. In this sense, 8 different experiments were conducted with the ALARM domain; in each one, a different class is assumed. Table II shows the prediction variables (classes) names and their domain size.

The description of each UCI domain can be downloaded from the UCI Repository site [Asuncion and Newman 2007]. The artificial domains were simulated in order to

Table I. Datasets Description with dataset name (Data), number of attributes plus class (AT), number of instances (IN) and number of classes (Cl).

Domain	#Attributes	#Instances	#Classes
Alarm	38	10000	See Table II
Car	7	1728	4
Kr-vs-kp	37	3196	2
Lung	57	32	3
Patient	9	90	3
Flare 1	13	323	7
Flare 2	13	1066	7
Synth1	32	10000	2
Synth2	32	10000	2
Synth3	32	10000	2

Table II. ALARM prediction variables and the number of possible values each one can assume.

Variable	#possible states
Anaphylaxis	2
Intubation	3
KinkedTube	2
Disconnect	2
Hypovolemia	2
InsuffAnesth	2
LVFailure	2
PulmEmboulus	2

verify the behavior of the proposed method. Such simulations were performed manually building Bayesian Networks (BNs) to encode a joint probability distribution over a set of random variables and, thus, reproducing hypothetical circumstances. Next, the BNs were used to generate synthetic datasets (SDs) containing 10,000 records (instances) which are representative of each BN.

There are two main benefits in working with datasets representing previous known circumstances. The first one is that it is possible to a priori know the truth dataset probability distribution and its characteristics. Hence, the results obtained in the experiments can be analyzed in a more consistent way. Second, it is possible to inspect the behavior of an algorithm in a very specific situation of interest.

In this sense, three synthetic BNs, namely, Synth1, Synth2 and Synth3 were built and their network structures are depicted in Figure 4. The Synth1 network represents a domain, having 32 variables, in which only one variable directly influences the class variable (the CMB has a single variable forming it) and all variables (nodes in the graph) have at most one parent. Therefore, the Synth1 structure represents a polytree. In real world applications problems polytrees hardly ever are suitable to model the probability distribution of the variables [Pearl 1988], thus, it is not common to have problems with these characteristics in practice. Polytrees, however, are suitable structures to verify the behavior of a classifier in problems where the variables have simple interdependencies relationships.

The Synth2 BN describes a domain, having 32 variables, in which 14 variables

directly influence the class variable. In this BN, each variable can have at most 3 parents and it allows more complex interdependency relationships among variables than polytrees structures. Therefore, Synth2 is a less restrictive model than Synth1 and problems that can be represented by such a BN (Synth2) are more common in practical Data Mining and Machine Learning problems.

The last synthetic BN (Synth3) also describes a domain with 32 variables. In this BN, however, the variables can have at most seven parents. Furthermore, all variables are present in the CMB.

Intending to perform a more robust comparative analysis, besides presenting classification results (ACCRs) obtained using the proposed MarkovPC and MarkovianPC algorithms, this section also shows the performance of the traditional PC and Naive Bayes Classifier [Duda and Hart 1973] when applied to all the 10 described domains. The Naive-Bayes classifier implementation can be found in the WEKA environment [Witten and Frank 2005]. The PC, MarkovPC and MarkovianPC were also implemented in the WEKA's framework and are available for research purposes upon request. Besides, all the experiments were conducted in the WEKA environment.

In all the performed classification a 10-fold cross validation strategy was applied, and the same training and test files were used by all algorithms. Table III presents the average obtained results (ACCRs).

Table III reveals that, considering the aforementioned domains, the computational effort (number of tests) required to induce a BC from data using both proposed methods was lower than that of the PC algorithm. In addition, on average, using the MarkovPC was possible to induce a BC using only 43.36% of the computational effort required by PC. Also, using the MarkovianPC it was possible to induce a BC using only 36.40% of the computational effort required by MarkovPC.

Only with the Patient domain, the MarkovPC and the MarkovianPC executed more comparative tests than the traditional PC. It happened, because both methods did not eliminate any variable form the model and considered all them forming the CMB'. This is a very interesting result to illustrate the extreme situation described in Section 4. As the Patient domain has 9 attributes and its independency list (IND) has two cardinalities sets (cardinality 0 and cardinality 1), the MarkovPC and the MarkovianPC performed (9 * 2) tests more than the traditional PC.

Trying to get more robust conclusions (when analyzing the ACCRs) the Wilcoxon statistical test [Hays 1994] was applied to compare the results obtained using the proposed algorithms as well as the PC and the Naive Bayes. The Wilcoxon test is a nonparametric procedure employed in a hypothesis testing situation involving a single sample in order to determine whether or not a sample is derived from a population in which the median θ is equal to a specified value. If the Wilcoxon signed-ranks test yields a significant result, the researcher can conclude there is a high likelihood the sample is derived from a population with a median value other than θ [Sheskin 2004].

On one hand, the achieved results allow concluding that the use of the PC, as well as the MarkovPC and the MarkovianPC algorithms brought no significant difference regarding the obtained ACCRs. On the other hand, results revealed that the Naive-Bayes classifier performance was significantly lower when compared to the other

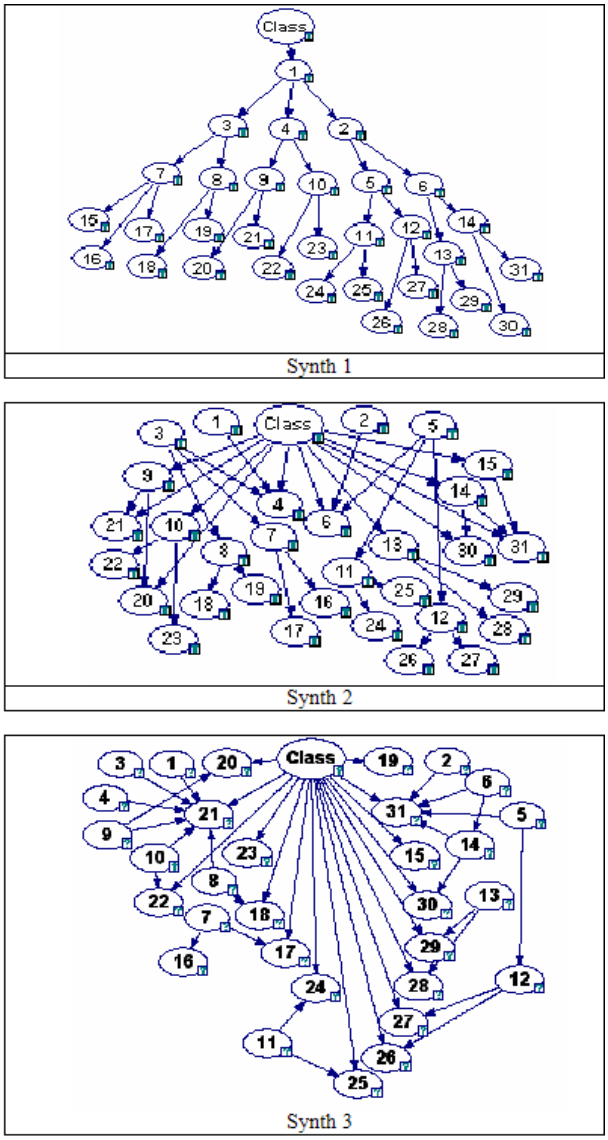

Figure 4. Bayesian Networks representing Synth1, Synth2 and Synth3 domains. The graphical representations were created using GeNie Software.

employed algorithms. Therefore, it is possible to consider that, besides needing less computational effort, MarkovPC, as well as, MarkovianPC performed as good as the traditional PC when the ACCRs are concerned. In addition, considering the particular domains presented in Table I, both proposed algorithms performed better than the Naive Bayes classifier.

When concerning the classifier complexity, it is possible to state that PC and Naive Bayes generate classifiers containing all variables present in the dataset do-

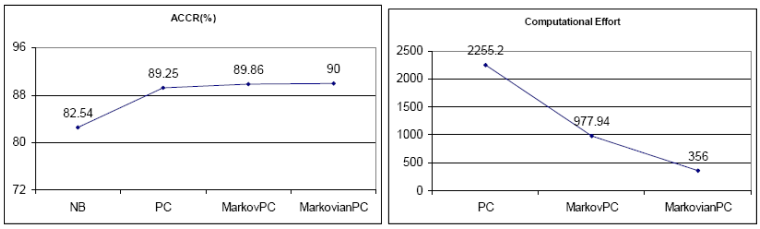

Figure 5. a) ACCRs obtained using the Naive-Bayes, the PC, the MarkovPC and the MarkovianPC algorithms. b) the average computational effort needed to induce a BC using the PC, the MarkovPC and the MarkovianPC algorithms.

Table III. Number of steps and the Average Correct Classification Rates obtained with each domain using PC, MarkovPC (MrkvPC), MarkovianPC (MrvnPC) and Naive Bayes (NB) algorithms.

Domain	Computational Effort (steps)			ACCR(%)			
	PC	MrkvPC	MrkvnPC	PC	MrkvPC	MrkvnPC	NB
ALARM Hypov.	2758	1121	330	97.98	98.38	98.420	96.55
ALARM LVFail.	2758	1135	379	98.95	99.03	99.038	96.41
ALARM Anaphyl.	2758	1052	183	98.98	98.98	98.989	97.26
ALARM Ins. An.	2758	1442	415	81.85	84.77	86.327	63.34
ALARM Pulm.	2758	1160	528	99.21	99.42	99.427	97.30
ALARM Intub.	2758	1458	502	97.55	98.46	98.471	84.96
ALARM Kinked.	2758	1327	430	98.87	98.91	98.918	85.30
ALARM Disconn.	2758	1145	437	96.16	98.74	98.749	92.95
Synth1	1663	797	445	89.16	89.16	89.160	83.42
Synth2	823	599	136	93.20	93.20	93.200	93.20
Synth3	908	610	123	85.30	86.74	87.320	83.16
Car	76	34	34	89.81	93.57	93.86	85.64
Kr-vs-kp	6129	2127	1386	96.08	94.08	94.023	87.76
Lung-Cancer	5881	2151	452	75.00	75.00	75.000	46.87
Patient	36	54	54	71.11	71.11	71.111	68.88
Solar-flare 1	418	176	70	72.75	73.06	72.755	66.25
Solar-flare 2	342	238	156	75.23	74.85	74.766	73.92
Average	2255.2	977.94	356	89.25	89.86	90	82.54
Standard Dev.	1787.1	643.45	315.72	10.33	10.38	10.38	14.28

mains. MarkovPC and MarkovianPC, however, tended to reduce the number of variables present in the induced classifiers. Considering the 17 simulations reported in Table III, on average, PC and Naive Bayes classifiers produced models having 31.52 variables. MarkovPC, on the other hand, produced classifiers having 7.52 variables on average while MarkovianPC reduced even more the classifiers complexity; on average, MarkovianPC produced classifiers having 3.58 variables. Thus, the classifiers induced by MarkovPC have only 23.85% of the variables present in the classifiers induced by the PC algorithm and the Naive Bayes approach. In addition, the classifiers induced using MarkovianPC have only 47.6% of the variables present in the classifiers induced by MarkovPC. Figure 5 summarizes the results obtained in the conducted experiments in terms of accuracy (Figure 5 a) and complexity (Figure 5b).

Analyzing Figure 5 one can observe that, on average, considering the conducted

experiments, the MarkovianPC algorithm tends to produce the higher ACCRs while needing less computational effort than the other employed algorithms. It is important to notice that the Naive Bayes is not present in Figure 5b because such a classifier do not need to build a model from data, thus, its computational effort to be induced can be considered zero.

6. CONCLUSIONS AND FUTURE WORK

This paper proposes and discusses a new CI BC learning approach based on the Markov Blanket concept. Two algorithms were implemented, namely the MarkovPC and the MarkovianPC, to induce BCs from data. Instead of using the Markov Blanket concept to select variables after the BN induction, as done in other works described in the literature, the proposed approach uses the Markov Blanket concept in order to impose some constraints and optimize the traditional PC algorithm while inducing the BC. It is important to state that this approach is designed specifically to classification tasks.

Experiments performed with a number of domains revealed that both, the MarkovPC and the MarkovianPC algorithms tend to be more accurate (in terms of ACCRs) than the traditional PC and Naive Bayes. In addition, both proposed algorithms produced simpler classifiers demanding less comparison tests, during the learning procedure, than the PC algorithm. Also, the MarkovianPC algorithm tends to be faster than MarkovPC. The approximate CMB' found by MarkovPC is proved to be consistent with CMB. And the CMMB found by the MarkovianPC can be considered a consistent and promising heuristic approach.

Authors intend to continue along this line of investigation and plan to introduce the main ideas present in MarkovPC as well as in MarkovianPC in other CI learning algorithms. Another interesting future work is to use the proposed Markov Blanket strategy when performing CI statistical tests.

7. ACKNOWLEDGMENTS

Authors acknowledge the Brazilian research agency FAPESP for its financial support.

REFERENCES

Asuncion, A. and Newman, D. 2007. UCI machine learning repository - http://www.ics.uci.edu/∼mlearn/MLRepository.html.

Beinlich, I., Suermondt, J., Chavez, M., and Cooper, G. 1988. The alarm monitoring system: A case study with two probablistic inference techniques for belief networks. In *Second European Conference on Artificial Intelligence in Medicine*.

Cheng, J., Greiner, R., Kelly, J., Bell, D., and Liu, W. 2002. Learning bayesian networks from data: an information-theory based approach. *Artificial Intelligence 137,* 1-2, 43–90.

Chickering, D. M. and Meek, C. 2006. On the incompatibility of faithfulness and monotone dag faithfulness. *Artificial Intelligence 170,* 8, 653–666.

Duda, R. O. and Hart, P. E. 1973. *Pattern Classification and Scene Analysis.* Wiley, New York.

Friedman, N., Geiger, D., and Goldszmidt, M. 1997. Bayesian network classifiers. *Machine Learning 29,* 2-3, 131–163.

Galvao, S. and Hruschka Jr., E. R. 2007. A markov blanket based strategy to optimize the induction of bayesian classifiers when using conditional independence learning algorithms. In

Data Warehousing and Knowledge Discovery, 9th International Conference, DaWaK 2007. 355–364.

Hays, W. 1994. *Statistics*, 5th ed. Wadsworth Publishing.

Hruschka Jr., E. R., Hruschka, E. R., and Ebecken, N. F. F. 2004. Feature selection by bayesian networks. In *Canadian Conference on AI - Lecture Notes in Computer Science V. 3060*. 370–379.

Mitchell, T. M. 1997. *Machine Learning*. The McGraw-Hill Companies, Inc.

Neapolitan, R. E. 2003. *Learning Bayesian networks*. Prentice Hall, Upper Saddle River, NJ.

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

Sheskin, D. J. 2004. *Handbook of Parametric and Nonparametric Statistical Procedures*, 3rd ed. Chapman and Hal/CRC.

Spirtes, P., Glymour, C., and Scheines, R. 2001. *Causation, Prediction, and Search, Second Edition (Adaptive Computation and Machine Learning)*. The MIT Press.

Spirtes, P. and Meek, C. 1995. Learning bayesian networks with discrete variables from data. In *KDD - Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*. 294–299.

Witten, I. H. and Frank, E. 2005. *Data Mining: Practical machine learning tools and techniques*, 2nd ed. Morgan Kaufmann, San Francisco.

Estevam R. Hruschka Junior received his B.Sc. and M.Sc. degrees in Computer Science from State University of Londrina, Brazil, and from University of Brasilia, Brazil, in 1994 and 1997, respectively, and his Ph.D. degree in Computational Systems from Federal University of Rio de Janeiro, Brazil, 2003. He is currently researcher and assistant professor at Federal University of Sao Carlos (UFSCar), Brazil. His main research interests are data mining and Bayesian networks, with particular emphasis on classification and clustering algorithms, feature selection, missing values imputation and evolutionary algorithms.

Sebastian D. C. de O. Galvao has a Master's degree in Computer Science from Federal University of Sao Carlos, Brazil, and has been interested in the Machine Learning research since 2003. Papers from the author include optimization of data mining tasks such as attribute selection and classification. Currently the author is working on the induction of Bayesian classifiers using structure learning by conditional independence tests.