An Efficient and Scalable Algorithm for Local Bayesian Network Structure Discovery

Sérgio Rodrigues de Morais and Alex Aussem

University of Lyon, F-69000, Lyon, University of Lyon 1,
LIESP Laboratory, 69622 Villeurbanne, France

Abstract. We present an efficient and scalable constraint-based algorithm, called Hybrid Parents and Children (HPC), to learn the parents and children of a target variable in a Bayesian network. Finding those variables is an important first step in many applications including Bayesian network structure learning, dimensionality reduction and feature selection. The algorithm combines ideas from incremental and divide-and-conquer methods in a principled and effective way, while still being sound in the sample limit. Extensive empirical experiments are provided on public synthetic and real-world data sets of various sample sizes. The most noteworthy feature of HPC is its ability to handle large neighborhoods contrary to current CB algorithm proposals. The number of calls to the statistical test, en hence the run-time, is empirically on the order $O(n^{1.09})$, where n is the number of variables, on the five benchmarks that we considered, and $O(n^{1.21})$ on a real drug design characterized by 138,351 features.

Keywords: Bayesian network structure learning, constraint-based methods, feature selection.

1 Introduction

This paper presents a novel algorithm which thoroughly exploits Bayesian networks (BN) theory in order to find the variables that are directly associated with a target, that is, those variables that remain probabilistic associated with a target even when conditioning on values of any subset of the other variables in a data set. Those variables are often called among the BN community as the *parents and children* of the target variable. A BN is a probabilistic model formed by a structure and parameters. The structure of a BN is a directed acyclic graph (DAG), whilst its parameters are conditional probability distributions associated with the variables in the model. The graph of a BN itself is an independence map (I-map), which is very useful for many applications, including feature subset selection, dimensionality reduction, and inferring causal relationships from observational data. However, the recent explosion of high dimensional data sets poses a serious challenge to existing BN learning algorithms. A principled solution to this problem is to directly seek a local network around the target without having to find the whole DAG.

J.L. Balcázar et al. (Eds.): ECML PKDD 2010, Part III, LNAI 6323, pp. 164–179, 2010.

Two types of BN structure learning methods have been proposed so far: *constraint-based* (CB) and *score-and-search* methods. Basically, CB learning methods systematically check the data for conditional independence relationships, whilst score-and-search methods make use of a score function for evaluating graphical structures with regard to the data set. While score-and-search methods are efficient for learning the full BN structure (see [18] for instance), the ability to scale up to hundreds of thousands of variables is a key advantage of CB methods over score-and-search methods. Several CB algorithms have been proposed recently [8, 11, 14–16, 20, 21] for local BN structure learning. They construct a local skeleton (i.e., the edges without their orientation) around the target node without having to construct the whole BN first, hence their scalability. Fortunately, the scalability of CB methods does not come at the loss of accuracy; they were recently shown to be among the top-ranking entrants in the WCCI2008 Causation and Prediction Challenge [9]. One of the best prediction accuracy was obtained by [4] using CB methods discussed in [20]. Even the more sophisticated approaches using novel structure-based causal discovery, such as [3], used the standard CB methods, in the first phase of the discovery process, to find a local skeleton.

CB methods can be divided in two classes: *incremental* methods (e.g., IAMB [20], BFMB [8]) and *divide-and-conquer* methods (e.g., MMMB [20]), PCMB [11]). While these algorithms are appropriate for situations where the number of parents and children (PC set) is relatively small, they are plagued by a severe problem: the number of false negatives (missing variables) increases swiftly as the size of the PC set increases[1]. There are mainly two reasons for this. The first is the unreliability of the conditional independence tests as the conditioning sets become large. This well known problem is common to all CB methods and has led several authors to reduce, as much as possible, the size of the conditioning sets with a view to enhancing the *data-efficiency* of their methods [8, 11]. The second reason is that the decisions for a node to enter the candidate PC set are often too severe and conservative. In practice, those problems plague all CB methods for target variables with many adjacent nodes and relatively few instances.

In this paper, we introduce an algorithm called *Hybrid Parents and Children* (HPC) that combines ideas from incremental and divide-and-conquer methods in order to alleviate the problem discussed above. A thorough discussion is provided for explaining why HPC can handle large number of adjacent nodes while still being sound in the sample limit. Extensive empirical experiments are then carried out on public synthetic data sets to assess its accuracy and scalability. Such experiments show that significant improvements in accuracy are obtained. In addition, the empirical number of calls to the independence test (and hence the *effective* complexity) is only $O(n^{1.09})$ in practice on synthetic data and $O(n^{1.21})$ on real-world data, where n is the number of variables. Finally, a proof of HPC's

[1] For instance, [4], the winners of the WCCI2008 challenge, outputs only 9 features out of 14 true features with a data set that consists of 500 instances.

correctness under the so called *faithfulness condition* is provided in the final part of this article.

2 Preliminaries

Formally, a BN is a tuple $< \mathbb{G}, P >$, where $\mathbb{G} =< \mathbf{U}, \mathbf{E} >$ is a directed acyclic graph (DAG) whose nodes represent the variables in the domain $\mathbf{U}$, and whose edges represent direct probabilistic dependencies between them. P denotes the joint probability distribution on $\mathbf{U}$. The BN structure encodes a set of conditional independence assumptions: that each node X_i is conditionally independent of all of its non descendants in $\mathbb{G}$ given its parents $\mathbf{Pa}_i^{\mathbb{G}}$. These independence assumptions, in turn, imply many other conditional independence statements, which can be extracted from the network using a simple graphical criterion called d-separation [12].

We denote by $X \perp_P Y|\mathbf{Z}$ the conditional independence between X and Y given the set of variables $\mathbf{Z}$ where P is the underlying probability distribution. Note that an exhaustive search of $\mathbf{Z}$ such that $X \perp_P Y|\mathbf{Z}$ is a combinatorial problem and can be intractable for high dimension data sets. We use $X \perp_{\mathbb{G}} Y|\mathbf{Z}$ to denote the assertion that X is d-separated from Y given $\mathbf{Z}$ in $\mathbb{G}$. We denote by $\mathbf{dSep}(X, Y)$, a set that d-separates X from Y. If $< \mathbb{G}, P >$ is a BN, $X \perp_P Y|\mathbf{Z}$ if $X \perp_{\mathbb{G}} Y|\mathbf{Z}$. The converse does not necessarily hold. We say that $< \mathbb{G}, P >$ satisfies the *faithfulness condition* if the d-separations in $\mathbb{G}$ identify *all and only* the conditional independencies in P, i.e., $X \perp_P Y|\mathbf{Z}$ if and only if $X \perp_{\mathbb{G}} Y|\mathbf{Z}$.

An important concept of BN is the Markov blanket of a variable, which is the set of variables that completely shields off this variable from the others. In other words, a Markov blanket $\mathbf{M}_T$ of T is any set of variables such that T is conditionally independent of all the remaining variables given $\mathbf{M}_T$. A Markov boundary, $\mathbf{MB}_T$, of T is any Markov blanket such that none of its proper subsets is a Markov blanket of T. Suppose $< \mathbb{G}, P >$ satisfies the faithfulness condition. Then, for all X, the set of parents, children of X, and parents of children (spouses) of X is the unique Markov boundary of X. A proof can be found for instance in [10]. We denote by $\mathbf{PC}_T^{\mathcal{G}}$, the set of parents and children of T in $\mathcal{G}$, and by $\mathbf{SP}_T^{\mathcal{G}}$, the set of spouses of T in $\mathcal{G}$, i.e., the variables that have common children with T. These sets are unique for all $\mathcal{G}$, such that $< \mathcal{G}, P >$ satisfies the faithfulness condition and so we will drop the superscript $\mathcal{G}$.

2.1 Constraint-Based Structure Learning

The induction of local BN structures is handled by CB methods through the identification of local neighborhoods. Hence their scalability to very high dimensional data sets. CB methods systematically check the data for conditional independence relationships in order to infer a target's neighborhood. Typically, the algorithms run a χ^2 independence test when the data set is discrete and a Fisher's Z test when it is continuous in order to decide on dependence or

independence, that is, upon the rejection or acceptance of the null hypothesis of conditional independence. The reliability of a conditional independence test is dependent on the number of instances in the data set and the degree of freedom of the test. A practical consideration regarding the reliability of a conditional independence test is the size of the conditioning set as measured by the number of variables in the set, which in turn determines the number of values that the variables in the set may jointly take. Large conditioning sets produce sparse contingency tables and unreliable tests. This is why it is difficult to learn the neighborhood of a node having a large degree with CB methods. The number of possible configurations of the variables grows exponentially with the size of the conditioning set.

3 Pitfalls and Related Work

Both families of CB methods, that is *divide-and-conquer* and *incremental* methods, can be very useful when working in very high dimension data sets because they do not have to search the whole structure of a BN in order to find the local network around a target. Divide-and-conquer methods are based on the identification of the direct neighborhood of a target, that is, its parents and children ($\mathbf{PC}_T$). Of course, divide-and-conquer methods can be iteratively used for finding local networks or the Markov boundary of a target, for instance that is exact what do the algorithms *MMMB* [19] and *PCMB* [11]. Those algorithms identify first the parents and children of T ($\mathbf{PC}_T$) and then the set of the spouses of T ($\mathbf{SP}_T$) for forming the Markov boundary of T ($\mathbf{MB}_T = \mathbf{PC}_T \cup \mathbf{SP}_T$). On the other hand incremental methods incrementally searches the whole Markov boundary of a target ($\mathbf{MB}_T$) without distinguishing the two subsets $\mathbf{PC}_T$ and $\mathbf{SP}_T$. However, incremental methods can also be used for searching $\mathbf{PC}_T$. This can be achieved by first finding $\mathbf{MB}_T$ and then eliminating the variables of $\mathbf{SP}_T$ ($\mathbf{PC}_T = \mathbf{MB}_T \setminus \mathbf{SP}_T$). This procedure is very simple and can be summarized as follows: $\forall X \in \mathbf{MB}_T, X \in \mathbf{SP}_T$ (that is, $X \notin \mathbf{PC}_T$) if and only if $\exists \mathbf{Z} \in (\mathbf{MB}_T \setminus X)$ such that $X \perp_P T|\mathbf{Z}$. Nonetheless incremental methods are considered as *data-inefficient* because it often makes use of unnecessary large conditioning sets [11].

Inferring automatically from data the set $\mathbf{PC}_T$ by the use of independence tests is not as easy as one could think at first sight. The following property serves as the common rule for discarding non-adjacent variables: Let $\mathbf{U}$ be the set of all the variables in the data set, thus, under the faithfulness assumption, X and Y are not adjacent in $\mathcal{G}$ if and only if $\exists \mathbf{Z} \in \mathbf{U} \setminus \{X \cup Y\}$ such that $X \perp Y|\mathbf{Z}$ [10]. An exhaustive search of $\mathbf{Z}$ is a combinatorial problem and can be intractable for high dimension data sets. To solve this problem parents and children learning algorithms generally search a candidate set for $\mathbf{PC}_T$ by starting with an empty set $\mathbf{PC}_T = \emptyset$ and iteratively adding to the current set $\mathbf{PC}_T$ the best candidate X such that there exists no set $\mathbf{Z} \subseteq \mathbf{PC}_T \setminus X$ such that $X \perp_P T|\mathbf{Z}$. Let's call this generic procedure *LearnPC*. It can be implemented in several ways. For instance, it is called *GetPCD* in [11] and *MMPC* in [20]. However, this rule actually leads

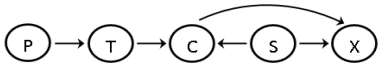

Fig. 1. Toy problem about PC learning: $\nexists \mathbf{Z} \in \mathbf{PC}_T$, so that, $X \perp_{\mathcal{G}} T|\mathbf{Z}$

to a superset of $\mathbf{PC}_T$ as noted in [11, 20]. Consequently, the algorithms must further process $\mathbf{PC}_T$ in order to search and eliminate false positives.

For sake of illustration, consider the example in Figure 1. Let T be the target and $\mathbf{U} = \{P, T, C, S, X\}$. The set of parents and children is clearly $\mathbf{PC}_T = \{P, C\}$. As we can see in Figure 1, $\nexists \mathbf{Z} \subseteq \mathbf{PC}_T \setminus X$ such that $T \perp_P X|\mathbf{Z}$. Therefore, to remove X, the output of *LearnPC*(T) must be further processed. Fortunately, owing to Theorem 1 [10], a false positive node X may be easily identified by testing whether $T \in$ *LearnPC*(X). In fact, $X \in \mathbf{PC}_T$ if and only if $[X \in$ *LearnPC*$(T)]$ & $[T \in$ *LearnPC*$(X)]$. This is the way divide-and-conquer methods correct the flaw discussed above. Clearly, for the boolean operator to return TRUE, both its operands should be TRUE. In practice, however, the procedure is conservative because makes it harder for a true positive node to enter the set $\mathbf{PC}_T$. Consequently, the procedure is highly sensitive to the false negative errors. This problem plagues divide-and-conquer methods for target variables with many adjacent nodes and relatively few instances. Loosely speaking, the larger $\mathbf{PC}_T$, the more likely it is to have false negative errors in the output of divide-and-conquer parents and children learning algorithms.

Theorem 1. *Let $\mathcal{G} = (V, E)$ be a DAG and $X, Y \in \boldsymbol{U}$. Then if X and Y are d-separated by some set, they are d-separated either by the set consisting of the parents of X or the set consisting of the parents of Y.*

The conservative procedure applied by divide-and-conquer methods is also a source of errors in the search process when their are some approximate deterministic relationships (see Definition 1) among the variables on the data set. For instance let us consider again Figure 1. Considering that the variables S and C are associated through a deterministic relationship, then clearly $C \perp T|S$. In this case divide-and-conquer methods will fail in finding the variable C as a child of the target T because the output of *LearnPC*(C) will contain only the variable S. Consequently, the necessarily boolean operator used by divide-and-conquer methods will exclude C from the set $\mathbf{PC}_T$ even if *LearnPC*(T) finds C as a parent or child of T. Deterministic relationships are a source of unfaithfulness, but a DAG $\mathcal{G}$ and a joint probability distribution P can still be faithful if some *approximate* deterministic relationships (ADR) are present in the data set. ADR can well lead to various errors in the search process as deterministic relationships do because conditional independence tests are highly unreliable in the presence of ADR. The existence of ADR in data is rather frequent. For instance, ADR are often present in survey data owing to hidden redundancies in

the questions [2, 17]. The exclusion of the obligation of applying the conservative boolean operator could help alleviating that problem.

Definition 1. *The association between the set of variables* $\mathbf{X}$ *and a target* T *is an approximate deterministic relationship if and only if the fraction of tuples that violate the deterministic dependency is at most equal to some threshold.*

Divide-and-conquer methods have tried to improve the *data-efficiency* of the learning process by reducing the size of conditioning sets in the independence tests. However, they are not always more *data-efficient* than incremental methods. For instance, Figure 2 shows a case where divide-and-conquer methods are even less *data-efficient* than incremental methods. As one can see in Figure 2 the maximum size of the conditioning set $\mathbf{Z}$ for a divide-and-conquer $LearnPC(T)$ will be 4, that is, $\mathbf{Z} = \{P, C, X, Y\}$, whilst for an incremental $LearnPC(T)$ it will be 3, that is, $\mathbf{Z} = \{P, C, S\}$.

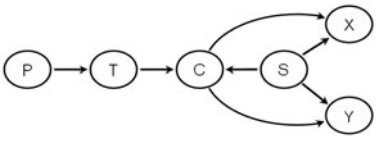

Fig. 2. Toy example where divide-and-conquer algorithms can be less *data-efficient* than incremental algorithms

4 The Hybrid Parents and Children Algorithm

In this section, we present the *Hybrid Parents and Children* (HPC) algorithm with the view to alleviate some of the shortcomings discussed previously. HPC combines characteristics from divide-and-conquer, incremental and ensemble methods in order to improve the accuracy of CB algorithms, specially when searching dense networks from data sets with relatively few instances. HPC (Algorithm 1) can be viewed as an ensemble method for combining many weak PC learners in an attempt to produce a stronger PC learner. HPC is based on three subroutines: *Data-Efficient Parents and Children Superset* (DE-PCS), *Data-Efficient Spouses Superset* (DE-SPS), and *Interleaved Incremental Association Parents and Children* (Inter-IAPC), a weak PC learner based on Inter-IAMB [19] that requires little computation. HPC may be thought of as a way to compensate for the large number of false negatives, at the output of the weak PC learner, by performing extra computations.

HPC receives a target node T, a data set $\mathcal{D}$ and a set of variables $\mathbf{U}$ as input and returns an estimation of $\mathbf{PC}_T$. It is hybrid in that it combines the benefits of incremental and divide-and-conquer methods. The procedure starts by extracting a superset $\mathbf{PCS}_T$ of $\mathbf{PC}_T$ (line 1) and a superset $\mathbf{SPS}_T$ of $\mathbf{SP}_T$

(line 2) with a severe restriction on the maximum conditioning size ($\mathbf{Z} <= 2$) in order to significantly increase the reliability of the tests. A first candidate PC set is then obtained by running the weak PC learner on $\mathbf{PCS}_T \cup \mathbf{SPS}_T$ (line 3). The key idea is the decentralized search at lines 4-8 that includes, in the candidate PC set, all variables in the superset $\mathbf{PCS}_T \cup \mathbf{SPS}_T$ that have T in their vicinity. Note that, in theory, X is in the output of Inter-IAPC(Y) if and only if Y is in the output of Inter-IAPC(X). However, in practice, this may not always be true, particularly when working in high-dimensional domains with relatively few instances. By loosening the criteria by which two nodes are said adjacent, the effective restrictions on the size of the neighborhood are now far less severe. The decentralized search has significant impact on the accuracy of HPC. It enables the algorithm to handle large neighborhoods while still being correct under the faithfulness condition. The proof of HPC's correctness is provided in Appendix A.

Algorithm 1. *HPC*

Require: T: target; $\mathcal{D}$: data set; $\mathbf{U}$: the set of variables
Ensure: $\mathbf{PC}_T$: Parents and Children of T

1: $[\mathbf{PCS}_T, \mathbf{dSep}] \leftarrow$ *DE-PCS*$(T, \mathcal{D})$
2: $\mathbf{SPS}_T \leftarrow$ *DE-SPS*$(T, \mathcal{D}, \mathbf{PCS}_T, \mathbf{dSep})$
3: $\mathbf{PC}_T \leftarrow$ *Inter-IAPC*$(T, \mathcal{D}(T \cup \mathbf{PCS}_T \cup \mathbf{SPS}_T))$
4: **for all** $X \in \mathbf{PCS}_T \setminus \mathbf{PC}_T$ **do**
5: **if** $T \in$ *Inter-IAPC*$(X, \mathcal{D}(T \cup \mathbf{PCS}_T \cup \mathbf{SPS}_T))$ **then**
6: $\mathbf{PC}_T \leftarrow \mathbf{PC}_T \cup X$
7: **end if**
8: **end for**

We now discuss the subroutines in more detail. Inter-IAPC (Algorithm 2) is a fast incremental method that receives a data set $\mathcal{D}$ and a target node T as its input and promptly returns a rough estimation of $\mathbf{PC}_T$, hence the term "weak" PC learner. Inter-IAPC is an straightforward extension of the algorithm Inter-IAMB [19]. Notice that neither MMPC [20] nor GetPC [11] should be used to implement this weak PC learner. The reason is that any break of symmetry of the PC relation in the output of these algorithms is an indication of a false negative member; the decentralized search would not aid in reducing the number of false negative variables at all. Inter-IAPC starts with a two-phase approach to infer $\mathbf{MB}_T$, that is, the Markov boundary of T. A growing phase attempts to iteratively add the best candidate variables to $\mathbf{MB}_T$, followed by a shrinking phase that attempts to remove as many irrelevant variables as possible, that is, the false negatives in the current set $\mathbf{MB}_T$. The function $dep(T, X|\mathbf{MB}_T)$ at line 4 returns a statistical estimation of the association between T and X given the current set $\mathbf{MB}_T$. The shrinking phase is interleaved with the growing phase. Interleaving the two phases allows to eliminate as soon as possible some of the false positives in the current Markov blanket as the algorithm progresses during the Markov boundary search. $\mathbf{PC}_T$ is obtained by removing the spouses of the

target from the final $\mathbf{MB}_T$ (lines 14-19). Inter-IAPC is very fast and sound (the proof of soundness is provided in Appendix A), despite its *data-inefficiency* in practice. The decentralized search in HPC is an attempt to alleviate this problem as discussed earlier.

Algorithm 2. *Inter-IAPC*

Require: T: target; D: data set; $\mathbf{U}$: set of variables;
Ensure: $\mathbf{PC}_T$: Parents and children of T;

1: $\mathbf{MB}_T \leftarrow \emptyset$
2: **repeat**
3: * *Add true positives to* $\mathbf{MB}_T$
4: $Y \leftarrow \mathbf{argmax}_{X \in (\mathbf{U} \setminus \mathbf{MB}_T \setminus T)} dep(T, X | \mathbf{MB}_T)$
5: **if** $T \not\perp Y | \mathbf{MB}_T$ **then**
6: $\mathbf{MB}_T \leftarrow \mathbf{MB}_T \cup Y$
7: **end if**

* *Remove false positives from* $\mathbf{MB}_T$
8: **for all** $X \in \mathbf{MB}_T$ **do**
9: **if** $T \perp X | (\mathbf{MB}_T \setminus X)$ **then**
10: $\mathbf{MB}_T \leftarrow \mathbf{MB}_T \setminus X$
11: **end if**
12: **end for**
13: **until** $\mathbf{MB}_T$ has not changed

* *Remove spouses of* T *from* $\mathbf{MB}_T$
14: $\mathbf{PC}_T \leftarrow \mathbf{MB}_T$
15: **for all** $X \in \mathbf{MB}_T$ **do**
16: **if** $\exists \mathbf{Z} \subseteq (\mathbf{MB}_T \setminus X)$ such that $T \perp X \mid \mathbf{Z}$ **then**
17: $\mathbf{PC}_T \leftarrow \mathbf{PC}_T \setminus X$
18: **end if**
19: **end for**

The subroutines DE-PCS (Algorithm 3) and DE-SPS (Algorithm 4) search a superset of $\mathbf{PC}_T$ and $\mathbf{SP}_T$ respectively with a severe restriction on the maximum conditioning size ($|\mathbf{Z}| <= 1$ in DE-PCS and $|\mathbf{Z}| <= 2$ in DE-SPS) in order to significantly increase the reliability of the tests. The variable filtering has two advantages : i) it allows HPC to scale to hundreds of thousands of variables by restricting the search to a subset of relevant variables, and ii) it eliminates many ADRs that produce many false negative errors in the output of the algorithm, as explained in the last section. DE-SPS works in two steps. First, a growing phase (lines 4-8) adds the variables that are d-separated from the target but still remain associated with the target when conditioned on another variable from $\mathbf{PCS}_T$. The shrinking phase (lines 9-16) discards irrelevant variables that are ancestors or descendants of a target's spouse. Pruning such irrelevant variables speeds up HPC.

Algorithm 3. *DE-PCS*

Require: T: target; $\mathcal{D}$: data set; $\mathbf{U}$: set of variables;
Ensure: $\mathbf{PCS}_T$: parents and children superset of T; **dSep**: d-separating sets;

Phase I: *Remove X if $T \perp X$*
PCS$_T \leftarrow \mathbf{U} \setminus T$
for all $X \in \mathbf{PCS}_T$ **do**
  if $(T \perp X)$ **then**
    PCS$_T \leftarrow \mathbf{PCS}_T \setminus X$
    dSep$(X) \leftarrow \emptyset$
  end if
end for

Phase II: *Remove X if $T \perp X|Y$*
for all $X \in \mathbf{PCS}_T$ **do**
  for all $Y \in \mathbf{PCS}_T \setminus X$ **do**
    if $(T \perp X \mid Y)$ **then**
      PCS$_T \leftarrow \mathbf{PCS}_T \setminus X$
      dSep$(X) \leftarrow Y$
      break loop FOR
    end if
  end for
end for

Algorithm 4. *DE-SPS*

Require: T: target; $\mathcal{D}$: data set; $\mathbf{U}$: the set of variables; $\mathbf{PCS}_T$: parents and children superset of T; **dSep**: d-separating sets;
Ensure: $\mathbf{SPS}_T$: Superset of the spouses of T;

SPS$_T \leftarrow \emptyset$
for all $X \in \mathbf{PCS}_T$ **do**
  SPS$_T^X \leftarrow \emptyset$
  for all $Y \in \mathbf{U} \setminus \{T \cup \mathbf{PCS}_T\}$ **do**
    if $(T \not\perp Y|\mathbf{dSep}(Y) \cup X)$ **then**
      SPS$_T^X \leftarrow \mathbf{SPS}_T^X \cup Y$
    end if
  end for
  for all $Y \in \mathbf{SPS}_T^X$ **do**
    for all $Z \in \mathbf{SPS}_T^X \setminus Y$ **do**
      if $(T \perp Y|X \cup Z)$ **then**
        SPS$_T^X \leftarrow \mathbf{SPS}_T^X \setminus Y$
        break loop FOR
      end if
    end for
  end for
  SPS$_T \leftarrow \mathbf{SPS}_T \cup \mathbf{SPS}_T^X$
end for

5 Experimental Validation

In this section, we assess the accuracy and the scalability of HPC through several empirical experiments. We first compared HPC with two distinguished CB algorithm proposals that appeared recently in the literature, namely MMPC[2] (Max-Min Parents and Children) [20] and GetPC[3] [11]. Both algorithms are correct under the faithfulness condition and are also scalable to high dimensional data sets. Only the authors' own implementations were used for the empirical experiments. HPC was also compared to Inter-IAPC (Algorithm 2), the weak PC learner. The same critical p-value ($\alpha = 0.05$) was used to make fair comparisons. We also compared HPC with two well known search-and-score global network learning algorithms, namely Greedy Equivalent Search (GES) [6] and SCA (Sparse Candidate Algorithm) [7]. For SCA, we used the implementation available in the Causal Explorer system [1]. The code uses the Bayesian scoring heuristic and an equivalent sample size of 10. The maximum allowed size for the candidate parents' sets k is set to k=10. For GES, we used the command-line tool in the WinMine Toolkit[4]. All the data sets used for the empirical experiments presented in this section were sampled from well-known BNs that have been previously used as benchmarks for BN learning algorithms, namely *Asia*, *Alarm*, *Barley*, *Child*, *Genes*, *Insulin*, *Insurance*, *Link*, *Mildew*, *Pigs* and *Carpa* (see [20] for details). Three samples sizes have been considered: 200, 500 and 1500. We do not claim that those data sets resemble real-world problems, however, they make it possible to compare the outputs of the algorithms with the known $\mathbf{PC}_T$ set of the BNs sampled.

5.1 Accuracy

Each CB algorithm was run 10 times on each node of each benchmark. The variables in the output of the algorithms were compared against the true neighbors. As GES and SCA are global network learning algorithms, they were run 10 times on each benchmark. To evaluate the accuracy, we combined precision (i.e., the number of true positives in the output divided by the number of nodes in the output) and recall (i.e., the number of true positives divided the true size of the PC) as $\sqrt{(1-precision)^2 + (1-recall)^2}$, to measure the Euclidean distance from perfect precision and recall, as proposed in [11]. Figure 3 plots the Euclidean distance, averaged over all nodes, as a function of the PC size. The advantage of *HPC* against the other algorithms is clearly noticeable.

5.2 Scalability

We now provide an empirical evaluation of the scalability of HPC on real and artificial high-dimensional data sets. First, the five benchmarks were replicated

[2] `http://discover1.mc.vanderbilt.edu/discover/public`
[3] `http://www.ida.liu.se/~jospe`
[4] WinMine Toolkit can be downloaded at `http://research.microsoft.com/en-us/um/people/dmax/WinMIne/Download.html`

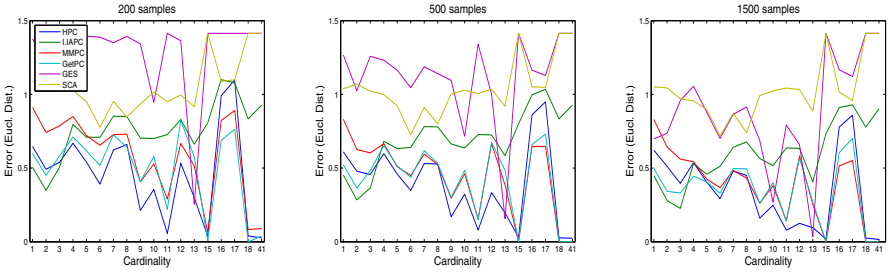

Fig. 3. Euclidean distance from perfect precision and recall versus the cardinality of the nodes degree. All algorithms were run 10 times on each variable of each of the 11 benchmark.

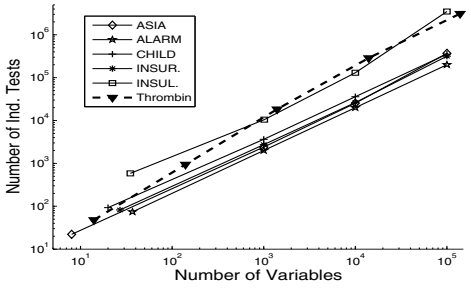

Fig. 4. Number of calls to the conditional independence tests on Thrombin database (dotted line) and on synthetic data sets (plain lines) from 5 benchmarks replicated several times, versus the number of variables.

several times (up to 100 000 variables) to increase artificially the number of variables. Each network is obtained by tiling several copies of the original network. The tiling is performed by maintaining the structural and probabilistic properties of the original network in the tiled network. The learning task is the same as in the previous subsection, but instead of the accuracy, we report in Figure 4 (in plain lines) the average number of conditional independence tests that were conducted for each tiled network (in log-log scale) as a function of the number of variables. The average value is estimated over 100 data sets with only 100 instances for each network size. The number of calls to the statistical test was empirically on the order $O(n^{1.09})$ where n is the number of variables, regardless of the size of the neighborhood of the target in the corresponding BN.

The scalability claims were based on taking a basic graph and creating a larger graph by tiling the smaller graph repeatedly. One could claim that the empirical number of $O(n^{1.09})$ may be optimistic as it hinges on our tilling construction. Running HPC on increasing parts of a real-world database would lend more validity to our statement. To this aim, we considered the Thrombin database which was provided by DuPont Pharmaceuticals for KDD Cup 2001. It is exemplary of a real drug design [5]. The training set contains 1909 instances characterized by

139,351 binary features. Each instance represents a drug compound tested for its ability to bind to a target site on Thrombin, a key receptor in blood clotting. Each compound is labeled with one out of two classes, either it binds or not. The task is again to learn the PC set of the target variable from 1909 given compounds (the learning data). We picked at random 0.01% 0.1%, 1%, 10% and the full set of variables including the target and used these subsets to learn the target neighborhood. The overall process was repeated 100 times by bootstrap. As done before we measured the average number of conditional independence tests as a function of the number of variables. Here again, the number of calls to the test varies by a multiplicative factor when the size of the neighborhood is varied. The results show that the number of calls to the statistical test is empirically on the order $O(n^{1.21})$, that is, slightly superior to the behavior on synthetic data. Nonetheless, the almost-linear time complexity of HPC shows promise for a variety of applications involving hundreds of thousands of variables.

6 Conclusion

We discussed a novel scalable algorithm for local BN structure learning, called *Hybrid Parents and Children* (HPC). Extensive simulations have been conducted on public synthetic and real-world data sets of various sample sizes to assess its accuracy and scalability. Significant improvements in accuracy were obtained in all experiments compared to state-of-the-art algorithms. The *effective* complexity is only $O(n^{1.09})$ in practice on the five benchmarks that we considered and $O(n^{1.21})$ on the real-world Thrombin database. The moderate time complexity of HPC shows great promise for a variety of applications involving hundreds of thousands of variables. The HPC algorithm was designed for detecting the parents and children of a vertex in a graphical model. Learning the parents and children of a target is a key routine used in constraint-based BN structure learning algorithms. HPC can be applied iteratively to find the Markov boundary of a target for classification purposes, the local or the whole skeleton of the BN although this is not discussed here for the sake of conciseness. Regarding causal inference, HPC assumes no hidden common causes, which is pretty unrealistic. On the other hand, it could easily be modified to detect hidden common causes [3, 13]. These adaptations of HPC are left for future work.

References

1. Aliferis, C., Tsamardinos, I., Statnikov, A., Brown, L.: Causal explorer: A causal probabilistic network learning toolkit for biomedical discovery. In: Proceedings of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Scienes, METMBS, Las Vegas, Nevada, USA, pp. 23–26. CSREA Press (2003)
2. Aussem, A., de Morais, S.R., Corbex, M.: Nasopharyngeal carcinoma data analysis with a novel Bayesian network skeleton learning. In: Bellazzi, R., Abu-Hanna, A., Hunter, J. (eds.) AIME 2007. LNCS (LNAI), vol. 4594, pp. 326–330. Springer, Heidelberg (2007)

3. Brown, L.E., Tsamardinos, I.: A strategy for making predictions under manipulation. In: JMLR: Workshop and Conference Proceedings, vol. 3, pp. 35–52 (2008)
4. Cawley, G.: Causal and non-causal feature selection for ridge regression. In: JMLR: Workshop and Conference Proceedings, vol. 3 (2008)
5. Cheng, J., Hatzis, C., Hayashi, H., Krogel, M.A., Morishita, S., Page, D., Sese, J.: KDD Cup 2001 Report. In: ACM SIGKDD Explorations, pp. 1–18 (2002)
6. Chickering, D.: Learning equivalence classes of bayesian-network structures. Machine Learning 2, 445–498 (2002)
7. Friedman, N.L., Nachman, I., Pe'er, D.: Learning bayesian network structure from massive datasets: the "sparse candidate" algorithm. In: Laskey, K.B., Prade, H. (eds.) Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, pp. 21–30. Morgan Kaufmann Publishers, San Francisco (1999)
8. Fu, S., Desmarais, M.: Tradeoff analysis of different Markov blanket local learning approaches. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 562–571. Springer, Heidelberg (2008)
9. Guyon, I., Aliferis, C., Cooper, G., Elisseef, A., Pellet, J.P., Statnikov, P.A.: Design and analysis of the causation and prediction challenge. In: JMLR: Workshop and Conference Proceedings, vol. 1, pp. 1–16. MIT Press, Boston (2008)
10. Neapolitan, R.E.: Learning Bayesian Networks. Pearson Prentice Hall, Upper Saddle River (2004)
11. Peña, J.M., Nilsson, R., Björkegren, J., Tegnér, J.: Towards scalable and data efficient learning of Markov boundaries. International Journal of Approximate Reasoning 45(2), 211–232 (2007)
12. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco (1988)
13. Pearl, J.: Causality: Models, Reasoning, and Inference. Cambridge University Press, Cambridge (2000)
14. Peña, J.: Learning gaussian graphical models of gene networks with false discovery rate control. In: Marchiori, E., Moore, J.H. (eds.) EvoBIO 2008. LNCS, vol. 4973, pp. 165–176. Springer, Heidelberg (2008)
15. Rodrigues de Morais, S., Aussem, A.: A novel scalable and data efficient feature subset selection algorithm. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 298–312. Springer, Heidelberg (2008)
16. Rodrigues de Morais, S., Aussem, A.: A novel Markov boundary based feature subset selection algorithm. Neurocomputing 73, 578–584 (2010)
17. Rodrigues de Morais, S., Aussem, A., Corbex, M.: Handling almost-deterministic relationships in constraint-based Bayesian network discovery: Application to cancer risk factor identification. In: 16th European Symposium on Artificial Neural Networks ESANN'08, pp. 101–106 (2008)
18. Steck, H.: Learning the Bayesian network structure: Dirichlet prior vs data. In: Conference on Uncertainty in Artificial Intelligence UAI'08, pp. 511–518 (2008)
19. Tsamardinos, I., Aliferis, C.F., Statnikov, A.R.: Algorithms for large scale Markov blanket discovery. In: Florida Artificial Intelligence Research Society Conference FLAIRS'03, pp. 376–381 (2003)
20. Tsamardinos, I., Brown, L.E., Aliferis, C.F.: The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning 65(1), 31–78 (2006)
21. Tsamardinos, I., Brown, L.E.: Bounding the false discovery rate in local Bayesian network learning. In: Proceedings AAAI National Conference on AI AAAI'08, pp. 1100–1105 (2008)

A Proof of Correctness

A structure learning algorithm from data is said to be correct (or sound) if it returns the correct DAG pattern (or a DAG in the correct equivalence class) under the assumptions that the independence tests are reliable and that the learning data set is a sample from a distribution P faithful to a DAG $\mathcal{G}$. The (ideal) assumption that the independence tests are reliable means that they decide (in)dependence if and only if the (in)dependence holds in P. Consequently, a parents and children learning algorithm is said to be correct (or sound) when under the assumptions that the independence tests are reliable and that the learning data set is a sample from a distribution P faithful to a DAG $\mathcal{G}$, the algorithm returns the correct set of parents and children of the target, that is, the target direct neighborhood. Correctness is a desirable asymptotic property though the underlying assumptions may not hold in practice. In general, we would want an edge to mean a direct dependence. Several definitions and intermediate theorems are required before we demonstrate *HPC*'s correctness under faithfulness condition.

Definition 2. *A Markov blanket $\boldsymbol{M}_T$ of T is any set of variables such that T is conditionally independent of all the remaining variables given $\boldsymbol{M}_T$. A Markov boundary, $\boldsymbol{MB}_T$, of T is any Markov blanket such that none of its proper subsets is a Markov blanket of T.*

Theorem 2. *Suppose $< \mathcal{G}, P >$ satisfies the faithfulness condition. Then for each variable X, the set of parents, children of X, and parents of children (spouses) of X is its unique Markov boundary.*

A proof can be found in [10]. Indeed, as $\mathbf{PCS}_T \cup \mathbf{SPS}_T$ is a subset of $\mathbf{U}$, a difficulty arises: a marginal distribution $P^{\mathbf{V}}$ of $\mathbf{V} \subset \mathbf{U}$ may not satisfy the faithfulness condition with any DAG even if $P^{\mathbf{U}}$ does. This is an example of embedded faithfulness, which is defined as follow:

Definition 3. *Let $P^{\boldsymbol{V}}$ be a distribution of the variables in $\boldsymbol{V}$ where $\boldsymbol{V} \subset \boldsymbol{U}$ and let $\mathcal{G} =< \mathbf{U}, \mathbf{E} >$ be a DAG. $< \mathcal{G}, P^{\boldsymbol{V}} >$ satisfies the embedded faithfulness condition if $\mathcal{G}$ entails all and only the conditional independencies in $P^{\boldsymbol{V}}$, for subsets including only elements of $\boldsymbol{V}$.*

We obtain embedded faithfulness by taking the marginal of a faithful distribution as shown by the next theorem:

Theorem 3. *Let $P^{\boldsymbol{U}}$ be a joint probability of the variables in $\boldsymbol{U}$ with $\boldsymbol{V} \subseteq \boldsymbol{U}$ and $\mathcal{G} =< \mathbf{U}, \mathbf{E} >$. If $< \mathcal{G}, P^{\boldsymbol{U}} >$ satisfies the faithfulness condition and $P^{\boldsymbol{V}}$ is the marginal distribution of $\boldsymbol{V}$, then $< \mathcal{G}, P^{\boldsymbol{V}} >$ satisfies the* embedded *faithful condition.*

The proof can be found in [10]. Note that not every distribution does admit an embedded faithful representation. This property is useful to prove the correctness of *HPC* under the faithfulness condition. Let $\mathbf{PC}_X^{\mathbf{U}}$ denote the variables $Y \in \mathbf{U}$

so that there is no set $\mathbf{Z} \subseteq \mathbf{U} \setminus \{X, Y\}$ such that $X \perp_P Y | \mathbf{Z}$. If $< \mathcal{G}, P^{\mathbf{U}} >$ satisfies the faithfulness condition, $\mathbf{PC}_X^{\mathbf{U}}$ are the parents and children of X in $\mathbf{U}$. In any case, $\mathbf{PC}_X^{\mathbf{U}}$ is the unique set of the variables Y_i that remain dependent on X conditioned on any set $\mathbf{Z} \subseteq \mathbf{U} \setminus \{X, Y_i\}$.

Theorem 4. *Let $\boldsymbol{U}$ be a set of random variables and $\mathcal{G} =< \mathbf{U}, \mathbf{E} >$ be a DAG. If $< \mathcal{G}, P^{\boldsymbol{U}} >$ satisfies the faithfulness condition, then every target T admits a unique Markov boundary $\boldsymbol{MB}_T^{\boldsymbol{U}}$. Moreover, for all $\boldsymbol{V}$ such that $\boldsymbol{MB}_T^{\boldsymbol{U}} \subseteq \boldsymbol{V} \subseteq \boldsymbol{U}$, T admits a unique Markov boundary over $\boldsymbol{V}$ and $\boldsymbol{MB}_T^{\boldsymbol{V}} = \boldsymbol{MB}_T^{\boldsymbol{U}}$.*

Proof: If $\mathbf{MB}_T^{\mathbf{U}}$ is the Markov boundary of T in $\mathbf{U}$, then T is independent of all variable $Y \in [\mathbf{V} \setminus (\mathbf{MB}_T^{\mathbf{U}} \cup T)]$ conditionally on $\mathbf{MB}_T^{\mathbf{U}}$, then $\mathbf{MB}_T^{\mathbf{U}}$ is a Markov blanket in $\mathbf{V}$. Moreover, none of the proper subsets of $\mathbf{MB}_T^{\mathbf{U}}$ is a Markov blanket of T in $\mathbf{V}$, so $\mathbf{MB}_T^{\mathbf{U}}$ is also a Markov boundary of T in $\mathbf{V}$. So if it is not the unique MB for T in $\mathbf{V}$ there exists some other set $\mathbf{S}_T$ not equal to $\mathbf{MB}_T^{\mathbf{U}}$, which is a MB of T in $\mathbf{V}$. Since $\mathbf{MB}_T^{\mathbf{U}} \neq \mathbf{S}_T$ and $\mathbf{MB}_T^{\mathbf{U}}$ cannot be a subset of $\mathbf{S}_T$, there is some $X \in \mathbf{MB}_T^{\mathbf{U}}$ such that $X \notin \mathbf{S}_T$. Since $\mathbf{S}_T$ is a MB for T, we would have $T \perp_P X | \mathbf{S}_T$. If X is a parent or child of T, we would not have $T \perp_{\mathcal{G}} X | \mathbf{S}_T$ which means we would have a conditional independence that is not entailed by d-separation in $\mathcal{G}$, which contradicts the faithfulness condition. If X is a parent of a child of T in $\mathcal{G}$, let Y be their common child in $\mathbf{U}$. If $Y \in \mathbf{S}_T$ we again would not have $T \perp_{\mathcal{G}} X | \mathbf{S}_T$. If $Y \notin \mathbf{S}_T$ we would have $T \perp_P Y | \mathbf{S}_T$ because $\mathbf{S}_T$ is a MB of T in $\mathbf{V}$ but we do not have $T \perp_{\mathcal{G}} Y | \mathbf{S}_T$ because T is a parent of Y in G. So again we would have a conditional independence which is not a d-separation in $\mathcal{G}$. This proves that there can not be such set $\mathbf{S}_T$. □

Theorem 5. *Let $\boldsymbol{U}$ be a set of random variables and T a target variable. Let $\mathcal{G} =< \mathbf{U}, \mathbf{E} >$ be a DAG such that $< \mathcal{G}, P^{\boldsymbol{U}} >$ satisfies the faithfulness condition. Let $\boldsymbol{V}$ be such that $\boldsymbol{MB}_T^{\boldsymbol{U}} \subseteq \boldsymbol{V} \subseteq \boldsymbol{U}$ then, $\boldsymbol{PC}_T^{\boldsymbol{V}} = \boldsymbol{PC}_T^{\boldsymbol{U}}$.*

Proof: Clearly $\mathbf{PC}_T^{\mathbf{U}} \subseteq \mathbf{PC}_T^{\mathbf{V}}$ as $\mathbf{MB}_T^{\mathbf{U}} \subseteq \mathbf{V} \subseteq \mathbf{U}$. If $X \in \mathbf{PC}_T^{\mathbf{V}}$ and $X \notin \mathbf{PC}_T^{\mathbf{U}}$, $\exists \mathbf{Z} \subseteq \mathbf{MB}_T^{\mathbf{U}} \setminus X$ such that $T \perp_P X | \mathbf{Z}$ because all non adjacent nodes may be d-separated in $\mathcal{G}$ by a subset of its Markov boundary. As $\mathbf{MB}_T^{\mathbf{U}} = \mathbf{MB}_T^{\mathbf{V}}$ owing to Theorem 4, so X and T can be d-separated in $\mathbf{V} \setminus \{X, T\}$. Therefore, X cannot be adjacent to T in $\mathbf{V}$. □

Theorem 6. *Let $\boldsymbol{U}$ be a set of random variables and T a target variable. Let $\mathcal{G} =< \mathbf{U}, \mathbf{E} >$ be a DAG such that $< \mathcal{G}, P^{\boldsymbol{U}} >$ satisfies the faithfulness condition. Let $\boldsymbol{V}$ be such that $\boldsymbol{MB}_T^{\boldsymbol{U}} \subseteq \boldsymbol{V} \subseteq \boldsymbol{U}$. Under the assumption that the independence tests are reliable, Inter-IAPC$(T, \mathcal{D}, \boldsymbol{V})$ returns $\boldsymbol{PC}_T^{\boldsymbol{U}}$. Moreover, let $X \in \boldsymbol{V} \setminus T$, then $\forall T \in \boldsymbol{V}$, T is in the output of Inter-IAPC$(X, \mathcal{D}, \boldsymbol{V})$ iff $X \in \boldsymbol{PC}_T^{\boldsymbol{U}}$.*

Proof: We prove first that *Inter-IAPC*$(T, \mathcal{D}, \mathbf{V})$ returns $\mathbf{PC}_T^{\mathbf{U}}$. In lines 1-13, *Inter-IAPC* seeks a minimal set $\mathbf{S}_T \subseteq \mathbf{V} \setminus T$ that renders $\mathbf{V} \setminus \mathbf{S}_T$ independent of T conditionally on $\mathbf{S}_T$. This set is unique owing to Theorem 4, therefore $\mathbf{S}_T = \mathbf{MB}_T^{\mathbf{V}} = \mathbf{MB}_T^{\mathbf{U}}$. In the backward phase, *Inter-IAPC* removes the variables $X \in \mathbf{MB}_T^{\mathbf{V}}$ such that $\exists \mathbf{Z} \subseteq (\mathbf{MB}_T^{\mathbf{V}} \setminus X)$ for which $T \perp X \mid \mathbf{Z}$. These variables are the spouses of T in $\mathcal{G}$, so *Inter-IAPC*$(T, \mathcal{D}, \mathbf{V})$ returns $\mathbf{PC}_T^{\mathbf{U}}$. Now, if $X \notin \mathbf{PC}_T^{\mathbf{U}}$

then $X \notin \mathbf{PC}_T^{\mathbf{V}}$ owing to Theorem 5. So there is a set $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$ such that $T \perp X \mid \mathbf{Z}$. Therefore, X cannot be in the output of *Inter-IAPC*$(T, \mathcal{D}, \mathbf{V})$, nor T can be in the output of *Inter-IAPC*$(X, \mathcal{D}, \mathbf{V})$. □

Theorem 7. *Under the assumptions that the independence tests are reliable and that the data set is a sample from a probability distribution $P^{\boldsymbol{U}}$ faithful to a DAG $\mathcal{G}$, then HPC$(T, \mathcal{D}, \boldsymbol{U})$ returns $\boldsymbol{PC}_T^{\boldsymbol{U}}$.*

Proof: Let $\mathbf{V} = (\mathbf{PCS} \cup \mathbf{SPS})$, then $\mathbf{V}$ is a superset of $\mathbf{MB}_T^{\mathbf{U}}$. Based on what is stated by Theorem 4 we know that $\mathbf{MB}_T^{\mathbf{V}} = \mathbf{MB}_T^{\mathbf{U}}$. If T is in the output of *Inter-IAPC*$(X, \mathbf{V}, \mathcal{D})$ then X should be in the output of *Inter-IAPC*$(T, \mathbf{V}, \mathcal{D})$ owing to Theorem 6. So *HPC* returns $\mathbf{PC}_T^{\mathbf{U}}$. □