A Comparison of Novel and State-of-the-Art Polynomial Bayesian Network Learning Algorithms

Laura E. Brown **Ioannis Tsamardinos** **Constantin F. Aliferis**
Vanderbilt University, Department of Biomedical Informatics
2209 Garland Avenue, Nashville, TN 37232-8340
laura.e.brown, ioannis.tsamardinos, constantin.aliferis@vanderbilt.edu

Abstract

Learning the most probable a posteriori Bayesian network from data has been shown to be an NP-Hard problem and typical state-of-the-art algorithms are exponential in the worst case. However, an important open problem in the field is to identify the least restrictive set of assumptions and corresponding algorithms under which learning the optimal network becomes polynomial. In this paper, we present a technique for learning the skeleton of a Bayesian network, called Polynomial Max-Min Skeleton (*PMMS*), and compare it with Three Phase Dependency Analysis, another state-of-the-art polynomial algorithm. This analysis considers both the theoretical and empirical differences between the two algorithms, and demonstrates *PMMS*'s advantages in both respects. When extended with a greedy hill-climbing Bayesian-scoring search to orient the edges, the novel algorithm proved more time efficient, scalable, and accurate in quality of reconstruction than most state-of-the-art Bayesian network learning algorithms. The results show promise of the existence of polynomial algorithms that are provably correct under minimal distributional assumptions.

Introduction

The problem of learning the most probable a posteriori Bayesian network (BN) from data under certain broad conditions is worst-case *NP*-hard (Chickering, Meek, & Heckerman 2003). Given the recent emergence, particularly in biology and medicine, of very high dimensional datasets with tens of thousands of variables, an important open research question is whether it is possible to design polynomial learning algorithms – with well-understood theoretical properties – that are not only time-efficient, but also exhibit reasonable average performance in terms of network reconstruction.

A first step towards answering this questions was the Three Phase Dependency Analysis (*TPDA*) algorithm (Cheng *et al.* 2002). The algorithm runs in polynomial time and correctly identifies the data-generating network, provided the data distribution is *monotone DAG-faithful*. Intuitively, the monotone DAG-faithful assumption requires that the conditional mutual information of two variables be a monotonic function of the "active paths" between the variables in the network structure: the more active paths open by a conditioning set, the greater the mutual information between the variables should be. Compared to other constraint-based learning algorithms, such as the *PC* (Spirtes, Glymour, & Scheines 2000), *TPDA* uses not only the binary (yes/no) results of tests of conditional independence between two variables, but also the strength of association as indicated by the conditional mutual information. This additional piece of information, in conjunction with the monotone faithfulness assumption, is what enables the algorithm to become polynomial.

One way to formally define monotone DAG-faithfulness was provided by Chickering & Meek (2003). Given their definition, the authors show that this assumption highly restricts the structure of the BNs that adhere to the condition. We note, however, that it may still be possible to identify other formulations of the assumption that are not as restrictive.

In this paper we present a new polynomial algorithm, known as Polynomial Max-Min Skeleton (*PMMS*), for learning the BN skeleton, i.e., the graph of the BN without regard to the direction of the edges. In certain cases, *PMMS* can correctly reconstruct skeletons of networks that are not monotone DAG-faithful and in which *TPDA* does not return the proper structure. In addition, *PMMS* employs a different search strategy for identifying d-separating subsets that exhibits better sample utilization. In an extensive empirical evaluation we compared *PMMS* with other constraint-based network learning algorithms on the task of reconstructing the network skeleton. *PMMS* proves to be a good trade-off between time and quality of reconstruction compared to all algorithms, and in general outperforms *TPDA* particularly for the smaller sample sizes.

To compare *PMMS* against search-and-score Bayesian network learning algorithms (i.e., including the edge orientations), we extended the algorithm with a greedy hill-climbing Bayesian-scoring search augmented with a TABU list (as in Friedman, Nachman, & Pe'er 1999) to orient the edges. The resulting algorithm is more time-efficient than its non-polynomial counterpart without sacrificing quality of reconstruction. In addition, this algorithm outperformed most other state-of-the-art Bayesian network learning algorithms. The evaluation points to the existence of polynomial algorithms that are provably correct under minimal distributional assumptions.

Background

We will denote the conditional independence of X and Y given $\mathbf{Z}$ in some distribution P with $Ind(X;Y|\mathbf{Z})$, dependence as $Dep(X;Y|\mathbf{Z})$, and the conditional mutual information used as a measure of association as $Inf(X;Y|\mathbf{Z})$ (the reference distribution P can be inferred by the context).

A Bayesian network (BN) is a tuple $\mathcal{N} = \langle \mathcal{G}, P \rangle$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$is a directed acyclic graph (DAG) with nodes representing the random variables $\mathcal{V}$ and P a joint probability distribution on $\mathcal{V}$. In addition, $\mathcal{G}$ and P must satisfy the Markov condition: every variable, $X \in \mathcal{V}$, is independent of any subset of its non-descendant variables conditioned on the set of its parents (Pearl 1988).

A graphical criterion called d-separation is useful for characterizing conditional independencies (Pearl 1988). First, we define the terms "collider" and "blocked path" to aid in the definition of d-separation. A *collider* is defined as a node W on path p if p contains two incoming edges into W, e.g., W is a collider in the chain $X \longrightarrow W \longleftarrow Y$. A path p from node X to node Y is *blocked* by a set of nodes $\mathbf{Z}$, if there is a node W on p for which one of the following two conditions hold: (1) W is not a collider and $W \in \mathbf{Z}$, or (2) W is a collider and none of W or its descendants are in $\mathbf{Z}$. A path that is not blocked is termed *open* or *active*. Two nodes X and Y are *d-separated* by $\mathbf{Z}$ in graph $\mathcal{G}$, denoted as $Dsep_{\mathcal{G}}(X;Y|\mathbf{Z})$, if and only if every path from X to Y is blocked by $\mathbf{Z}$ (Pearl 1988). Two nodes are d-connected if they are not d-separated. For a BN $\mathcal{N} = \langle \mathcal{G}, P \rangle$, $Dsep_{\mathcal{G}}(\mathbf{X};\mathbf{Y}|\mathbf{Z}) \Rightarrow Ind(\mathbf{X};\mathbf{Y}|\mathbf{Z})$ (Pearl 1988).

A BN $\mathcal{N} = \langle \mathcal{G}, P \rangle$ satisfies the *faithfulness condition* (called faithful network) if the Markov Condition applied on $\mathcal{G}$ entails all and only the conditional independencies in P (Spirtes, Glymour, & Scheines 2000). We will call a distribution P for which there exists a faithful network $\langle \mathcal{G}, P \rangle$, a faithful distribution and $\mathcal{G}$ a *perfect map* of P (Pearl 1988). In a faithful BN $\langle \mathcal{G}, P \rangle$ $Dsep_{\mathcal{G}}(X;Y|\mathbf{Z}) \Leftrightarrow Ind(X;Y|\mathbf{Z})$ (Pearl 1988).

It can be proven that in a faithful BN, an edge between X and Y exists if and only if there is no d-separating set $\mathbf{Z}$ such that $Ind(X;Y|\mathbf{Z})$ (Spirtes, Glymour, & Scheines 2000). Algorithms following the constraint-based approach in BN learning (Spirtes, Glymour, & Scheines 2000; Cheng *et al.* 2002) estimate from data the conditional independencies and return only the edges which satisfy the above condition. These algorithms differ in the strategies for searching for such d-separating sets.

The Polynomial Max-Min Skeleton

We will denote with $\mathbf{PC}_T^{\mathcal{G}}$ the parents and children of T in the BN $\langle \mathcal{G}, P \rangle$, i.e., all nodes with an edge to and from T. This set is unique for all $\mathcal{G}$, such that $\langle \mathcal{G}, P \rangle$ is a faithful Bayesian network to the same distribution P (Pearl 1988) and so we will drop the superscript $\mathcal{G}$ when the distribution P can be inferred by the context.

We define the minimum association of X and T relative to a feature subset $\mathbf{Z}$, denoted as $\textsc{MinAssoc}(X;T|\mathbf{Z})$, as

$$\textsc{MinAssoc}(X;T|\mathbf{Z}) = \min_{\mathbf{S} \subseteq \mathbf{Z}} Assoc(X;T|\mathbf{S})$$

Algorithm 1 *PMMPC* Algorithm

```
 1: procedure PMMPC (T,$\mathcal{D}$)
        %Phase I: Forward
 2:     CPC = $\emptyset$
 3:     repeat
 4:         $F = \arg\max_{X \in \mathcal{V}} \textsc{GreedyMinAssoc}(X; T;$
                $\mathbf{CPC}; Assoc(X;T|\emptyset); \emptyset)$
 5:         $assoc = \max_{X \in \mathcal{V}} \textsc{GreedyMinAssoc}(X; T;$
                $\mathbf{CPC}; Assoc(X;T|\emptyset); \emptyset)$
 6:         if $assoc \neq 0$ then
 7:             CPC = CPC $\cup F$
 8:     until CPC has not changed
        %Phase II: Backward
 9:     for all $X \in$ CPC do
10:         if $\textsc{GreedyMinAssoc}(X; T;$ CPC;
                $Assoc(X;T|\emptyset); \emptyset) = 0$ then
11:             CPC = CPC $\setminus \{X\}$
12:     return CPC
13: end procedure

14: function GreedyMinAssoc( $X, T, \mathbf{Z}, minval, minarg$)
15:     $min = \min_{S \in \mathbf{Z}} Assoc(X;T|minarg \cup \{S\})$
16:     $arg = \arg\min_{S \in \mathbf{Z}} Assoc(X;T|minarg \cup \{S\})$
17:     if (($min < minval$) AND ($\mathbf{Z} \setminus minarg \neq \emptyset$)) then
18:         $minval = \textsc{GreedyMinAssoc}(X; T;$
                $\mathbf{Z} \setminus minarg; min; minarg \cup \{arg\})$
19:     return $minval$
20: end function
```

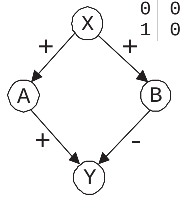
i.e., as the minimum association achieved between X and T over all subsets of $\mathbf{Z}$. $Assoc(X;T|\mathbf{S})$, the association between two variables given a conditioning set, can implemented with a number of statistical or information theoretic measures of association, e.g., by the conditional mutual information $Inf(X;T|\mathbf{S})$. In our implementation we prefer a statistically oriented test and we use the negative p-value returned by the χ^2 test of independence $Ind(X;T|\mathbf{S})$ (the smaller the p-value, the higher we consider the association between X and T) as in Spirtes, Glymour, & Scheines (2000). A requirement for $Assoc(X;T|\mathbf{S})$ is to return zero when $Ind(X;T|\mathbf{S})$.

The Polynomial Max-Min Skeleton algorithm (*PMMS*) is run on a dataset and returns the skeleton network. *PMMS* works by calling the Polynomial Max-Min Parents and Children algorithm (*PMMPC*) for each variable. *PMMPC* identifies an approximation of $\mathbf{PC}_T$ given a target node, T, and the data. Once the parents and children set has been discovered for each node, *PMMS* pieces together the identified edges into the network skeleton.

PMMPC$(T, \mathcal{D})$, the main subroutine of the *PMMS* algorithm, discovers the $\mathbf{PC}_T$ using a two-phase scheme (shown in Algorithm 1)[1].

In phase I, the forward phase, variables sequentially enter a candidate parents and children set of T, denoted as **CPC**, by use of the $\textsc{Max-Min Heuristic}$:

[1] *PMMPC* is a polynomial variant of the Max-Min Parents and Children (*MMPC*) algorithm (Tsamardinos, Aliferis, & Statnikov 2003). The *MMPC* algorithm can be created from Algorithm 1 by replacing the calls to $\textsc{GreedyMinAssoc}$ at lines 4, 5, and 10 with the function $\textsc{MinAssoc}$.



X	P(X)
0	0.5
1	0.5

A	X	P(A|X)
0	0	0.8
1	0	0.2
0	1	0.2
1	1	0.8

B	X	P(B|X)
0	0	0.8
1	0	0.2
0	1	0.2
1	1	0.8

Y	A	B	P(Y|A,B)
0	0	0	0.3
0	1	0	0.1
0	0	1	0.95
0	1	1	0.75
1	0	0	0.7
1	1	0	0.9
1	0	1	0.05
1	1	1	0.25

Figure 1: A network DAG and probabilities tables.

Select the variable that *maximizes the minimum association* with T relative to **CPC** (hence the name of the algorithm).

The heuristic is admissible in the sense that all variables with an edge to or from T and possibly more will eventually enter **CPC**. The intuitive justification for the heuristic is to select the variable that remains highly associated with T despite our best efforts to make the variable independent of T. Phase I stops when all remaining variables are independent of the target T given some subset of **CPC**, i.e., when the maximum minimum association reaches zero.

Conditioning on all subsets of **CPC** to identify the $\textsc{MinAssoc}(X; T|\mathbf{CPC})$ requires an exponential number of calls to $Assoc$. In this polynomial version of the algorithm, *PMMPC*, we greedily search for the subset $\mathbf{S} \subseteq \mathbf{CPC}$ that achieves the minimum association between X and T conditioned over the subsets of **CPC**. The function $\textsc{GreedyMinAssoc}(X, T, \mathbf{CPC}, minval, minarg)$ starts with $minval$ as the current estimate of the minimum and $minarg$ as the current estimate of the minimizer $\mathbf{S} \subseteq \mathbf{CPC}$. Initially, $minarg = \emptyset$ and $minval = Assoc(X; T|\,\emptyset)$. It then augments $minarg$ by the member S of **CPC** that reduces the association $Assoc(X; T|minarg \cup \{\mathbf{S}\})$ the most. It continues in this fashion recursively until we condition on the full **CPC** or the minimum association achieved between X and T cannot be further reduced by augmenting $minarg$ by a single variable.

For example, examine the network structure in Figure 1, the set **Z**=$\{A, B\}$ is a minimal d-separating set of X and Y. Assuming, $Assoc(X; Y|\,\emptyset) \geq Assoc(X; Y|\,\{A\}) \geq Assoc(X; Y|\,\{A, B\})$ then **Z**, a d-separating set, can be discovered in a greedy fashion.

In phase II, the backward phase, *PMMPC* attempts to remove some of the false positives that may have entered in the first phase. The false positives are removed by testing $Ind(X; T|\mathbf{S})$ for some subset of the candidate parents and children set, $\mathbf{S} \subseteq \mathbf{CPC}$. If the independence holds, X is removed from **CPC**. The existence of a $\mathbf{S} \subseteq \mathbf{CPC}$ for which $Ind(X; T|\mathbf{S})$ is approximated by testing whether $\textsc{GreedyMinAssoc}$ returns zero.

For example, in the network structure of Figure 1 it is possible that X enters **CPC** before both A and B when the target is Y. Phase II, however, will remove X once both A and B are in **CPC** and the test $\textsc{GreedyMinAssoc}(X; Y, \{A, B\}, Assoc(X; Y|\,\emptyset), \emptyset)$ returns zero (since $Assoc(X; Y|\,\{A, B\}) = 0$). [2]

A Theoretical Analysis of *PMMS*

PMMS is a polynomial variant of the skeleton identification phase of the Max-Min Hill-Climbing (*MMHC*) algorithm (Brown, Tsamardinos, & Aliferis 2004; Tsamardinos, Brown, & Aliferis 2005). The difference between the algorithms is that *MMHC* implements the non-polynomial version of the parents and children algorithm, *MMPC*, that performs an exponential search for a d-separating set, while the polynomial version, *PMMS*, employs the greedy polynomial *PMMPC*. *MMHC* provably correctly identifies the BN skeleton, provided the distribution of the data $\mathcal{D}$ is faithful and the association and independence tests are reliable given the available data. Thus, any false positives introduced by *PMMS* are due to the use of the polynomial approximation of the heuristic. Notice that *PMMS* does not introduce any false negatives (i.e., will not miss any skeleton edges) when the assumptions hold: since the network to reconstruct is faithful, every $X \in \mathbf{PC}_T$ has a non-zero association with T conditioned on any variable subset, and thus, will enter and never be removed from in **CPC** (see Lemma 1 in Tsamardinos, Brown, & Aliferis 2005).

Following the complexity analysis of the non-polynomial algorithm (Tsamardinos, Brown, & Aliferis 2005) and noticing that the exponential search has been substituted with a polynomial greedy search, one can show that *PMMS* will perform at most $O(k \cdot PC \cdot |V|^2)$ calls to function $Assoc$, where k is the maximum size of any conditioning set, PC the average size of the parents and children set, and $|V|$ the number of variables. In the worst case, this may grow to $O(|V|^4)$. Notice that both the average and worst case complexity of the algorithm match the corresponding complexities of $O(|V|^2)$ and $O(|V|^4)$ of the Three Phase Dependency Analysis algorithm.

For each pair of variables X and Y, constraint-based algorithms search for a d-separating set **Z**, such that $Ind(X; T|\mathbf{Z})$. *PMMS* raises the following question: when will a greedy search identify such a d-separating set **Z** and when will it fail?

Cheng *et al.* (2002) use the analogy of active paths between two variables and pipes of flow of information to aid in addressing this question. The monotone DAG-faithful condition assumes that the more active paths, the larger the information (association) flow between two variables. Following this intuitive analogy, let us assume that a d-separating set between X and Y is $\mathbf{Z} = \{A, B\}$ as in Figure 1. We intuitively expect that as we condition on subsets of **Z** of increasing size and are closing active paths between X and Y the association will be dropping. If this is correct, then a greedy search will build up **Z** starting from the empty set.

[2]In our implementation we may remove additional false positives by checking whether the symmetry $X \in \mathbf{PC}_Y \Leftrightarrow Y \in \mathbf{PC}_X$ holds (for more details see Tsamardinos, Brown, & Aliferis 2005).

There are, however, situations where this assumption does not hold. We extend the analogy to consider not only the magnitude of the associations carried by the different paths, but also whether they are positive or negative in direction. The association carried over multiple paths is the sum of the individual associations of the paths.

We then intuitively expect that if two paths, e.g., $X \to A \to Y$ and $X \to B \to Y$ carry associations from X to Y that cancel each other out, then by conditioning on A and by closing the active path $X \to A \to Y$ the association between X and Y may rise, i.e., $Assoc(X;Y|\emptyset) < Assoc(X;Y|\{A\})$. This situation is depicted in Figure 1 when we consider the network structure *and* the probabilities given. The plus and minus signs on the edges denote a positive and a negative association respectively, e.g., A is more likely to get the same value as X (positive association) while Y is more likely to get a different value than B (negative association).

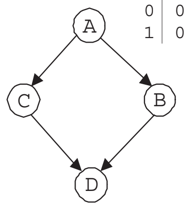
PMMS will fail to identify $\mathbf{Z} = \{A, B\}$ as a d-separating set for X and Y on the network of Figure 1 with the probabilities given, and thus return the false positive edge X—Y in the output skeleton. This is because $\mathit{Inf}(X;Y|\emptyset) = 0.014 < \mathit{Inf}(X;Y|\{A\}) = 0.015$ and also $\mathit{Inf}(X;Y|\emptyset) = 0.014 < \mathit{Inf}(X;Y|\{B\}) = 0.032$. When running $\mathit{PMMPC}(Y, \mathcal{D})$ both A and B will enter **CPC** (since they cannot be d-separated from Y); however, when trying to identify the minimum association of X and Y *the greedy search will stop conditioning on additional variables of **CPC**, because the information increases when conditioning on either A or B alone.* The backward phase of *PMMPC* will not remove X from the **CPC** because the greedy search is also used here. Therefore, $\mathit{PMMPC}(Y, \mathcal{D})$ will return $\{A, B, X\}$.

A slight adjustment to the probabilities of the network in Figure 1 (specifically, $P(Y{=}0|A{=}0, B{=}1){=}0.9$ and $P(Y{=}1|A{=}0, B{=}1){=}0.1$), results in the greedy heuristic giving correct results (i.e., it is able to find a minimal d-separating set). With the new probabilities, when blocking the active path $X \to A \to Y$ information decreases: $\mathit{Inf}(X;Y|\emptyset) = 0.0152 \geq \mathit{Inf}(X;Y|\{A\}) = 0.011$. This allows the greedy search to continue and discover $\{A, B\}$ as a d-separating set. This is despite the fact that closing the other active path $X \to B \to Y$ makes information increase: $\mathit{Inf}(X;Y|\emptyset) = 0.0152 \ngeq \mathit{Inf}(X;Y|\{B\}) = 0.030$. Thus, even for active paths with cancellation of association, it is unlikely that for all paths the association will increase when blocked, and thereby force the greedy search to stop.

Comparison of *PMMS* with *TPDA*

PMMS and *TPDA* are similar at a fundamental level. Both have a forward phase where variables enter a candidate parent and children set for each node T. This set is the **CPC** in the *PMMS* algorithm and the set of neighbors of T in the current draft of the skeleton in *TPDA*. They also both have a backward phase where variables are removed from this set. During these phases, both algorithms attempt to discover a set $\mathbf{Z}$ such that $Assoc(X;T|\mathbf{Z}) = 0$ for every X considered.

A major difference between the algorithms, however, is that *PMMS* builds up $\mathbf{Z}$ starting from the empty set. In contrast *TPDA* starts by conditioning on the full set of the candidate parents and children (some nodes may be removed from the conditioning set by considering the current skeleton graph) and removing nodes from this set to reach $\mathbf{Z}$.



A	P(A)
0	0.5
1	0.5

B	A	P(B|A)
0	0	0.7
1	0	0.3
0	1	0.3
1	1	0.7

C	A	P(C|A)
0	0	0.8
1	0	0.2
0	1	0.2
1	1	0.8

D	B	C	P(D|B,C)
0	0	0	0.9
0	1	0	0.5
0	0	1	0.5
0	1	1	0.1
1	0	0	0.1
1	1	0	0.5
1	0	1	0.5
1	1	1	0.9

Figure 2: An example Bayesian network from Cheng *et al.* (2002), slightly modified to make the conditional probability table values for variable D properly sum to one.

When the available sample is not enough to condition on the full set of the parents and children set of T, we expect *TPDA*'s strategy to fail to accurately estimate the conditional mutual information between T and any other node. On the other hand *PMMS* starts with the smallest conditioning set possible – the empty set – and proceeds with conditioning on larger sets only when the sample allows so. Thus, we expect *PMMS* to better reconstruct the skeleton when the available sample is low relative to the parent and children set sizes.

On the other hand, *TPDA*'s strategy may pay off for relatively large sample sizes. For example, suppose that X and T can be d-separated by $\mathbf{S} \subseteq \mathbf{Z}$, where $\mathbf{Z}$ is the current estimate of $\mathbf{PC}_T$. If the available sample is enough to condition on $\mathbf{Z}$ it may be possible that $Ind(X;T|\mathbf{Z})$, which *TPDA* will discover with a single call to mutual information. In contrast, *PMMS* has to perform at least $|\mathbf{S}|$ calls to $Assoc$ to identify $\mathbf{S}$.

This different search strategy also gives rise to different correctness properties. For example, *PMMS* will correctly reconstruct the non-monotone DAG-faithful network in Cheng *et al.* (2002), where *TPDA* fails (see Figure 2). Similarly, *TPDA* will correctly reconstruct the (non-monotone DAG-faithful) network in Figure 1 that is problematic for *PMMS*. The above intuitions are exactly corroborated by the empirical results presented in the next section.

Experimental Evaluation

For the subsequent experiments, the greedy hill-climbing search with a TABU list (*GTS*, following Friedman, Nachman, & Pe'er 1999), *PC* [3], Three Phase Dependency Analysis (*TPDA*), Max-Min Hill-Climbing (*MMHC*), and Polynomial Max-Min Skeleton (*PMMS*) have been implemented in Matlab 6.5. Optimal Reinsertion (*OR*, Moore & Wong 2003), Sparse Candidate (*SC*, Friedman, Nachman, & Pe'er 1999), and Greedy Equivalent Search (*GES*, Chickering

[3] *PC* is implemented as described in (Spirtes, Glymour, & Scheines 2000) and the Tetrad 3 documentation.

Table 1: Bayesian networks used in the evaluation sorted by number of variables.

Network	# Vars	# Edges	Domain Range
CHILD	20	25	2 - 6
INSURANCE	27	52	2 - 5
MILDEW	35	46	3 - 100
ALARM	37	46	2 - 4
BARLEY	48	84	2 - 67
HAILFINDER	56	66	2 - 11
CHILD3	60	79	2 - 6
INSURANCE3	81	163	2 - 5
CHILD5	100	126	2 - 6
ALARM3	111	149	2 - 4
HAILFINDER3	168	283	2 - 11
ALARM5	185	265	2 - 4

2002) use publicly available implementations by the original authors for the first two cases and the Tetrad 4 software for the last one. The statistical threshold used in *PC*, *MMHC*, and *PMMS* was the standard 5%; a threshold of 1% was used in *TPDA* as suggested by the authors. *OR* is an anytime algorithm and it was set to run both one and two times the amount of the time *MMHC* required on the same dataset. In addition, the algorithm was run with values chosen from the set $\{5, 10, 20\}$ for the parameter to constrain the number of parents to consider. The best performing variant of *OR* per dataset is reported in the tables below. *SC* was run with k = 5 or 10, where k is the size of candidate parents sets. Again, the best performing variant of the *SC* algorithm is reported. The parameter space for both algorithms is suggested by the algorithms' authors. Bayesian scoring with the equivalent sample size of 10 was used in *SC*, *GTS*, and *MMHC*.

The networks used in the evaluation are mainly obtained from real decision support systems covering a wide range of real life applications (many of the networks used are available at the Bayesian Network Repository http://www.cs.huji.ac.il/labs/compbio/Repository/). To experiment with larger networks than what is available in the public domain, a method developed by Statnikov, Tsamardinos, & Aliferis (2003) was used to generate large BNs by tiling several copies of smaller BNs. The tiling is performed in a way that maintains the structural and probabilistic properties of the original network in the tiled network (the number next to the name denoting the number of tiles). Overall, 12 networks were used in this evaluation with information on the networks given in Table 1.

From each network, 10 datasets were randomly sampled in sizes of 500, 1000, and 5000. The performance metrics of an algorithm were the averages over those 10 datasets for each sample size and network. All algorithms were run on Pentium Xeons, 2.4GHz, 2GB RAM running Linux. The evaluation here extends the results of Tsamardinos, Brown, & Aliferis (2005).

Table 2: Comparing Algorithms for Learning the Skeleton Network. The median of the number of statistical tests run, errors of commission, and errors of omission. *PMMS* requires fewer statistical tests than the non-polynomial algorithms *PC* and *MMHC* while maintaining comparable quality against *MMHC* and better than *PC* for small sample sizes.

	Number of Statistics		
Alg.	500 SS	1000 SS	5000 SS
PMMS	3.7K	4.9K	7.5K
MMHC	3.9K	5.9K	10.0K
PC	81.2K	78.5K	50.6K
TPDA	12.1K	8.8K	5.3K

	Total Errors = Errors of Commission + Errors of Omission		
Alg.	500 SS	1000 SS	5000 SS
PMMS	69.7 = 48.5 + 21.2	60.0 = 44.3 + 15.7	57.5 = 45.4 + 12.1
MMHC	64.7 = 43.5 + 21.2	47.1 = 31.3 + 15.8	34.9 = 22.5 + 12.4
PC	131.0 = 108.2 + 22.8	75.4 = 59.8 + 15.6	50.2 = 39.8 + 10.4
TPDA	176.7 = 129.6 + 47.1	103.8 = 70.2 + 33.6	35.2 = 9.3 + 25.9

Results of Evaluation

Skeleton Identification. First, the *PMMS* algorithm is compared with the other constraint-based methods for learning the skeleton network, namely the skeleton identification phase of *MMHC* (i.e., the non-polynomial version of *PMMS*), *PC*, and *TPDA* (results in Table 2). The metrics reported are found by calculating the median over all networks for a given algorithm and sample size (the median was used over the mean, because the distribution of values was usually skewed; *PMMS* performance is even better when measured by the mean). The first section of Table 2 reports the number of statistical tests performed by the algorithm. To compare the quality of the learned skeleton, the number of errors of commission (extra edges) and errors of omission (missing edges) are presented in the next section of Table 2.

PMMS in general outperforms the other polynomial algorithm *TPDA*. The only exception is at sample size 5000, corroborating our intuitions presented in the previous section. As expected *PMMS* requires fewer statistical tests than the non-polynomial algorithms *PC* and *MMHC* while maintaining comparable quality against *MMHC* and better than *PC* for small sample sizes. Since *PMMS* performs only a subset of the association calls completed by *MMHC*, as expected, the results show a larger number of errors of commission and a smaller number of errors of omission.

Bayesian Network Reconstruction. We extended *PMMS* to orient the edges of the identified skeleton and further refine the network with a greedy hill-climbing search and score procedure using the BDeu score (Heckerman, Geiger, & Chickering 1995) and augmented with a TABU list (as in

Table 3: Median Normalized Results. Each metric is normalized by dividing the metric for a particular sample size and network by the corresponding metric value for *PMMHC*. A median normalized metric value smaller than one corresponds to an algorithm with a faster time, better score, or fewer structural errors than *PMMHC*. *PMMHC* is the fastest algorithm on average. *PMMHC* proves more time efficient and accurate than most other state-of-the-art Bayesian network learning algorithms.

Time Results								
SS	*PMMHC*	*MMHC*	*PC*	*TPDA*	Best *OR*	Best *SC*	*GTS*	*GES*
500	1.00	1.45	2.00	9.11	1.60	3.72	5.22	2222.35
1000	1.00	1.20	1.94	3.80	1.41	4.37	5.38	1684.91
5000	1.00	1.44	1.73	0.88	1.77	4.03	5.46	778.50
Ave.	1.00	1.36	1.89	4.60	1.59	4.04	5.36	1561.92

Bayesian Score Results								
SS	*PMMHC*	*MMHC*	*PC*	*TPDA*	Best *OR*	Best *SC*	*GTS*	*GES*
500	1.00	1.00	1.13	1.34	1.00	1.01	0.99	1.05
1000	1.00	1.00	1.14	1.32	1.00	1.00	0.99	1.02
5000	1.00	1.00	1.02	1.10	1.00	1.01	1.00	1.11
Ave.	1.00	1.00	1.10	1.25	1.00	1.00	0.99	1.06

Structural Hamming Distance Results								
SS	*PMMHC*	*MMHC*	*PC*	*TPDA*	Best *OR*	Best *SC*	*GTS*	*GES*
500	1.00	1.00	2.94	3.02	1.06	1.24	1.09	1.14
1000	1.00	0.99	2.41	2.08	1.18	1.20	1.11	0.98
5000	1.00	1.00	1.72	1.54	1.43	1.46	1.42	1.28
Ave.	1.00	1.00	2.36	2.21	1.22	1.30	1.21	1.13

Friedman, Nachman, & Pe'er 1999). The greedy search was constrained to add edges only identified by *PMMS*. We call the resulting algorithm *PMMHC* and it is the polynomial equivalent of *MMHC*. We then compared *PMMHC* with other state-of-the-art BN learning algorithms.

We choose to use a search-and-score approach for learning the orientation of the network, the *PC* and *TPDA* algorithms both use a polynomial orientation procedure that identifies and directs the edges for v-structures and any compelled edges.

Three metrics were used to compare performance of the algorithms for learning the complete BNs. The first metric is execution time. This metric is used as a measure of computational efficiency, though it should be noted it is highly dependent on the specific implementation. The second metric used as a measure of reconstruction quality is the BDeu score of the final DAG returned by the algorithm on a separate testing set of data. The BDeu score is attractive because it assigns the same score to networks that are statistically indistinguishable from each other (i.e. , Markov equivalent).

The third metric reported that also measures quality is Structural Hamming Distance (*SHD*) (Tsamardinos, Brown, & Aliferis 2005). The Structural Hamming Distance directly compares the structure of the corresponding equivalence classes of the learned and original networks. The distance between the two is the number of the following operators required to make the DAG patterns match: add or delete an undirected edge, and add, remove, or reverse the orientation of a directed edge (see Tsamardinos, Brown, & Aliferis 2005 for more details and use of this metric). Thus, an algorithm will be penalized by an increase of the score by one for learning a DAG pattern with an extra un-oriented edge and by one for not orienting an edge that should have been oriented. The reason for using a metric on DAG patterns is so we do not penalize an algorithm for not differentiating between statistically indistinguishable orientations.

The median *normalized* results are in Table 3. The normalized metrics reported are found by dividing the metric for a given algorithm, network, and sample size by the corresponding metric for *PMMHC*. The median of these normalized values was then used to compare algorithms. A normalized metric of greater than one implies an algorithm is worse than *PMMHC* on the same learning task for that metric (i.e., slower, with more structural errors, or with smaller Bayesian scoring). The normalization essentially gives the same weight in the average to each network irrespective of its size, both in terms of sample and variables, as well as the difficulty of learning. Without it, the larger networks would dominate the median calculations.

PMMHC is the fastest algorithm on average, with *MMHC* being about 35% slower. *TPDA* is actually faster than *PMMHC* at higher sample but is significantly slower at smaller sample sizes. We note that *OR* should be excluded from this comparison since it was set to run 1 or 2 times the time of *MMHC* and that *GES*'s Tetrad implementation does not contain all optimizations suggested by its authors.

All the algorithms illustrate similar Bayesian score results with *GTS* posting the lowest and *TPDA* the highest scores. In terms of *SHD*, *MMHC* outperforms *PMMHC* finding slightly fewer errors for sample sizes 1000. Since *MMHC* started with a better skeleton (see Table 2) this result implies that the hill-climbing procedure was able to remove most of the false positives introduced by *PMMHC* and bring the *SHD* difference closer to zero. *GES* also learns a network with fewer structural errors for sample size 1000. In comparing *PMMHC* and *MMHC*, the polynomial algorithm gains significant computational efficiency for only a slight drop in quality.

It is important to observe that *TPDA* begins edge orientation with a better skeleton than *PMMHC* for sample size 5000. However, the resulting Bayesian networks have significantly more structural errors after the orientation phase than the output networks of *PMMHC*. This indicates that while constraint-based algorithms are time-efficient and high-quality for skeleton identification, the constraint-based techniques for orienting the edges as the ones used by the *PC* and *TPDA* are highly inaccurate when compared with Bayesian scoring techniques used by all other algorithms in our experiments.

Discussion and Conclusion

The problem of learning the most probable a posteriori under certain conditions BN from data is *NP*-hard. In this paper, we developed and explored an approximate polynomial learning algorithm called Polynomial Max-Min Skeleton (*PMMS*), which has been shown empirically to improve time-efficiency with a minimum degradation of reconstruction quality. Extensive empirical results also show *PMMS* (extended to also orient the edges) outperforming most state-of-the-art BN learning algorithms.

The results presented here indicate that polynomial strategies for learning BNs with high quality are feasible; an important open question is the theoretical properties of such algorithms and the minimal, least restrictive set of assumptions required for their optimality. In addition, our results provide evidence for the efficiency with which constraint-based algorithms can identify the skeleton of a network and the advantages of search-and-score procedures for orienting the edges in the skeleton over constraint-based ones.

Acknowledgements

The first author is supported by the National Library of Medicine grant T15 LM07450-01. The second and third author are supported by the National Library of Medicine grant LM-7948-01.

References

Brown, L.; Tsamardinos, I.; and Aliferis, C. 2004. A novel algorithm for scalable and accurate bayesian network learning. In *11th World Congress on Medical Informatics (MEDINFO)*. San Francisco, California: MEDINFO.

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., and Meek, C. 2003. Monotone dag faithfulness: A bad assumption. Technical Report MSR-TR-2003-16, Microsoft Research.

Chickering, D.; Meek, C.; and Heckerman, D. 2003. Large-sample learning of bayesian networks is np-hard. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, 124–133. UAI.

Chickering, D. 2002. Optimal structure identification with greedy search. *Journal of Machine Learning Research* 3(1):507–554.

Friedman, N.; Nachman, I.; and Pe'er, D. 1999. Learning bayesian network structure from massive datasets: The "sparse candidate" algorithm. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99)*, 206–215. UAI.

Heckerman, D.; Geiger, D.; and Chickering, D. 1995. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning* 20(3):197–243.

Moore, A., and Wong, W. 2003. Optimal reinsertion: A new search operator for accelerated and more accurate bayesian network structure learning. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*. ICML.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems*. San Mateo, CA: Morgan Kaufmannn Publishers.

Spirtes, P.; Glymour, C.; and Scheines, R. 2000. *Causation, Prediction, and Search*. Cambridge, MA: MIT Press, 2nd edition.

Statnikov, A.; Tsamardinos, I.; and Aliferis, C. 2003. An algorithm for generation of large bayesian networks. Technical Report TR-03-01, Vanderbilt University.

Tsamardinos, I.; Aliferis, C.; and Statnikov, A. 2003. Time and sample efficeint discovery of markov blankets and direct causal relations. In *Proceedings of Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 673–678. Washington, DC: ACM SIGKDD.

Tsamardinos, I.; Brown, L.; and Aliferis, C. 2005. The max-min hill-climbing bayesian network structure learning algorithm. Technical Report DSL-TR-05-01, Vanderbilt University.