Finding All Bayesian Network Structures within a Factor of Optimal

Zhenyu A. Liao$^1$, Charupriya Sharma$^1$, James Cussens$^2$ and Peter van Beek$^1$

$^1$David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON Canada
{z6liao, c9sharma, vanbeek}@uwaterloo.ca

$^2$Department of Computer Science
University of York
York, United Kingdom
james.cussens@york.ac.uk

Abstract
A Bayesian network is a widely used probabilistic graphical model with applications in knowledge discovery and prediction. Learning a Bayesian network (BN) from data can be cast as an optimization problem using the well-known score-and-search approach. However, selecting a single model (i.e., the best scoring BN) can be misleading or may not achieve the best possible accuracy. An alternative to committing to a single model is to perform some form of Bayesian or frequentist model averaging, where the space of possible BNs is sampled or enumerated in some fashion. Unfortunately, existing approaches for model averaging either severely restrict the structure of the Bayesian network or have only been shown to scale to networks with fewer than 30 random variables. In this paper, we propose a novel approach to model averaging inspired by performance guarantees in approximation algorithms. Our approach has two primary advantages. First, our approach only considers credible models in that they are optimal or near-optimal in score. Second, our approach is more efficient and scales to significantly larger Bayesian networks than existing approaches.

Introduction
A Bayesian network is a widely used probabilistic graphical model with applications in knowledge discovery, explanation, and prediction (Darwiche 2009, Koller and Friedman 2009). A Bayesian network (BN) can be learned from data using the well-known score-and-search approach, where a scoring function is used to evaluate the fit of a proposed BN to the data, and the space of directed acyclic graphs (DAGs) is searched for the best-scoring BN. However, selecting a single model (i.e., the best-scoring BN) may not always be the best choice. When one is using BNs for knowledge discovery and explanation with limited data, selecting a single model may be misleading as there may be many other BNs that have scores that are very close to optimal and the posterior probability of even the best-scoring BN is often close to zero. As well, when one is using BNs for prediction, selecting a single model may not achieve the best possible accuracy.

An alternative to committing to a single model is to perform some form of Bayesian or frequentist model averaging (Claeskens and Hjort 2008, Hoeting et al. 1999, Koller and Friedman 2009). In the context of knowledge discovery, Bayesian model averaging allows one to estimate, for example, the posterior probability that an edge is present, rather than just knowing whether the edge is present in the best-scoring network. Previous work has proposed Bayesian and frequentist model averaging approaches to network structure learning that enumerate the space of all possible DAGs (Koivisto and Sood 2004), sample from the space of all possible DAGs (He, Tian, and Wu 2016, Madigan and Raftery 1994), consider the space of all DAGs consistent with a given ordering of the random variables (Buntine 1991, Dash and Cooper 2004), consider the space of tree-structured or other restricted DAGs (Madigan and Raftery 1994, Meilă and Jaakkola 2000), and consider only the $k$-best scoring DAGs for some given value of $k$ (Chen, Choi, and Darwiche 2015, Chen, Choi, and Darwiche 2016, Chen, Darwiche, and Choi 2018, Chen and Tian 2014, He, Tian, and Wu 2016, Tian, He, and Ram 2010). Unfortunately, these existing approaches either severely restrict the structure of the Bayesian network, such as only allowing tree-structured networks or only considering a single ordering, or have only been shown to scale to small Bayesian networks with fewer than 30 random variables.

In this paper, we propose a novel approach to model averaging for BN structure learning that is inspired by performance guarantees in approximation algorithms. Let $OPT$ be the score of the optimal BN and assume without loss of generality that the optimization problem is to find the minimum-score BN. Instead of finding the $k$-best networks for some fixed value of $k$, we propose to find all Bayesian networks $\mathcal{G}$ that are within a factor $\rho$ of optimal; i.e.,

$$OPT \leq score(\mathcal{G}) \leq \rho \cdot OPT,$$

for some given value of $\rho \geq 1$, or equivalently,

$$OPT \leq score(\mathcal{G}) \leq OPT + \epsilon,$$

for $\epsilon = (\rho - 1) \cdot OPT$. Instead of choosing arbitrary values for $\epsilon$, $\epsilon \geq 0$, we show that for the two scoring functions BIC/MDL and BDeu, a good choice for the value of $\epsilon$ is closely related to the Bayes factor, a model selection criterion summarized in (Kass and Raftery 1995).

Our approach has two primary advantages. First, our approach only considers credible models in that they are optimal or near-optimal in score. Approaches that enumerate...
or sample from the space of all possible models consider DAGs with scores that can be far from optimal; for example, for the BIC scoring function the ratio of worst-scoring to best-scoring network can be four or five orders of magnitude\footnote{Madigan and Raftery (1994) deem such models discredited when they make a similar argument for not considering models whose probability is greater than a factor from the most probable.}. A similar but more restricted case can be made against the approach which finds the $k$-best networks since there is no \textit{a priori} way to know how to set the parameter $k$ such that only credible networks are considered. Second, and perhaps most importantly, our approach is significantly more efficient and scales to Bayesian networks with almost 60 random variables. Existing methods for finding the optimal Bayesian network structure, e.g., (Bartlett and Cussens 2013, van Beek and Hoffmann 2015) rely heavily for their success on a significant body of pruning rules that remove from consideration many candidate parent sets both before and during the search. We show that many of these pruning rules can be naturally generalized to preserve the Bayesian networks that are within a factor of optimal. We modify GOBNILP (Bartlett and Cussens 2013), a state-of-the-art method for finding an optimal Bayesian network, to implement our generalized pruning rules and to find all \textit{near-optimal} networks. We show in an experimental evaluation that the modified GOBNILP scales to significantly larger networks without resorting to restricting the structure of the Bayesian networks that are learned.

\section*{Background}

In this section, we briefly review the necessary background in Bayesian networks and scoring functions, and define the Bayesian network structure learning problem (for more background on these topics see (Darwiche 2009, Koller and Friedman 2009)).

\subsection*{Bayesian Networks}

A Bayesian network (BN) is a probabilistic graphical model that consists of a labeled directed acyclic graph (DAG), $G = (V, E)$ in which the vertices $V = \{V_1, \ldots, V_n\}$ correspond to $n$ random variables, the edges $E$ represent direct influence of one random variable on another, and each vertex $V_i$ is labeled with a conditional probability distribution $P(V_i | \Pi_i)$ that specifies the dependence of the variable $V_i$ on its set of parents $\Pi_i$ in the DAG. A BN can alternatively be viewed as a factorized representation of the joint probability distribution over the random variables and as an encoding of the Markov condition on the nodes; i.e., given its parents, every variable is conditionally independent of its non-descendants.

Each random variable $V_i$ has state space $\Omega_i = \{v_{i1}, \ldots, v_{ir_i}\}$, where $r_i$ is the cardinality of $\Omega_i$ and typically $r_i \geq 2$. Each $\Pi_i$ has state space $\Omega_{\Pi_i} = \{\pi_{i1}, \ldots, \pi_{ir_{\Pi_i}}\}$. We use $r_{\Pi_i}$ to refer to the number of possible instantiations of the parent set $\Pi_i$ of $V_i$ (see Figure 1). The set $\theta = \{\theta_{ijk}\}$ for all $i = \{1, \ldots, n\}$, $j = \{1, \ldots, r_{\Pi_i}\}$ and $k = \{1, \ldots, r_i\}$ represents parameters in $G$ where each element in $\theta$, $\theta_{ijk} = P(v_{ik} | \pi_{ij})$.

The predominant method for Bayesian network structure learning (BNSL) from data is the \textit{score-and-search} method. Let $I = \{I_1, \ldots, I_N\}$ be a dataset where each instance $I_i$ is an $n$-tuple that is a complete instantiation of the variables in $V$. A \textit{scoring function} $\sigma(G | I)$ assigns a real value measuring the quality of $G = (V, E)$ given the data $I$. Without loss of generality, we assume that a lower score represents a better quality network structure and omit $I$ when the data is clear from context.

\begin{definition}
Given a non-negative constant $\epsilon$ and a dataset $I = \{I_1, \ldots, I_N\}$, a \textit{credible network} $G$ is a network that has a score $\sigma(G)$ such that $OPT - \epsilon \leq \sigma(G) \leq OPT + \epsilon$, where $OPT$ is the score of the optimal Bayesian network.
\end{definition}

In this paper, we focus on solving a problem we call the $\epsilon$-Bayesian Network Structure Learning ($\epsilon$BNSL). Note that the BNSL for the optimal network(s) is a special case of $\epsilon$BNSL where $\epsilon = 0$.

\begin{definition}
Given a non-negative constant $\epsilon$, a dataset $I = \{I_1, \ldots, I_N\}$ over random variables $V = \{V_1, \ldots, V_n\}$ and a scoring function $\sigma$, the $\epsilon$-Bayesian Network Structure Learning ($\epsilon$BNSL) problem is to find all credible networks.
\end{definition}

\subsection*{Scoring Functions}

Scoring functions usually balance goodness of fit to the data with a penalty term for model complexity to avoid overfitting. Common scoring functions include BIC/MDL (Lam and Bacchus 1994, Schwarz 1978) and BDeu (Buntine 1991, Heckerman, Geiger, and Chickering 1995). An important property of these (and most) scoring functions is decomposability, where the score of the entire network $\sigma(G)$ can be rewritten as the sum of local scores associated to each vertex $\sum_{i=1}^{n} \sigma(V_i, \Pi_i)$ that only depends on $V_i$ and its parent set $\Pi_i$ in $G$. The local score is abbreviated below as $\sigma(\Pi_i)$ when the local node $V_i$ is clear from context. Pruning techniques can be used to reduce the number of candidate parent sets that need to be considered, but in the worst-case the number of candidate parent sets for each variable $V_i$ is exponential in $n$, where $n$ is the number of vertices in the DAG.

In this work, we focus on the Bayesian Information Criterion (BIC) and the Bayesian Dirichlet, specifically BDeu,
scoring functions. The BIC scoring function in this paper is defined as
\[
BIC : \sigma(G) = \max_\theta L_{G, I}(\theta) - t(G) \cdot w.
\]
Here, \( w = \frac{\log N}{2} \), \( t(G) \) is a penalty term and \( L_{G, I}(\theta) \) is the log likelihood, given by
\[
L_{G, I}(\theta) = \sum_{i=1}^{n} \sum_{j=1}^{r_i} \sum_{k=1}^{r_k} \log \theta_{ijk}^{n_{ijk}},
\]
where \( n_{ijk} \) is the number of instances in \( I \) where \( v_{ik} \) and \( \pi_{ij} \) co-occur. As the BIC function is decomposable, we can associate a score to \( \Pi_i \), a candidate parent set of \( V_i \) as follows,
\[
BIC : \sigma(\Pi_i) = \max_\theta L(\theta_i) - t(\Pi_i) \cdot w.
\]
Here, \( L(\theta_i) = \sum_{j=1}^{r_i} \sum_{k=1}^{r_k} \log \theta_{ijk} \) and \( t(\Pi_i) = \gamma_{\Pi_i}(r_i - 1) \). The BDeu scoring function in this paper is defined as
\[
\text{BDeu} : \sigma(G) = \sum_{i=1}^{n} \sum_{j=1}^{r_i} \log \frac{\Gamma(\alpha_i)}{\Gamma(\alpha_i + n_{ij})} + \sum_{i=1}^{n} \sum_{j=1}^{r_i} \sum_{k=1}^{r_k} \log \frac{\Gamma(\alpha_i + n_{ijk})}{\Gamma(\alpha_i)}.
\]
where \( \alpha \) is the equivalent sample size and \( n_{ij} = \sum_k n_{ijk} \). As the BDeu function is decomposable, we can associate a score to \( \Pi_i \), a candidate parent set of \( V_i \) as follows,
\[
\text{BDeu} : \sigma(\Pi_i) = \sum_{j=1}^{r_i} \left( \log \frac{\Gamma(\alpha_i)}{\Gamma(\alpha_i + n_{ij})} \right) + \sum_{k=1}^{r_k} \log \frac{\Gamma(\alpha_i + n_{ijk})}{\Gamma(\alpha_i)}.
\]

The Bayes Factor

In this section, we show that a good choice for the value of \( \epsilon \) for the \( \epsilon \)BNSL problem is closely related to the Bayes factor (BF), a model selection criterion summarized in (Kass and Raftery 1995).

The BF was proposed by Jeffreys as an alternative to significance test (Jeffreys 1967). It was thoroughly examined as a practical model selection tool in (Kass and Raftery 1995). Let \( G_0 \) and \( G_1 \) be DAGs (BNs) in the set of all DAGs \( G \) defined over \( V \). The BF in the context of BNs is defined as,
\[
BF(G_0, G_1) = \frac{P(I \mid G_0)}{P(I \mid G_1)}
\]

namely the odds of the probability of the data predicted by network \( G_0 \) and \( G_1 \). The actual calculation of the BF often relies on Bayes’ Theorem as follows,
\[
P(G_0 \mid I) P(I \mid G_0) = P(I \mid G_1) P(G_0) = P(I, G_0)
\]

Since it is typical to assume the prior over models is uniform in \( \epsilon \)BNSL, the BF can then be obtained using either \( P(I \mid G) \) or \( P(I, G) \forall G \in G \). We use those two representations to show how BIC and BDeu scores relate to the BF.

Using Laplace approximation and other simplifications in (Ripley 1996), Ripley derived the following approximation to the logarithm of the marginal likelihood for network \( G \) (a similar derivation is given in (Claeskens and Hjort 2008)),
\[
\log P(I \mid G) = L_{G, I}(\hat{\theta}) - t(G) \cdot \frac{\log N}{2} + t(G) \cdot \frac{\log 2\pi}{2} - \frac{1}{2} \log |J_{G, I}(\hat{\theta})| + \log P(\hat{\theta} \mid G),
\]

where \( \hat{\theta} \) is the maximum likelihood estimate of model parameters and \( J_{G, I}(\hat{\theta}) \) is the Hessian matrix evaluated at \( \hat{\theta} \). It follows that,
\[
\log P(I \mid G) = -BIC(I, G) + O(1).
\]

The above equation shows that the BIC score was designed to approximate the log marginal likelihood. If we drop the lower-order term, we can then obtain the following equation,
\[
BIC(I, G_1) - BIC(I, G_0) = \log \frac{P(I \mid G_0)}{P(I \mid G_1)} = \log BF(G_0, G_1).
\]

It has been indicated in (Kass and Raftery 1995) that as \( N \to \infty \), the difference of the two BIC scores, dubbed the Schwarz criterion, approaches the true value of \( \log BF \) such that,
\[
\frac{BIC(I, G_1) - BIC(I, G_0) - \log BF(G_0, G_1)}{\log BF(G_0, G_1)} \to 0.
\]

Therefore, the difference of two BIC scores can be used as a rough approximation to \( \log BF \). Note that some papers define BIC to be twice as large as the BIC defined in this paper, but the above relationship still holds albeit with twice the logarithm of the BF.

Similarly, the difference of the BDeu scores can be expressed in terms of the BF. In fact, the BDeu score is the log marginal likelihood where there are Dirichlet distributions over the parameters (Buntine 1991, Heckerman, Geiger, and Chickering 1995), i.e.,
\[
\log P(I \mid G) = -BDeu(I, G),
\]

and thus
\[
BDeu(I, G_1) - BDeu(I, G_0) = \log \frac{P(I, G_0)}{P(I, G_1)} = \log BF(G_0, G_1).
\]

The above results are consistent with the observation in (Kass and Raftery 1995) that the \( \log BF \) can be interpreted as a measure for the relative success of two models at predicting data, sometimes referred to as the “weight of evidence”, without assuming either model is true. The desired value of BF, however, is often specific to a study and determined with domain knowledge, e.g., a BF of 1000 is more appropriate in forensic science. Heckerman, Geiger, and Chickering (1995) proposed the following interpreting
scale for the BF: a BF of 1 to 3 bears only anecdotal evidence, a BF of 3 to 20 suggests some positive evidence that \( G_0 \) is better, a BF of 20 to 150 suggests strong evidence in favor of \( G_0 \), and a BF greater than 150 indicates very strong evidence. If we deem 20 to be the desired BF in eBNSL, i.e., \( G_0 = G^* \) and \( \epsilon = \log(20) \), then any network with a score less than \( \log(20) \) away from the optimal score would be credible, otherwise it would be discredited. Note that the ratio of posterior probabilities was defined as \( \lambda \) in (Tian, He, and Ram 2010, Chen and Tian 2014) and was used as a metric to assess arbitrary values of \( k \) in finding the \( k \)-best networks.

Finally, the eBNSL problem using the BIC or BDeu scoring function given a desired BF can be written as,

\[
OPT \leq \text{score}(\mathcal{G}) \leq OPT + \log BF.
\]  

**Pruning Rules for Candidate Parent Sets**

To find all near-optimal BNs given a BF, the local score \( \sigma(\Pi_i) \) for each candidate parent set \( \Pi_i \subseteq 2^{V-V_i} \) and each random variable \( V_i \) must be computed. As this is very costly prohibitive, the search space of candidate parent sets can be pruned, provided that global optimality constraints are not violated.

A candidate parent set \( \Pi_i \) can be safely pruned given a non-negative constant \( \epsilon \in \mathbb{R}^+ \) if \( \Pi_i \) cannot be the parent set of \( V_i \) in any network in the set of credible networks. Note that for \( \epsilon = 0 \), the set of credible networks just contains the optimal network(s). We discuss the original rules and their generalization below and proofs for them can be found in the supplemental material.

Teyssier and Koller (2005) gave a pruning rule for all decomposable scoring functions. This rule compares the score of a candidate parent set to those of its subsets. We give a relaxed version of the rule.

**Lemma 1.** Given a vertex variable \( V_i \), candidate parent sets \( \Pi_i \) and \( \Pi'_i \), and some \( \epsilon \in \mathbb{R}^+ \), if \( \Pi_j \subseteq \Pi'_j \) and \( \sigma(\Pi_j) + \epsilon \geq \sigma(\Pi'_j) \), \( \Pi'_j \) can be safely pruned.

**Pruning with BIC/MDL Score**

A pruning rule comparing the BIC score and penalty associated to a candidate parent set to those of its subsets was introduced in (de Campos and Ji 2011). The following theorem gives a relaxed version of that rule.

**Theorem 2.** Given a vertex variable \( V_j \), candidate parent sets \( \Pi_j \) and \( \Pi'_j \), and some \( \epsilon \in \mathbb{R}^+ \), if \( \Pi_j \subseteq \Pi'_j \) and \( \sigma(\Pi_j) - t(\Pi'_j) + \epsilon < 0 \), \( \Pi'_j \) and all supersets of \( \Pi'_j \) can be safely pruned if \( \sigma \) is the BIC function.

Another pruning rule for BIC appears in (de Campos and Ji 2011). This provides a bound on the number of possible instantiations of subsets of a candidate parent set. The following theorem relaxes that rule.

**Theorem 3.** Given a vertex variable \( V_i \), and a candidate parent set \( \Pi_i \) such that \( r_{ii} > \frac{N \log 2}{\epsilon w_{i-1}} \) for some \( \epsilon \in \mathbb{R}^+ \), if \( \Pi_i \subseteq \Pi'_i \), then \( \Pi'_i \) can be safely pruned if \( \sigma \) is the BIC scoring function.

The following corollary of Theorem 3 gives a useful upper bound on the size of a candidate parent set.

**Corollary 4.** Given a vertex variable \( V_i \), and candidate parent set \( \Pi_i \), if \( \Pi_i \) has more than \( |\log_2 N + \epsilon| \) elements, for some \( \epsilon \in \mathbb{R}^+ \), \( \Pi_i \) can be safely pruned if \( \sigma \) is the BIC scoring function.

Corollary 4 provides an upper-bound on the size of parent sets based solely on the sample size. The following table summarizes such an upper-bound given different amounts of data \( N \) and a BF of 20.

| \( |\Pi| \) | 10 | 12 | 15 | 18 | 20 | 25 | 30 | 50 | 100 |
|---|---|---|---|---|---|---|---|---|---|
| \( N \) & 100 & 500 & \( 10^3 \) & \( 5 \times 10^3 \) & \( 10^4 \) & \( 5 \times 10^4 \) & \( 10^5 \) |

The entropy of a candidate parent set is also a useful measure for pruning. A pruning rule, given by (de Campos et al. 2018), provides an upper bound on conditional entropy of candidate parent sets and their subsets. We give a relaxed version of their rule. First, we note that entropy for a vertex variable \( V_i \) is given by,

\[
H(V_i) = -\sum_{k=1}^{r_{ik}} \frac{n_{ik}}{N} \log \frac{n_{ik}}{N},
\]

where \( n_{ik} \) represents how many instances in the dataset contain \( v_{ik} \), where \( v_{ik} \) is an element in the state space \( \Omega_i \) of \( V_i \). Similarly, entropy for a candidate parent set \( \Pi_i \) is given by,

\[
H(\Pi_i) = -\sum_{j=1}^{r_{ij}} \frac{n_{ij}}{N} \log \frac{n_{ij}}{N}.
\]

Conditional information is given by

\[
H(X \mid Y) = H(X \cup Y) - H(Y).
\]

**Theorem 5.** Given a vertex variable \( V_i \), and candidate parent set \( \Pi_i \), let \( V_j \notin \Pi_i \) such that \( N \cdot \min \{ H(V_i \mid \Pi_i), H(V_j \mid \Pi_i) \} \geq (1 - r_j) \cdot t(\Pi_i) + \epsilon \) for some \( \epsilon \in \mathbb{R}^+ \). Then the candidate parent set \( \Pi'_i = \Pi_i \cup \{ V_j \} \) and all its supersets can be safely pruned if \( \sigma \) is the BIC scoring function.

**Pruning with BDeu Score**

A pruning rule for the BDeu scoring function appears in (de Campos et al. 2018) and a more general version is included in (Cussens and Bartlett 2012). Here, we present a relaxed version of the rule in (Cussens and Bartlett 2012).

**Theorem 6.** Given a vertex variable \( V_i \), and candidate parent sets \( \Pi_i \) and \( \Pi'_i \) such that \( \Pi_i \subset \Pi'_i \) and \( \Pi_i \neq \Pi'_i \), let \( r_{ii}^+ (\Pi'_i) \) be the number of positive counts in the contingency table for \( \Pi'_i \). If \( \sigma(\Pi_i) + \epsilon < r_{ii}^+ (\Pi'_i) \cdot \log r_{ii}^+ \) for some \( \epsilon \in \mathbb{R}^+ \) then \( \Pi'_i \) and the supersets of \( \Pi'_i \) can be safely pruned if \( \sigma \) is the BDeu scoring function.

**Experimental Evaluation**

In this section, we evaluate the proposed BF based method and compare its performance with published \( k \)-best solvers.

Our proposed method is more memory efficient comparing to the \( k \)-best based solvers in BDeu scoring and often
Table 1: The search time \( T \), the number of collected networks \(|G|\) and the number of MECs \(|M|\) in the collected networks at BF = 3, 20 and 150 using BIC, where \( n \) is the number of random variables in the dataset, \( N \) is the number of instances in the dataset and OT = Out of Time.

collects more networks in a shorter period of time. With the pruning rules generalized above, our method can scale up to datasets with 57 variables in BIC scoring, whereas the previous best results are reported on a network of 29 variables using the \( k \)-best approach with score pruning (Chen, Darwiche, and Choi 2018).

The datasets are obtained from the UCI Machine Learning Repository (Dheeru and Karra Taniskidou 2017) and the Bayesian Network Repository\(^2\). Some of the complete local scoring files are downloaded from the GOBNILP website\(^3\) and are used for the \( k \)-best related experiments only. Since not all solvers in the \( k \)-best experiments can take in scoring files, we exclude the time to compute local scores from the comparison. Both BIC/MDL (Schwarz 1978, Lam and Bacchus 1994) and BDeu (Buntine 1991, Heckerman, Geiger, and Chickering 1995) scoring functions are used where applicable. All experiments are conducted on computers with 2.2 GHz Intel E7-4850V3 processors. Each experiment is limited to 64 GB of memory and 24 hours of CPU time.

### The Bayes Factor Approach

We modified the development version of GOBNILP, referred below as GOBNILP\(_{\text{dev}}\), to apply pruning rules presented above during scoring and supplied appropriate parameter settings for collecting near-optimal networks. The code is compiled with SCIP 6.0.0 and CPLEX 12.8.0. GOBNILP extends the SCIP Optimization Suite (Gleixner et al. 2018) by adding a constraint handler for handling the acyclicity constraint for DAGs. If multiple BNs are required GOBNILP\(_{\text{dev}}\) just calls SCIP to ask it to collect feasible solutions. In this mode, when SCIP finds a solution, the solution is stored,ting that solution infeasible and the search continues. This differs from (and is much more efficient than) GOBNILP’s current method for finding \( k \)-best BNs where an entirely new search is started each time a new BN is found. Recent version of SCIP have a separate “reoptimization” method which might allow better \( k \)-best performance for GOBNILP but we do not explore that here. By default when SCIP is asked to collect solutions it turns off all cutting plane algorithms. This led to very poor GOBNILP performance since GOBNILP relies on cutting plane generation. Therefore, this default setting is overridden in GOBNILP\(_{\text{dev}}\) to allow cutting planes when collecting solutions. To find only solutions with objective no worse than \((OPT + \epsilon)\), SCIP’s SCIPsetObjlimit function is used. Note that, for efficiency reasons, this is not effected by adding a linear constraint.

We first use GOBNILP\(_{\text{dev}}\) to find the optimal scores since GOBNILP\(_{\text{dev}}\) takes objective limit \((OPT + \epsilon)\) for enumerating feasible networks. Then all networks falling into the limit are collected with a counting limit of 150,000. Finally the collected networks are categorized into Markov equivalence classes (MECs), where two networks belong to the same MEC iff they have the same skeleton and v-
Table 2: The search time $T$ and the number of collected networks $k$, $|G_k|$ and $|G_{20}|$ for KBest, KBestEC and GOBNILP_dev (BF = 20) using BDeu, where $n$ is the number of random variables in the dataset, $N$ is the number of instances in the dataset, OM = Out of Memory, OT = Out of Time and ES = Error in Scoring. Note that $|G_k|$ is the number of DAGs covered by the $k$-best MECs in KBestEC and $|M_{20}|$ is the number of MECs in the networks collected by GOBNILP_dev.

Bayes Factor vs. $k$-Best

In this section, we compare our approach with published solvers that are able to find a subset of top-scoring networks with the given parameter $k$. The solvers under consideration are KBest$_{12b}^4$ from (Tian, He, and Ram 2010), KBestEC$^5$ from (Chen and Tian 2014), and GOBNILP 1.6.3$^6$ (Bartlett and Cussens 2013), referred to as KBest, KBestEC and GOBNILP below. The first two solvers are based on the dynamic programming approach introduced in (Silander and Myllymäki 1995). Due to the lack of support for BIC in KBest and KBestEC, only BDeu with a equivalent sample size of one is used in corresponding experiments.

---

| Data      | $n$ | $N$ | $T_k$ (s) | $k$ | $T_{EC}$ (s) | $|G_k|$ | $T_{20}$ (s) | $|G_{20}|$ | $|M_{20}|$ |
|-----------|-----|-----|-----------|----|-------------|--------|-------------|---------|---------|
| tic tac toe | 10  | 958 | 0.2       | 10 | 0.5         | 67     | 0.6         | 152     | 24      |
| wine      | 14  | 178 | 3.4       | 10 | 12.0        | 60     | 35.9        | 8,734   | 6,262   |
| adult     | 14  | 32,561 | 3.3     | 10 | 633.5        | 68     | 9.3         | 792     | 19      |
| nlcs       | 16  | 3,236 | 406.6   | 100| 13,224.6     | OT     | 125.5       | 652     | 326     |
| msnbc     | 17  | 58,265 | 26.0    | 10 | 18,788.0     | 200    | 56,344.8    | 200     | 10      |
| letter     | 17  | 20,000 | 34.1    | 10 | 101.9        | 30     | 60          | 621     | 207     |
| voting     | 17  | 435 | 33.5      | 10 | 99.8         | 52     | 8,418.8     | 29,073  | 6,761   |
| zoo        | 17  | 101 | 1,041.7   | 100| 1,829.2      | 3,392  | 41,503.9    | 41,503.9| 207     |
| hepatitis  | 20  | 155 | 351.2     | 10 | 872.3        | 89     | 8,418.8     | 29,073  | 6,761   |
| parkinsons | 23  | 195 | 3,908.2   | 10 | 1,515.9      | 150,000| 1,843.4     | 100     | 3,534   |
| autos      | 26  | 159 | OM        | 1   | OM          | OM     | 30         | 30      | 30      |
| insurance  | 27  | 1,000 | OM       | 1   | OM          | OM     | 8.3        | 1,081   | 133     |

The results also indicate that the number of collected networks and the number of MECs at three BF levels varies substantially across different datasets. In general, datasets with smaller sample sizes tend to have more networks collected at a given BF since near-optimal networks have similar posterior probabilities to the best network. Although the desired level of BF for a study, like the p-value, is often determined with domain knowledge, the proposed approach, given sufficient samples, will produce meaningful results that can be used for further analysis.
The most recent stable version of GOBNILP is 1.6.3 that works with SCIP 3.2.1. The default configuration is used and experiments are conducted for both BIC and BDeu scoring functions. However, the $k$-best results are omitted here due to its poor performance. Despite that GOBNILP can iteratively find the $k$-best networks in descending order by adding linear constraints, the pruning rules designed to find the best network are turned off to preserve sub-optimal networks. In fact, the memory usage often exceeded 64 GB during the initial ILP formulation, indicating that the lack of pruning rules posed serious challenge for GOBNILP. GOBNILP dev, on the other hand, can take advantage of the pruning rules designed to find the best network are turned off to preserve sub-optimal networks. In fact, the memory usage often exceeded 64 GB during the initial ILP formulation, indicating that the lack of pruning rules posed serious challenge for GOBNILP. GOBNILP dev, on the other hand, can take advantage of the pruning rules presented above in the proposed BF approach and its results compare favorably to KBest and KBestEC.

The experimental results of KBest, KBestEC and GOBNILP dev are reported in Table 2, where $n$ is the number of random variables in the dataset, $N$ is the number of instances in the dataset, and $k$ is the number of top scoring networks. The search time $T$ is reported for KBest, KBestEC and GOBNILP dev (BF = 20). The number of DAGs covered by the $k$ MECs $|\mathcal{G}_k|$ is reported for KBestEC. In comparison, the last two columns are the number of found networks $|\mathcal{G}_{20}|$ and the number of MECs $|\mathcal{M}_{20}|$ using the BF approach with a given BF of 20 and BDeu scoring function.

As the number of requested networks $k$ increases, the search time for both KBest and KBestEC grows exponentially. The KBest and KBestEC are designed to address problems of size fewer than 20 variables, and so they have some difficulty with larger datasets. They also fail to generate correct scoring files for msnbc. KBestEC seems to successfully expand the coverage of DAGs with some overhead for checking equivalence classes. However, KBestEC took much longer than KBest for some instances, e.g., nlts and letter, and the number of DAGs covered by the found MECs is inconsistent for nlts, letter and zoo. The search time for the BF approach is improved over the $k$-best approach except for datasets with very large sample sizes. The generalized pruning rules are very effective in reducing the search space, indicating that the lack of data leads to many BNs with similar probabilities. It would be reasonable to choose a large value for $k$ in model averaging when data is scarce and vice versa, but only the BF approach is able to automatically find the appropriate and credible set of networks for further analysis.

**Conclusion**

Existing approaches for model averaging for Bayesian network structure learning either severely restrict the structure of the Bayesian network or have only been shown to scale to networks with fewer than 30 random variables. In this paper, we proposed a novel approach to model averaging inspired by performance guarantees in approximation algorithms that considers all networks within a factor of optimal. Our approach has two primary advantages. First, our approach only considers credible models in that they are optimal or near-optimal in score. Second, our approach is significantly more efficient and scales to much larger Bayesian networks than existing approaches. We modified GOBNILP (Bartlett and Cussens 2013), a state-of-the-art method for finding an optimal Bayesian network, to implement our generalized pruning rules and to find all near-optimal networks. Our experimental results demonstrate that the modified GOBNILP scales to significantly larger networks without resorting to restricting the structure of the Bayesian networks that are learned.

---

8 Obtained through correspondence with the author.

8 We are unable to generate BDeu score files for datasets with over 30 variables.
References


