Learning Bounded Tree-width Bayesian Networks via Sampling

Siqi Nie[1], Cassio P. de Campos[2], and Qiang Ji[1]

[1] Department of Electrical, Computer and Systems Engineering,
Rensselaer Polytechnic Institute, USA
[2] School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast, United Kingdom

Abstract. Learning Bayesian networks with bounded tree-width has attracted much attention recently, because low tree-width allows exact inference to be performed efficiently. Some existing methods [12, 14] tackle the problem by using k-trees to learn the optimal Bayesian network with tree-width up to k. In this paper, we propose a sampling method to efficiently find representative k-trees by introducing an Informative score function to characterize the quality of a k-tree. The proposed algorithm can efficiently learn a Bayesian network with tree-width at most k. Experiment results indicate that our approach is comparable with exact methods, but is much more computationally efficient.

Keywords: Bayesian network, structure learning, bounded tree-width

1 Introduction

Bayesian networks (BNs) are widely used probabilistic graphical models. Learning Bayesian networks from data has been widely studied in decades. In this paper we present our approach of score-based Bayesian network structure learning with some special constraint.

It is well known that the complexity of exact inference in a Bayesian network is related to the tree-width of the network [13]. To simplify the inference computation, one attempt that has received growing attention recently is to learn a Bayesian network with bounded tree-width. Moreover, some empirical results [10] demonstrate that bounding the tree-width of a Bayesian network achieves better generalization performance.

Several algorithms have been proposed to learn Bayesian networks with bounded tree-width. Korhonen and Parviainen [12] proposed a dynamic programming based algorithm for learning n-node Bayesian networks of tree-width at most k. Their algorithm guarantees to find the optimal structure maximizing a given score function subject to the tree-width constraint. Parviainen et al. [15] developed an integer programming approach to solve the problem. It iteratively creates a cutting plane on the current solution to avoid exponentially many constraints. However, both algorithms work only with small tree-widths. Berg et al. [3] transferred the problem into a weighted maximum satisfiability problem and solved it by weighted MAX-SAT solvers. Nie et al. [14] introduced an integer programming and a sampling methods to address this problem.

In this work, we present a novel method of score-based Bayesian network structure learning with bounded tree-width via sampling. We design an approximate approach based on sampling k-trees, which are the maximal graphs of tree-width k. The sampling method is based on a fast bijection between k-trees and Dandelion codes [5]. We design a sampling scheme, called *Distance Preferable Sampling* (DPS), in order to effectively cover the space of k-trees using limited samples, in which we give a larger probability for a sample in the unexplored area of the space, based on the existing samples. Smart rules to explore the sample space are essential, because we can only compute a few best structures respecting sampled k-trees in a reasonable amount of time. To evaluate the sampled k-trees, we design an *Informative Score* (I-score) function as the criterion for accepting or rejecting k-trees based on independence tests and BDeu scores, which is used as a prior information for the k-trees. Different from the method proposed in [14], this work focuses on identifying high quality k-trees, instead of uniformly sampling. Given each sampled k-tree, we employ the algorithm of [12] to find the optimal Bayesian network as a subgraph of it, which we denote as K&P method from now on.

This paper is structured as follows. We first introduce some definitions and notations for Bayesian networks and tree-width in Section 2. Then we discuss the proposed sampling method for learning Bayesian networks with bounded tree-width in Section 3. Experimental results are given in Section 4. Finally we conclude the paper in Section 5.

2 Preliminaries

2.1 Learning Bayesian Networks

A Bayesian network uses a directed acyclic graph (DAG) to represent a set of random variables $X = \{X_i : i \in N\}, N = \{1, 2, ..., n\}$ and their conditional (in)dependencies. Arcs of the DAG encode parent-child relations. Denote X_{pa_i} as the parent set of variable X_i. Conditional probability tables $p(x_i|x_{pa_i})$ are given accordingly, where x_i and x_{pa_i} are instantiations of X_i and X_{pa_i}. We consider categorical variables in this work.

The structure learning task of Bayesian network is to identify the "best" DAG from data. In this paper we consider the score-based Bayesian network structure learning problem, in which a score $s(G)$ is assigned to each DAG G. The commonly used score functions (such as BIC [17], and BDeu [4, 6, 11]) are decomposable, i.e., the overall score can be written as the summation of local score functions, $s(G) = \sum_{i \in N} s_i(X_{pa_i})$. For each variable, its score is only related to its parent set. We assume that local scores have been computed in advance and can be retrieved in constant time.

2.2 Learning BN with tree-width bound

The *width* of a tree decomposition of an undirected graph is the size of its largest clique minus one. The *tree-width* of an undirected graph is the minimum width among all possible tree decompositions of the graph. We define *tree-width* $tw(G)$ of a DAG G as the tree-width of its moral graph, which is obtained by connecting nodes with a common child, and making all edges undirected.

The objective of this work is to find a graph G^*,

$$G^* = \arg\max_{G} \sum_{i \in N} s_i(X_{pa_i}), \quad \text{s.t.} \quad tw(G) \leq k \,. \tag{1}$$

Directly computing the tree-width of a graph is intractable [1]. One way of imposing the tree-width constraint is to use the k-tree, the maximal graphs with tree-width k, and no more edges can be added to them without increasing the tree-width (see [16] for details). Therefore, every graph with tree-width at most k is a subgraph of a k-tree. Learning Bayesian network from a k-tree automatically satisfies the tree-width constraint if we ensure that the moral graph of the learned Bayesian network is a subgraph of the k-tree. A k-tree is denoted by $T_k \in \mathcal{T}_{n,k}$, where $\mathcal{T}_{n,k}$ is the set of all k-trees over n nodes.

3 Sampling k-trees using Dandelion codes

The basic idea is to efficiently search for k-trees with “high quality” and then use K&P algorithm to learn the optimal Bayesian network from the selected k-trees. This is accomplished in two steps. First, we propose a sampling method that can effectively cover the space of k-trees to obtain representative k-trees. Second, we establish an *informative score* (I-score) function to evaluate the quality of each k-tree.

3.1 Effective k-tree Sampling

Directly sampling a k-tree is not trivial. Caminiti et al. [5] proposed to establish a one-to-one correspondence between a k-tree and what is called *Dandelion* codes. The space of Dandelion codes is denoted by $\mathcal{A}_{n,k}$. A code $(Q, S) \in \mathcal{A}_{n,k}$ is a pair where $Q \subseteq N$ is a set of integers of size k and S is a $2 \times (n-k-2)$ matrix of integers drawn from $N \cup \{\epsilon\}$, where ϵ is an arbitrary number not in N (see [5] for details).

Dandelion codes can be sampled uniformly at random by a trivial linear-time algorithm that uniformly chooses k elements out of N to build Q, and then uniformly samples n–k–2 pairs of integers in $N \cup \{\epsilon\}$. Such property of Dandelion codes naturally makes a uniform prior for k-trees, which is a quite good prior in the absence of other prior knowledge [9]. However, uniform sampling generates each sample independently, and totally ignores previous samples, which makes it possible to generate the very same sample twice, or at least samples that are too close to each other. Considering the large size of the space of all Dandelion codes

($\binom{n}{k}(k(n-k)+1)^{n-k-2}$) and the relatively small amount of samples that we can process, we would prefer the samples to be as evenly distributed as possible. This is accomplished by generating the next sample from some currently unexplored area of the sampling space. Driven by this idea, we define the *Distance Preferable Sampling* (DPS). Given the samples of Dandelion codes $A^{(1)}, A^{(2)}, \cdots, A^{(j-1)}$ obtained so far, we want to decide how to sample the next $A^{(j)}$. A kernel density function for a new sample can be defined as

$$q(A^{(j)}) = \frac{1}{j-1}\sum_{i=1}^{j-1} K(\|A^{(j)} - A^{(i)}\|)\,, \tag{2}$$

where $A^{(j)} \in \mathcal{A}_{n,k}$ is the jth Dandelion code sample. $q(A^{(j)})$ depends on all the previous samples, with its value decreasing as $A^{(j)}$ moves away from existing samples. $K(\cdot)$ is a kernel function, (e.g., a Gaussian). The distance between two Dandelion codes is defined as

$$\|A^{(j)} - A^{(i)}\| = \|Q^{(j)} - Q^{(i)}\|_2 + \|S^{(j)} - S^{(i)}\|_{2,1}\,, \tag{3}$$

where $\|\cdot\|_2$ is the L_2 norm. $S^{(j)}$ is processed as a $2 \times (n-k-2)$ matrix, and $\|\cdot\|_{2,1}$ is the $L_{2,1}$ norm.

Since we intend to explore the regions which have not yet been sampled, we design a proposal distribution as follows:

$$p(A^{(j)}) = 1 - \frac{q(A^{(j)})}{K(0)}\,. \tag{4}$$

$p(A^{(j)})$ increases as sample $A^{(j)}$ moves away from all the existing samples. Following the proposal distribution, we use the rejection sampling algorithm (Algorithm 1) to generate a sample of Dandelion codes, and then employ the implementation of [5] to decode it into a k-tree.

3.2 Informative Score for k-trees

Given a k-tree, the computational complexity of the method of [12] for constructing a Bayesian network subject to the k-tree is super-exponential in k ($O(k \cdot 3^k \cdot (k+1)! \cdot n)$). Hence, one cannot hope to use it with too many k-trees, given current computational resources. Instead of learning from every k-tree without distinction, we define the I-score function to evaluate how well a k-tree

Algorithm 1 Sampling a Dandelion code using Distance Preferable Sampling

```
Input  Previous samples of Dandelion codes $A^{(1)}, \ldots, A^{(j-1)}$.
Output  a new sample of Dandelion code $A^{(j)}$.
1 Uniformly sample a Dandelion code $A^{(j)}$ in the feasible region;
2 If $j = 1$, the sample is accepted. If not, the sample is accepted with probability
  $p(A^{(j)})$;
3 If $A^{(j)}$ is rejected, return to step 1 for another sample, until a sample is accepted.
```

"fits the data", hence can produce a Bayesian network with high quality. The I-score of a k-tree T_k is defined as

$$IS(T_k) = \frac{S_{mi}(T_k)}{|S_l(T_k)|} \,. \tag{5}$$

The numerator, $S_{mi}(T_k)$, measures how much information is lost by representing data using the k-tree. Let e_{ij} denote the edge connecting node i and j, and let I_{ij} denote the mutual information of node i and j. Then,

$$S_{mi}(T_k) = \sum_{i,j} I_{ij} - \sum_{e_{ij} \notin T_k} I_{ij} \,. \tag{6}$$

If an edge e_{ij} is not included in the k-tree, we subtract the mutual information corresponding to that edge from the optimal score. S_{mi} is a measurement of the consistency of the k-tree and the data, and can be interpreted either as the sum of the mutual information covered by the k-tree or as constant minus the sum of the mutual information lost by the k-tree. Larger S_{mi} indicates the k-tree fits the data well, from the independent test perspective.

On the other hand, the denominator $S_l(T_k)$ is defined as the score (e.g., BIC, BDeu scores) of the best pseudo subgraph of the k-tree by dropping the acyclic constraint.

$$S_l(T_k) = \max_{m(G) \subseteq T_k} \sum_{i \in N} s_i(x_{pa_i}) \,, \tag{7}$$

where $m(G)$ is the moral graph of DAG G, and $s_i(x_{pa_i})$ is the local score function for x_i given parent set x_{pa_i}.

The best pseudo subgraph of a k-tree is constructed by choosing the best parent set for each node in terms of local scores, compatible with the k-tree, in a greedy way. Combining all the parent sets will result in a directed, possibly cyclic, graph. Therefore, given the pre-computed scores for each variable, score S_l can be computed in linear time. Since the value of S_l is negative, for practical reasons we use the term $1/|S_l(T_k)|$ in the I-score formulation.

The I-score for a k-tree combines the independence test approach and score-based approach for learning Bayesian networks. It can be very efficiently evaluated for any given k-tree, as computing S_{mi} requires only mutual information of pairs of nodes (which can all be pre-computed, so time complexity is at most $O(n^2)$ over all multiple runs of the algorithm).

With the I-score for a proposed k-tree, we then accept a k-tree with probability

$$\alpha = \min\left(1, \frac{IS(T_k)}{IS(T_k^*)}\right) \,, \tag{8}$$

where T_k^* is the current k-tree with the largest I-score. Notice that we do not set a hard constraint for accepting or rejecting a k-tree, due to the fact that even for a k-tree with relatively small I-score, it is still possible for it to contain a good subgraph.

Algorithm 2 Learning a Bayesian network structure of bounded tree-width by sampling Dandelion codes.

```
Input score function $s_i$, $\forall i \in N$, mutual information $I_{ij}$, $\forall i, j \in N$
Output a DAG $G^{\text{best}}$.
1 Initialize $Pa_i^{\text{best}}$ as an empty set for all $i \in N$;
2 (Rejection Procedure 1) Sample a Dandelion code $(Q, S) \in \mathcal{A}_{n,k}$ according to Algorithm 1;
3 (Rejection Procedure 2) Decode $(Q, S)$ into $T_k \in \mathcal{T}_{n,k}$, accept it with probability $\alpha$ (Equation 8);
4 Repeat Step 2 and 3 until $m$ $k$-trees are accepted. Sort them in descending order based on their I-scores. From the top use the implementation of [12] to learn a Bayesian network. Keep the structure with the highest BDeu score.
5 If time limit is not reached after $m$ $k$-trees, restart from step 2.
```

3.3 BN Learning from Sampled *k*-trees

Combining the ideas in Sections 3.1 and 3.2, we present Algorithm 2 as an approximate algorithm for learning Bayesian networks of bounded tree-width. Due to the fact that k-trees with large I-scores are more likely to have better subgraphs, we give them high priority to learn the corresponding Bayesian network. This is reflected in Step 4 of Algorithm 2. A certain amount of k-trees are sampled, and then sorted based on their I-scores. The process starts with the k-trees of the largest I-score in the sorted list. If time allows, all k-trees are examined, and the procedure restarts. Given a k-tree as the super structure, the implementation of K&P is employed to learn the optimal Bayesian network. The goal of Algorithm 2 is to restrict the calls to K&P (which is a time consuming method in k, even if linear in n) only to k-trees that are promising.

4 Experiments

To empirically evaluate our method, we use a collection of data sets from the UCI repository [2] of varying dimensionality. Table 1 contains the details about the data sets used in the experiments. Firstly, we show the effectiveness of the I-score for accepting or rejecting a sampled k-tree. Secondly, we compare the BDeu scores of the learned Bayesian networks.

Table 1: Dimensions of data sets.

DATASET	nursery	breast	housing	adult	zoo	letter	mushroom	wdbc
VAR.	9	10	14	15	17	17	22	31
SAMPLES	12960	699	506	32561	101	20000	8124	569

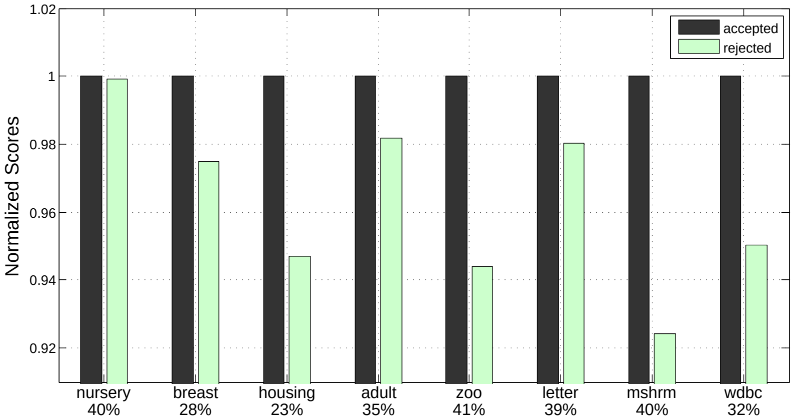

Fig. 1: Effect of the rejection process. The maximum BDeu scores of the Bayesian networks learned from the accepted k-trees, compared with those from the rejected k-trees. Best scores are normalized to 1. The rejection rates are presented at bottom.

4.1 Informative Score

In this section, we evaluate the I-score as a measurement of how good a k-tree would be to "produce" a Bayesian network (moralized) structure as its subgraph. Eight data sets are used (*nursery, breast, housing, adult, zoo, letter, mushroom,* and *wdbc*), whose dimensions are summarized in Table 2 and 3. The numbers of samples vary from 100 to 20,000. Non-binary variables are binarized over the median value. In all experiments, we maximize the Bayesian Dirichlet equivalent uniform (BDeu) score with equivalent sample size equal to one [11]. To evaluate the effect of our rejection of k-trees, we sampled 500 k-trees, and counted the number of rejections during the k-tree selection (Step 3 in Algorithm 2). If a k-tree is rejected, we still compute the BDeu score of its optimal Bayesian network for comparison. Figure 1 shows the ratio of rejection (at bottom) and relation between best scores of Bayesian networks learned from both the accepted and the rejected k-trees. The scores are normalized so that best score is 1. In all data sets, BDeu scores of Bayesian network learned from rejected k-trees never exceeded the scores from accepted ones. Using the rejection process, we see that 20% to 40% of the k-trees were rejected. Such variation in the rejection rates is due to the randomness of the samples, because if a k-tree with high I-score is sampled in an early stage, later samples have a high probability to be rejected.

4.2 Bayesian Network Learning

In this section we compare the BDeu scores of structures learned by our method against scores from two exact methods as baseline methods, namely, the K&P

Table 2: Computational time of the K&P method to find the optimal Bayesian network structure, and the proposed method to sample 100 k-trees, as well as the resulting BDeu scores of the networks found by both methods. Empty cells indicate that the method failed to solve the problem because of excessive memory consumption. s, m mean seconds and minutes, respectively.

		Time				Score			
Method	k	nursery	breast	housing	adult	nursery	breast	housing	adult
		n=9	n=10	n=14	n=15	n=9	n=10	n=14	n=15
	2	7s	26s	128m	137m	-72160	-2688.4	-3295.4	-201532
K&P	3	72s	5m	–	–	-72159	-2685.8	–	–
	4	12m	103m	–	–	-72159	-2685.3	–	–
	5	131m	–	–	–	-72159	–	–	–
	2	5s	8s	16s	18s	-72218	-2690.5	-3409.6	-202852
Proposed	3	70s	76s	3m	4m	-72204	-2692.5	-3413.4	-204186
	4	9m	10m	36m	50m	-72159	-2691.9	-3285.0	-202432
	5	80m	232m	631m	896m	-72159	-2694.0	-3296.9	-202699

algorithm[3] and the B&B method[4] [7, 8]. The comparison with exact methods allows us to evaluate the proposed algorithm in terms of the difference in scores.

Due to the complexity of K&P method, it is only applicable to some relatively small data sets, hence our comparisons are restricted to those cases. The detailed computational time that K&P uses is given in Table 2. The algorithm is run using a desktop computer with 64GB of memory. Maximum number of parents is set to three. Due to the huge amount of memory cost, for *housing* and *adult* data sets with tree-width more than 2, as well as *breast* with tree-width bound 5, the algorithm failed to give a solution. Correspondingly, we sampled 100 k-trees and recorded the running time for the proposed algorithm to give a solution, given the same data set and the same choice of maximum tree-width. The BDeu scores of the best Bayesian networks found with both algorithms are also presented. By examining only a small portion of k-trees, the proposed algorithm finds solutions with an BDeu score difference less than 1% for most cases. Only in the *housing* data set with tree-width equal to 2, our algorithm have a 3% score difference to the exact solution, which is reasonable after only 16 seconds of computation. Generally speaking, the proposed algorithm achieves comparable results to those of the exact method in terms of BDeu score difference. Yet when considering the time and memory costs of the exact solution, the proposed algorithm is more efficient against the competing method by several orders of magnitude.

Besides efficiency, the proposed algorithm can be used on larger data sets with up to 31 nodes and larger values for the tree-width bound (*zoo, letter, mushroom*, and *wdbc*) (Table 3). Note that the B&B method does not have the tree-width constraint, so the learned structures are supposed to have larger BDeu

[3] http://www.cs.helsinki.fi/u/jazkorho/aistats-2013/

[4] http://www.ecse.rpi.edu/∼cvrl/structlearning.html

Table 3: BDeu scores for relatively larger data sets and lager tree-widths, compared with the B&B method without tree-width constraint. Running time is ten minutes. Averaged over ten repetitions.

data set	nodes	k=2	k=3	k=4	k=5	B&B
zoo	17	-644.1	-623.8	-609.1	-649.1	-565.2
letter	17	-195677	-192289	-192373	-194349	-184530
mushroom	22	-73697	-74367	-68523	-73902	-68237
wdbc	31	-8435.1	-8320.8	-8352.1	-8316.9	-6933.8

Table 4: BDeu scores of BNs learned using different sampling methods with data set *letter*, normalized using the best score of each column. UNI means uniform sampling; DPS means Distance Preferable Sampling; α means that we employed the acceptance probability α. Larger numbers indicate worse performance.

Method	k=2	k=3	k=4
UNI	1.019	1.046	1.039
DPS	1.018	1.045	1.038
DPS+α	1	1	1

scores. However, the score difference is not very significant, which indicates the bounding the tree-width can learn good structures in terms of scores.

To further study the benefit of the DPS and I-score based sampling, we also implemented the algorithm using the uniformly sampled Dandelion codes without sorting or rejection. The BDeu scores on the *letter* data set are compared, with different choices of tree-widths. According to Table 4, DPS outperforms uniform sampling, even if by a small margin. A great portion of the gain of performance is from rejecting k-trees based on I-scores.To summarize, we are able to focus on better k-trees by employing non-uniform sampling and sorting them according to some meaningful measure.

5 Conclusion

In this paper we present a sampling method for learning Bayesian networks with bounded tree-width. The sampling is based on a bijection between Dandelion codes and k-trees. We design a Distance Preferable Sampling scheme to effectively cover the space of k-trees, as well as an Informative score function to evaluate each k-tree. These ideas allow to quickly find representative k-trees of high quality. Experiments indicate that the proposed method reaches comparable accuracy to the exact algorithms in terms of BDeu scores, but is much more efficient in terms of learning speed, and can scale up to larger networks and larger tree-widths.

Acknowledgements This work is supported in part by the grant N00014-12-1-0868 from the US Office of Navy Research.

References

[1] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in ak-tree. *SIAM Journal on Algebraic Discrete Methods*, 8(2):277–284, 1987.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

[3] J. Berg, M. Järvisalo, and B. Malone. Learning optimal bounded treewidth bayesian networks via maximum satisfiability. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pages 86–95, 2014.

[4] W. Buntine. Theory refinement on Bayesian networks. In *Proc. 7th Conf. on Uncertainty in AI*, pages 52–60, 1991.

[5] S. Caminiti, E. G. Fusco, and R. Petreschi. Bijective linear time coding and decoding for k-trees. *Theory of Comp. Systems*, 46(2):284–300, 2010.

[6] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Mach. Learning*, 9(4):309–347, 1992.

[7] C. P. de Campos and Q. Ji. Efficient structure learning of bayesian networks using constraints. *The Journal of Machine Learning Research*, 12:663–689, 2011.

[8] C. P. de Campos, Z. Zeng, and Q. Ji. Structure learning of Bayesian networks using constraints. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 113–120, Montreal, Quebec, Canada, 2009.

[9] D. Eaton and K. Murphy. Bayesian structure learning using dynamic programming and mcmc. In *Proc. 23rd Conf. on Uncertainty in AI*, pages 101–108, 2007.

[10] G. Elidan and S. Gould. Learning Bounded Treewidth Bayesian Networks. *J. of Mach. Learning Res.*, 9:2699–2731, 2008.

[11] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Mach. Learning*, 20(3):197–243, 1995.

[12] J. H. Korhonen and P. Parviainen. Exact Learning of Bounded Tree-width Bayesian Networks. In *Proc. 16th Int. Conf. on AI and Stat.*, pages 370–378, 2013. JMLR W&CP 31.

[13] J. H. P. Kwisthout, H. L. Bodlaender, and L. C. van der Gaag. The Necessity of Bounded Treewidth for Efficient Inference in Bayesian Networks. In *Proc. 19th European Conf. on AI*, pages 237–242, 2010.

[14] S. Nie, D. D. Mauá, C. P. de Campos, and Q. Ji. Advances in learning bayesian networks of bounded treewidth. In *Advances in Neural Information Processing Systems*, pages 2285–2293, 2014.

[15] P. Parviainen, H. S. Farahani, and J. Lagergren. Learning bounded tree-width bayesian networks using integer linear programming. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pages 751–759, 2014.

[16] H. P. Patil. On the structure of k-trees. *J. Combin. Inform. System Sci*, 11(2-4): 57–64, 1986.

[17] G. Schwarz. Estimating the dimension of a model. *Annals of Stat.*, 6(2):461–464, 1978.