Learning Bayesian Network Structure using LP Relaxations

Tommi Jaakkola MIT CSAIL
David Sontag MIT CSAIL
Amir Globerson Hebrew University
Marina Meila University of Washington

Abstract

We propose to solve the combinatorial problem of finding the highest scoring Bayesian network structure from data. This structure learning problem can be viewed as an inference problem where the variables specify the choice of parents for each node in the graph. The key combinatorial difficulty arises from the global constraint that the graph structure has to be acyclic. We cast the structure learning problem as a linear program over the polytope defined by valid acyclic structures. In relaxing this problem, we maintain an outer bound approximation to the polytope and iteratively tighten it by searching over a new class of valid constraints. If an integral solution is found, it is guaranteed to be the optimal Bayesian network. When the relaxation is not tight, the fast dual algorithms we develop remain useful in combination with a branch and bound method. Empirical results suggest that the method is competitive or faster than alternative exact methods based on dynamic programming.

1 Introduction

Bayesian networks and their many extensions (e.g., relational models) are common tools in multi-variate data analysis across disciplines. Their success follows in large part from the insights about the problem being modeled that can be derived directly from the associated directed acyclic graph (DAG) structure. Since this DAG is not often known apriori, one often needs to learn it from data. A common approach to this structure learning problem is to rank graph structures via a scoring metric that measures how well each model structure fits the data. When the available data may be drawn from an arbitrary distribution, the resulting combinatorial search problem is known to be NP-hard [Chickering, 1996; Chickering *et al.* , 2004]. The search problem remains intractable even if we limit the number of parents of each node in the graph to be at most 2 (the case of one parent per node – directed trees – is easy to solve). The hardness result extends even to poly-trees with at most two parents [Dasgupta, 1999].

The core difficulty of the learning problem arises from the fact that a valid graph has to be acyclic. The acyclic constraint is global and ties together the choices of parents for each node in the graph. A wealth of structure learning methods have been developed to address this difficulty. These methods are generally divided into exact methods based on dynamic programming ideas and extensions (e.g., Koivisto & Sood [2004]; Silander & Myllymäki [2006]; Parviainen & Koivisto [2009]) or approximate methods based on local or stochastic search. Without additional constraints, exact methods are limited to relatively small problems (around 30 nodes) as both computation and memory requirements scale exponentially with the number of nodes in the graph. Local search methods, on the other hand, remain applicable to a broader class of structure learning problems but, in contrast, fail to guarantee optimality. Significant improvements to these methods were obtained by either searching over equivalence classes of network structures [Chickering, 2002] or casting the search over different orderings of the variables [Friedman & Koller, 2003; Teyssier & Koller, 2005]. More recently, branch-and-bound has been applied to exact learning of Bayesian network structure [de Campos *et al.* , 2009]. Besides guaranteeing optimality at the termination of the search, the approach maintains an estimate of how far the current solution is from the optimal structure.

Independence tests provide an alternative paradigm to the above score based structure learning approaches [Spirtes *et al.* , 2001]. If the data is truly drawn from a distribution that has exactly the same conditional independencies as some Bayesian network, independence tests can be used to provably recover the true distribu-

Appearing in Proceedings of the 13^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy. Volume 9 of JMLR: W&CP 9.

tion. If the Bayesian network has bounded in-degree, this approach uses both polynomial time and requires only a polynomial amount of data. However, applying this method to real-world data is difficult, both because the outcomes of the independence tests may be inconsistent as well as because the data generating distributions typically do not satisfy the underlying distributional assumptions.

We propose to solve the combinatorial problem of finding the highest scoring Bayesian network structure by first formulating the problem as a Linear Program (LP) over the polytope defined by valid acyclic graphs, and subsequently using an outer bound approximation to the polytope. In contrast to stochastic or greedy local search, this is a *global* solution method where the LP relaxation upper bounds the score of the optimal structure. If the solution to the relaxed problem is integral, we are guaranteed that it is the optimal structure. Otherwise, we can use the fractional solution to guide branch-and-bound towards finding the optimal structure. Guo & Schuurmans [2006] also proposed an LP approximation for structure learning, but solving it required semidefinite programming. Furthermore, the outer bounds we use here are novel and include those of Guo & Schuurmans [2006] as a special case.

We solve the LP relaxation using a simple coordinate-descent algorithm in the dual. The method is fast and, in combination with dual-subgradient steps, effectively avoids getting stuck in poor solutions. The LP relaxation is iteratively tightened in a cutting plane fashion by searching over a new class of valid inequalities. The gain from each additional inequality constraint (if violated) is easily assessed in the dual. Moreover, the dual formulation maintains an upper bound on the score of the optimal structure at all times and this can be exploited in guiding branch and bound partitions.

2 Background

Our goal is to find the highest scoring ("optimal") Bayesian network structure G given a dataset D. We assume that the scoring metric is decomposable in the sense that it can be written as a sum of terms where each term depends only on the choice of parents $pa_i = s_i$ for each node i in the Bayesian network. Here s_i is a set valued variable specifying the parents of i. Most typical scoring metrics, including the BDe metric [Heckerman *et al.* , 1995] and BIC, are decomposable and can be written as

$$\text{score}(G; D) = \sum_{i=1}^{n} \text{score}(i|pa_i = s_i; D) = \sum_{i=1}^{n} W_i(s_i) \quad (1)$$

where we have adopted a shorthand $W_i(s_i)$ for the local scores and n is the number of random variables. Finding the best structure $\hat{G} = G(\hat{s})$ corresponds to finding the best choice of parent sets $\hat{s} = \{\hat{s}_1, \ldots, \hat{s}_n\}$.

The key difficulty in maximizing $\text{score}(G(s); D)$ with respect to s stems from the fact that $G(s)$ has to be acyclic. As a result, the parent sets $s_1, \ldots, s_n$ cannot be chosen independently from each other. Most of our paper is focused on dealing with this constraint.

The other difficulty has to do with a potentially large number of values that each s_i can take (all subsets of the other variables). This set can be be pruned effectively in practice, however. For example, if $W_i(s_i) \geq W_i(s_i')$ for $s_i \subset s_i'$, then we never have to consider s_i' (cf. de Campos *et al.* [2009]). Choosing a larger set of parents s_i' would merely impose stronger constraints on the other variables without a commensurate gain in the score. We will also typically limit *a priori* the maximum number of parents allowed for each variable.

Let $P_a(i)$ denote the collection of already pruned parent sets for node i. Note that $P_a(i)$ is necessarily based on the Markov Blanket of i and will also include variables that are not parents of i.

3 Structure learning as an LP

In what follows, we cast the structure learning problem as a linear program over a polytope $\mathcal{P}$ of valid acyclic structures. The polytope is defined as a convex hull of vertices where each vertex corresponds to a valid acyclic structure. The vertices are represented as binary vectors where each coordinate corresponds to a possible choice of parents for a variable. More specifically, we represent an acyclic graph with a binary vector $\boldsymbol{\eta} = [\boldsymbol{\eta}_1; \ldots; \boldsymbol{\eta}_n]$ where each $\boldsymbol{\eta}_i$ is an indicator vector (of dimension $|P_a(i)|$) specifying the parent set chosen for the corresponding node. In other words, if node i selects parents s_i, then $\eta_i(s_i) = 1$ and all the remaining coordinates of $\boldsymbol{\eta}_i$ are zero. Note also that $\boldsymbol{\eta}$ is a sparse vector with exactly n coordinates equal to one. We use $\boldsymbol{\eta}(s)$ to denote the binary vector corresponding to the graph $G(s)$ obtained by selecting parent sets $s = [s_1, \ldots, s_n]$.

The polytope $\mathcal{P}$ is now the convex hull of all $\boldsymbol{\eta}(s)$ where $s_i \in P_a(i)$ and $s = [s_1, \ldots, s_n]$ correspond to a DAG.[1] The key property of this polytope is that $\boldsymbol{\eta}(s)$ for any graph $G(s)$ *with cycles* is guaranteed to lie outside $\mathcal{P}$.

With a slight abuse of notation, we will use $\boldsymbol{\eta}$ also for the interior points $\boldsymbol{\eta} \in \mathcal{P}$ that correspond to weighted averages of binary vectors representing acyclic graphs. We are now ready to state the structure learning prob-

[1] Note that the dimension and structure of the polytope are different for different pruning strategies.

lem as a linear program:

$$\begin{array}{ll} \max & \boldsymbol{\eta} \cdot \boldsymbol{W} = \sum_{i=1}^{n} \sum_{s_i \in P_a(i)} \eta_i(s_i)\, W_i(s_i) \\ \text{s.t.} & \boldsymbol{\eta} \in \mathcal{P} \end{array} \tag{2}$$

The optimal value of this linear program is obtained at a vertex that corresponds to the highest scoring Bayesian network. The complexity of the structure learning problem is now entirely hidden in the exponentially many facets (linear half-space constraints) that are needed to specify $\mathcal{P}$.

We remark that $\mathcal{P}$ defined above is different from the *acyclic subgraph polytope* $\mathcal{P}_{dag}$ studied extensively in polyhedral combinatorics. $\mathcal{P}_{dag}$ is defined as the convex hull of the edge indicator vectors for every set of edges specifying an acyclic graph [Grötschel *et al.* , 1985]. However, the score associated with a Bayesian network is not a linear function of individual edge selections, but rather depends on the set of incoming edges (parent sets). Thus $\mathcal{P}_{dag}$ would not suffice to cast the structure learning problem as a linear program.

4 LP Relaxation

We seek to relax the linear program by finding an outer bound approximation to the polytope $\mathcal{P}$. We identify here two strategies for relaxing the polytope: first, by projecting $\boldsymbol{\eta} \in \mathcal{P}$ to a known polytope defined over the choice of directed edges, and, second, introducing a new class of constraints directly outer bounding $\mathcal{P}$.

Any point (interior or vertex) $\boldsymbol{\eta} \in \mathcal{P}$ corresponds to a distribution over parent set choices reflecting one or more acyclic graphs. Based on $\boldsymbol{\eta}$, we can easily calculate the probability that any directed edge such as (j, i) (an edge from j to i) is present, i.e.

$$\mu_{ji} = \sum_{s_i \in P_a(i)} \eta_i(s_i) \delta(j \in s_i), \tag{3}$$

where $\delta(j \in s_i)$ is an indicator function. By concatenating all such μ_{ji} into a vector $\boldsymbol{\mu}$ of directed edge selections, we have defined a linear projection from $\boldsymbol{\eta} \in \mathcal{P}$ to the acyclic subgraph polytope $\boldsymbol{\mu} \in \mathcal{P}_{dag}$. Any known facet of $\mathcal{P}_{dag}$ can consequently be introduced as a constraint on $\boldsymbol{\eta}$ by lifting. In particular, cycle inequalities of the form

$$\sum_{(j,i) \in E_C} \mu_{ji} \leq |E_C| - 1, \tag{4}$$

where a cycle is represented as a sequence of directed edges E_C, are facet defining though not sufficient for specifying $\mathcal{P}_{dag}$ [Grötschel *et al.* , 1985]. The corresponding lifted constraint on $\boldsymbol{\eta}$ is obtained by expanding the definition of μ_{ji} to obtain:

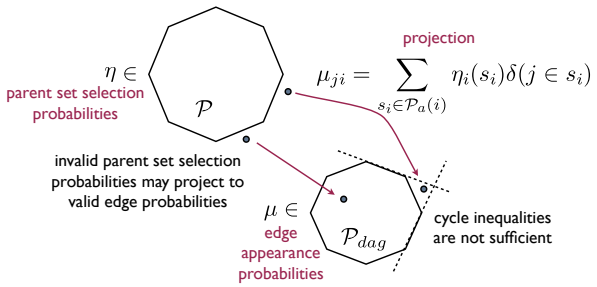

Figure 1: Projection and lifting between parent set selections and edge selections.

$$\sum_{(j,i) \in E_C} \sum_{s_i \in P_a(i)} \eta_i(s_i) \delta(j \in s_i) \leq |E_C| - 1 \tag{5}$$

We call the polytope over the parent set selections arising from all such lifted cycle inequalities, together with the simple constraints $\eta_i(s_i) \geq 0$ and $\sum_{s_i} \eta_i(s_i) = 1$, the cycle relaxation $\mathcal{P}_{cycle}$. It can be shown that these cycle inequalities are equivalent to the transitivity constraints used in Guo & Schuurmans [2006].

The cycle relaxation is not tight in general for two reasons (see Figure 1). First, cycle inequalities generally provide an outer bound on $\mathcal{P}_{dag}$ that is not tight. Thus, they permit marginal edge selections that do not arise as marginals from any valid distribution over directed acyclic graphs (DAGs). Note, however, that there are cases when cycle inequalities are provably exact. For example, when G is planar (i.e., can be drawn on a plane without crossing edges), the cycle inequalities exactly define $\mathcal{P}_{dag}$ [Grötschel *et al.* , 1985]. The setting rarely occurs in practice when $n > 4$. Other facet defining inequalities for $\mathcal{P}_{dag}$ are known, such as the fence inequalities [Grötschel *et al.* , 1985]. Such constraints involve edge selection variables that correspond to a non-planar graph (at least 5 nodes).

The second and more subtle reason for why the above cycle relaxation is not tight is that the parent set selection variables $\eta_i(s_i)$, which were necessary to formulate a linear objective, couple the edge variables. Rather than select each parent of i independently, we are forced to make coordinated selections as specified by each $s_i \in P_a(i)$. Thus, the edge selection marginals $\boldsymbol{\mu}$ resulting from any distribution over s_i (such as $\eta_i(s_i)$) are dependent, even for two edges $j \to i$ and $k \to i$ that cannot form a cycle. The acyclic subgraph polytope $\mathcal{P}_{dag}$ only represents $\boldsymbol{\mu}$ and thus does not consider correlated choices of edges.

We illustrate this with the following example. Consider estimating a Bayesian network over three binary

variables, y_1, y_2, and y_3, which are related only by having an even parity. As a result, the scores for possible parent sets are symmetric where each variable prefers the other two as parents. The maximizing solution to the cycle relaxation would be:

$$\eta_1(s_1) = 1/2, s_1 = \{2,3\}, \quad \eta_1(s_1) = 1/2, s_1 = \{\emptyset\}$$
$$\eta_2(s_2) = 1/2, s_2 = \{1,3\}, \quad \eta_2(s_2) = 1/2, s_2 = \{\emptyset\}$$
$$\eta_3(s_3) = 1/2, s_3 = \{1,2\}, \quad \eta_3(s_3) = 1/2, s_3 = \{\emptyset\}$$

The resulting edge marginals μ_{ji} are all $1/2$ and clearly satisfy the cycle inequalities $1/2 + 1/2 + 1/2 \leq 2$ and similarly for 2-cycles. The di-graph of possible edges is a triangle, therefore planar, and the cycle inequalities fully specify $\mathcal{P}_{dag}$. However, the solution is not a valid distribution over DAGs, but a fractional vertex that lies outside $\mathcal{P}$.

4.1 A new class of valid constraints

The above discussion implies that one needs to go beyond standard $\mathcal{P}_{dag}$ constraints to obtain tight relaxations of $\mathcal{P}$. Here we introduce such a class of additional constraints, and later show how they can be optimized over (see Section 5).

Given a subset (or cluster) of nodes $C \subseteq V$, we consider the following linear constraint on $\boldsymbol{\eta}$:

$$(c1) \quad \sum_{i \in C} \sum_{s_i \in P_a(i)} \eta_i(s_i) I_C(s_i) \geq 1 \tag{6}$$

where $I_C(s_i)$ is an indicator function for $C \cap s_i = \emptyset$, i.e., that the parent set selection s_i either lies outside the cluster C or is the empty set. The constraint enforces the fact that, in an acyclic graph, any subset of nodes must have at least one node whose parents lie outside the cluster. The constraint subsumes all lifted cycle inequalities for cycles of length $|C|$ within the cluster.

For a set $\mathcal{C}$ of clusters, we define the polytope $\mathcal{P}_{cluster}(\mathcal{C})$ to be the set of $\boldsymbol{\eta}$ that satisfy $(c1)$ for all $C \in \mathcal{C}$, as well as the simple constraints $\eta_i(s_i) \geq 0$, $\sum_{s_i} \eta_i(s_i) = 1$. For the case where $\mathcal{C}$ is all subsets of V, we denote the resulting polytope by $\mathcal{P}_{cluster}$. Although $\mathcal{P}_{cluster}$ is generally not equal to $\mathcal{P}$, it is strong enough to fully specify $\mathcal{P}$ when the Bayesian networks are restricted to the class of *branching programs* or directed trees. In this special case, where each variable is restricted to have either zero or one parent, $\mathcal{P}$ is equivalent to the directed *minimum spanning tree polytope*, which is fully characterized by the (c1) constraints Magnanti & Wolsey [1995]. Thus, these new constraints provide a substantially stronger guarantee than that given by the cycle relaxation.

In what follows, we provide an algorithm that uses such cluster-based constraints for approximating the structure learning problem.

5 Dual Bound Optimization

Our goal is to approximate the exact structure learning LP in Eq. 2 by replacing $\mathcal{P}$ with an outer bound that is as tight as possible. The previous discussion suggests replacing $\mathcal{P}$ with $\mathcal{P}_{cluster}$ yielding the following LP:

$$\max_{\boldsymbol{\eta} \in \mathcal{P}_{cluster}} \boldsymbol{\eta} \cdot \boldsymbol{W} \tag{7}$$

Exact optimization of Eq. 7 is generally not feasible, since $\mathcal{P}_{cluster}$ contains an exponential number of inequalities. However, as we show next, using the dual of Eq. 7 allows us to often solve this problem in practice. Our optimization scheme is similar in structure to column generation and iterative constraint generation approaches to approximate inference [Sontag *et al.* , 2008]. We begin by considering the dual of Eq. 7. The dual variables in this case are λ_C (one per cluster) and the dual itself is given by:

$$\begin{array}{ll} \min & \sum_{i=1}^{n} \max_{s_i \in P_a(i)} \left[W_i(s_i) + \sum_{C: i \in C} \lambda_C I_C(s_i) \right] - \sum_C \lambda_C \\ \text{s.t.} & \lambda_C \geq 0, \forall C \subseteq V \end{array} \tag{8}$$

Note that the constraints are simple non-negativity constraints on λ_C. Furthermore, setting $\lambda_C = 0$ means that we are not enforcing the corresponding constraint.

Since there are exponentially many λ_C variables, we do not optimize over all of them. Instead, we set all λ_C to zero except for $C \in \mathcal{C}$ for some set of $\mathcal{C}$, and increase $\mathcal{C}$ gradually. The algorithm proceeds as follows:

1. Update λ_C for all $C \in \mathcal{C}$ so that the dual objective is decreased.
2. Decode a DAG from the current values of λ. If its score is equal to the dual objective, we have solved the problem exactly.
3. If solution is not exact, choose a new cluster to add to $\mathcal{C}$ and go back to step 1.

The above steps are repeated until the problem is solved or the set $\mathcal{C}$ becomes too large. We initialize $\mathcal{C}$ to $\emptyset$. The following sections describe these steps in more detail.

5.1 Dual Updates on λ_C

To decrease the dual objective, we perform coordinate descent on the λ_C (e.g., see Sontag *et al.* [2008]). This turns out to correspond to simple closed form updates, derived below.

If we fix $\lambda_{C'}$, $C' \neq C$, then the part of the objective pertaining to λ_C is given by

$$J_C(\lambda_C) = \sum_{i \in C} \max_{s_i \in P_a(i)} [W_{C;i}(s_i) + \lambda_C I_C(s_i)] - \lambda_C \tag{9}$$

where $W_{C;i}(s_i) = W_i(s_i) + \sum_{C' \neq C: i \in C'} \lambda_{C'} I_{C'}(s_i)$. The minimum of $J_C(\lambda_C)$ can be obtained explicitly. To this end, we can maximize over s_i conditioned on the value of the cluster indicator and get:

$$W^1_{C;i} = \max_{s_i \in P_a(i):\, I_C(s_i)=1} W_{C;i}(s_i) \quad (10)$$

$$W^0_{C;i} = \max_{s_i \in P_a(i):\, I_C(s_i)=0} W_{C;i}(s_i) \quad (11)$$

If the maximization is over an empty set, the corresponding value is $-\infty$. We subsequently sort the differences $\delta_i = W^0_{C;i} - W^1_{C;i}$, $i \in C$, so that $\delta_{i_1} \leq \delta_{i_2} \leq \cdots \leq \delta_{i_{|C|}}$. In the absence of the non-negativity constraint for λ_C, the minimum value of the piecewise linear function $J_C(\lambda_C)$ would be obtained within the interval $[\delta_{i_1}, \delta_{i_2}]$ (the value is constant within this interval). The constrained optimum can thus be chosen to be $\lambda_C = \max\{(\delta_{i_1} + \delta_{i_2})/2,\, 0\}$.

The above coordinate descent algorithm may get stuck in suboptimal λ (since the objective is not strictly convex). For this reason, in optimizing the dual, we alternate between the block coordinate updates above and the following normalized subgradient steps:

$$\lambda_C \leftarrow \lambda_C + \epsilon \;\text{ if }\; \sum_{i \in C} I_C(\hat{s}_i) = 0 \quad (12)$$

$$\lambda_C \leftarrow \max\{\lambda_C - \epsilon, 0\} \;\text{ if }\; \sum_{i \in C} I_C(\hat{s}_i) > 1 \quad (13)$$

where $\hat{s}_i$ is any parent set choice that maximizes $W_i(s_i) + \sum_{C': i \in C'} \lambda_{C'} I_{C'}(s_i)$. The variation in the solution introduced by the subgradient steps also helps in identifying violated constraints. We decrease the step size ϵ after each round of adding violated cluster constraints according to $\epsilon_t = \epsilon_0/\sqrt{t}$.

5.2 Decoding a DAG

Once we have solved the dual problem, we would like to obtain the corresponding primal solution. When the relaxation is not tight, we would still like to obtain an approximate integral solution, a particular setting of the parent set variables $s_1, \ldots, s_n$. This integral solution is also used as a lower bound in the branch and bound algorithm discussed below. We provide here a brief description of a new decoding algorithm that does not require the dual variables to be at the optimum, i.e., decoding can be done at any point in the course of the dual algorithm.

The dual objective involves terms that are maximized locally with respect to the parent set variables. A decoding based on such a dual objective means a consistent setting of all the parent set variables (a setting that represents a DAG). If this can be done without changing the dual value, then the dual is tight and the configuration $s_1, \ldots, s_n$ is the MAP solution. This is known as a certificate of optimality.

The decoding algorithm is defined in two stages. In the first stage, we induce an ordering over the variables based on the dual solution and then, in the second, simply maximize over the parent sets consistent with the ordering using the original scores $W_i(s_i)$. The ordering is obtained iteratively based on the dual scores

$$W_i(s_i; \lambda) = W_i(s_i) + \sum_{C: i \in C} \lambda_C I_C(s_i) \quad (14)$$

as follows. Initialize $P_1 = \emptyset$. For $t = 1, \ldots, n$,

$$R_i = \max_{s_i \in P_a(i)} W_i(s_i; \lambda) - \max_{s_i \in P_a(i),\, s_i \subseteq P_t} W_i(s_i; \lambda) \quad (15)$$

$$i_t = \arg\min_{i \in V \setminus P_t} R_i, \quad P_{t+1} = P_t \cup \{i_t\} \quad (16)$$

The resulting node ordering $i_1, \ldots, i_n$ will lead to decoded values $\hat{s}_1, \ldots, \hat{s}_n$ in the second stage. The choice of ordering is critical. By choosing $i \in V \setminus P_t$ in iteration t we exclude all $j \in V \setminus P_t$, $j \neq i$ from the set of possible parents of i as they would come later in the ordering. The iterative criterion aims to minimize the regret due to such exclusions. The lemma below further motivates the criterion.

Lemma 1. *Let $i_1, \ldots, i_n$ be any ordering and $P_t = \{i_1, \ldots, i_{t-1}\}$ be the set of nodes preceding i_t in the ordering. The regret R_{i_t} is computed as in the algorithm based on P_t. Then*

$$\textit{current dual value} \geq \textit{decoded value} + \sum_{t=1}^{n} R_{i_t} \quad (17)$$

Proof. The proof is a sequence of inequalities beginning with the dual value.

$$\begin{aligned}
&\sum_{t=1}^{n} \max_{s_{i_t} \in P_a(i_t)} W_{i_t}(s_{i_t}; \lambda) - \sum_C \lambda_C \\
&= \sum_{t=1}^{n} \left[\max_{s_{i_t} \in P_a(i_t),\, s_{i_t} \subseteq P_t} W_{i_t}(s_{i_t}; \lambda) + R_{i_t} \right] - \sum_C \lambda_C \\
&\geq \sum_{t=1}^{n} [W_{i_t}(\hat{s}_{i_t}; \lambda) + R_{i_t}] - \sum_C \lambda_C \\
&= \sum_{t=1}^{n} \left[W_{i_t}(\hat{s}_{i_t}) + \sum_{C: i_t \in C} \lambda_C I_C(\hat{s}_{i_t}) + R_{i_t} \right] - \sum_C \lambda_C \\
&= \sum_{t=1}^{n} W_{i_t}(\hat{s}_{i_t}) + \sum_C \lambda_C [\sum_{i \in C} I_C(\hat{s}_i) - 1] + \sum_{t=1}^{n} R_{i_t} \\
&\geq \sum_{t=1}^{n} W_{i_t}(\hat{s}_{i_t}) + \sum_{t=1}^{n} R_{i_t}
\end{aligned}$$

where $\hat{s}_1, \ldots, \hat{s}_n$ is the decoded solution. Because the decoded solution is an acyclic graph, $\sum_{i \in C} I_C(\hat{s}_i) \geq 1$ for all clusters C. This is the last inequality. □

5.3 Choosing clusters to add

In order to rank alternative clusters C to add to $\mathcal{C}$, we would ideally use the decrease in the dual afforded by incorporating each cluster. This would, however, mean that we would have to re-estimate all λ in response to each possible new cluster to include. A simpler alternative is to evaluate how much the objective would decrease if we optimized λ_C while keeping the other dual variables fixed. This decrease is, in fact, exactly $\max\{\delta_{i_1}, 0\}$.

We do not have a full separation algorithm for cluster constraints (c1). However, we heuristically search for the violated constraints in two distinct ways. First, we will find the most violated cycle inequality and introduce a cluster constraint (that subsumes the cycle inequality) for the corresponding set of nodes. The most violated cycle in the dual can be found easily using a modified all-pairs shortest path algorithm. To this end, we evaluate for each directed edge

$$\delta_{ji} = \max_{s_i \in P_a(i):\, j \in s_i} [W_i(s_i) + \sum_{C: i \in C} \lambda_C I_C(s_i)] \quad (18)$$

$$- \max_{s_i \in P_a(i):\, j \notin s_i} [W_i(s_i) + \sum_{C: i \in C} \lambda_C I_C(s_i)] \quad (19)$$

$\delta_{ji} > 0$ indicates that the current dual solution supports including a directed edge (j, i); none of the edges for which $\delta_{ji} < 0$ would appear in the corresponding primal solution.

We look for the cycles that maximize the minimum value of δ_{ji} along the cycle. Let $\Delta_{k\to l}$ represent the minimum value of δ_{ji} along a path from k to l. We maximize $\Delta_{k\to l}$ while keeping track of the shortest path that attains the value. $p_{k\to l}$ point to the first node in the selected directed path from k to l. A simple modified all-pairs shortest path algorithm for evaluating these is given by:

(0) Initialize $\Delta_{j\to i} = \delta_{ji}$ and $p_{j\to i} = i$ for all possible directed edges we can select; $\Delta_{j\to i} = -\infty$ for the remaining edges.

(1) For $k = 1, \ldots, n$, $i = 1, \ldots, n$, $j = 1, \ldots, n$

if $\min\{\Delta_{i\to k}, \Delta_{k\to j}\} > \Delta_{i\to j}$
then $\Delta_{i\to j} = \min\{\Delta_{i\to k}, \Delta_{k\to j}\}$, $p_{i\to j} = p_{i\to k}$

As a result, the cycle that contains k and l and also maximizes the minimum value of δ_{ji} along the cycle has value $\min\{\Delta_{k\to l}, \Delta_{l\to k}\}$. The cycle can be retraced starting with the pointers $p_{k\to l}$ and $p_{l\to k}$. We only incorporate cycles for which $\min\{\Delta_{k\to l}, \Delta_{l\to k}\} > 0$, i.e., cycles that are immediately useful.

Some of the cluster constraints may be violated even if all of the cycle inequalities are satisfied, as the clusters are strictly stronger constraints. Our second strategy for finding violated clusters is based on greedily growing clusters, starting from each individual node. The criterion for adding a node k to a cluster C is simply the gain in the dual value resulting from enforcing the cluster constraint for $C \cup k$. Using the notation introduced above for the dual algorithm (see Eqs. 10 and 11), we add node k to cluster C if it maximizes

$$\min_{i \in C \cup k} (W^0_{\{C\cup k\},i} - W^1_{\{C\cup k\},i}) - \min_{i \in C} (W^0_{C,i} - W^1_{C,i}) \quad (20)$$

and the difference is non-negative. These values can be efficiently updated in the course of iteratively growing the cluster. If the resulting value of $\min_{i\in C}(W^0_{C,i} - W^1_{C,i})$ (the gain) is not strictly positive, the cluster is not included in the dual.

6 Branch and bound

When the LP relaxation is not tight, it is important to combine the relaxation with a branch and bound approach so as to obtain a certificate of optimality for the decoded structure. Since the dual linear program maintains an upper bound on the LP value at any point in the dual optimization, it is particularly well-suited for use as part of a branch and bound method. Our approach proceeds as follows. We first solve the LP in the dual. If the decoded value does not agree with the dual value (the LP is not tight), we divide the problem into two parts, initializing each part with the same clusters and the dual variables. The two dual LPs are then re-optimized separately. The branch and bound method maintains the best decoded value obtained in any branch and prunes away branches whose LP value falls below the decoded value. The branch with the highest LP value (most likely to contain the optimal solution) is always selected for division. The process continues until the best decoded value agrees with the highest LP value.

Our heuristic criterion for division is based on identifying a key node i and an associated cluster C. The parent sets of i are subsequently divided according to whether they lie outside the cluster ($I_C(s_i) = 1$) or overlap with the nodes in the cluster ($I_C(s_i) = 0$). The resulting $I_C(s_i) = 1$ branch will set $\lambda_C = 0$ after re-solving since all the parent sets of i will satisfy the cluster constraint. The $I_C(s_i) = 0$ branch will have to identify another variable $j \in C$ to satisfy the cluster constraint, therefore increasing the value of λ_C. Progress is guaranteed in either branch provided that node i was used to satisfy the constraint for C. In order to ensure that the dual value actually decreases due to branching, two or more clusters must prefer different selections of parent sets for node i. This competition between clusters is manifested in the coordinate

descent steps ($I_C(s_i)$ turns from zero to one after optimizing the cluster constraint). We identify the variable with the largest number of flips as the key variable.

7 Experiments

In this section, we provide empirical results demonstrating that our method can be used to find good solutions (either exact, or within a small optimality gap) to structure learning problems.

To compare our approach to exact dynamic programming (DP) methods, we modified and further optimized the DP approach described in Silander & Myllymäki [2006] so that it can take advantage of pruned parent set choices for each node. The reported times do not include the time required to generate the parent set choices, since this was common to both methods.

We used four reference problems: 1) expression data for a subset of 22 microRNAs out of 218, described in Lu *et al.* [2005], 2) binary phenotypic data for *C. elegans* involving 25 variables, 3) WDBC from the UCI repository with 31 variables, 4) 1000 samples from the Alarm network with 37 variables. The pruned parent set scores for these reference problems can be found on the supplementary website.[2] DP results were obtained for all but the Alarm network (however, the projected time is provided).

We show in Figure 2 a comparison of the running times to exactly solve these structure learning problems to optimality. Although the results on the first three problems are comparable, our method solves the Alarm network orders of magnitude faster. We used the BDe scoring metric for these comparisons.

Figure 3 illustrates how decoded solutions obtained in the course of our algorithm are useful even prior to convergence. In particular, the optimal structure is found much before the dual value agrees with the decoded value (relaxation is tight). Our method can be stopped at any point, with the guarantee that the score of the optimal structure does not deviate from the decoded structure by more than the difference between the dual and the decoded values.

We next compare our method to the branch and bound approach of de Campos *et al.* [2009], using the code provided by the authors.[3] We refer to their algorithm as SL and to ours as BBLP (for Branch and Bound with Linear Programming). Both algorithms were run with a constraint of in-degree at most 4 and the score used BIC as the complexity penalty.

[2] `http://groups.csail.mit.edu/tml/sl/`

[3] `http://www.ecse.rpi.edu/~cvrl/structlearning.html`.

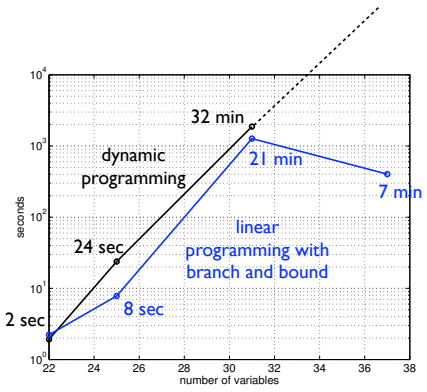

Figure 2: Time (in seconds) to solve reference problems with DP (black) and LP with branch and bound (blue). Each variable was restricted to have at most 4 parents for both methods. A projected time is shown for DP for the Alarm problem. Note that the y-axis is shown in log scale.

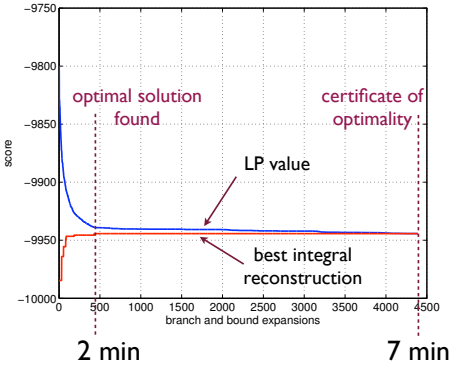

Figure 3: The evolution of the LP value (upper bound) and the best decoded value as a function of branch and bound iterations, for the Alarm problem.

We compare the algorithms on the datasets described above (RNA, Phenotype and Alarm) as well as seven other UCI datasets tested in de Campos *et al.* [2009] with the pre-processing described therein. Since both algorithms have efficient implementations and are anytime, we run both of them up to five minutes and report the gap between the lower and upper bounds obtained.[4]. As can be seen in Table 1, BBLP obtains a lower gap for all instances studied. Moreover, our termination times were substantially shorter (not shown).

Finally, we use the phenotype data with the BDe scoring metric to illustrate the importance of the (c1) con-

[4] The runtimes for this experiment were shorter than those described earlier in Figure 2 due to the different hardware used, and the fact that a BIC score was used as opposed to BDe.

	BBLP	SL
adult	0	5.1
car	0	0
letter	0	0
lung	0.002	2.5
mushroom	5	10
nursery	0	0
wdbc	0	6.8
zoo	0	0
RNA	0	0
phenotype	0	3.35
alarm	0	20

Table 1: Comparison of the structure learning algorithm of de Campos *et al.* [2009] (denoted by SL) and the LP method described here (denoted by BBLP), for eleven datasets (see description in the text). The numbers give the gap in percentage between the lower and upper bounds (normalized by the lower bound), after running both algorithms for up to five minutes.

straints (see Eq. 6). After enforcing all cycle inequalities (see Eq. 5), the LP relaxation gives an upper bound of -5993. Next, we added as many (c1) constraints as we could find using a heuristic separation algorithm (see Section 5.3). Together with the cycle inequalities, these obtained an upper bound of -6064. The score of the optimal structure is -6071. Note that the cycles + (c1) bound may actually be tighter, as the separation heuristic could have missed some violated (c1) constraints. More generally, across all of our experiments, we found that the (c1) constraints were indispensable for branch and bound to succeed at finding the optimal structures.

8 Discussion

We have presented a new exact method for finding the Bayesian network structure. In contrast to other exact methods, our approach derives from linear programming relaxations of the structure learning problem. We provide a fast dual algorithm for solving the associated LP, introduce new constraints on the polytope of valid acyclic graphs, a decoding algorithm for finding an integer solution, and a mechanism for guiding a branch-and-bound method on the basis of the dual relaxation, so as to obtain a certificate of optimality. In terms of the methodology, the linear programming formulation also has strong ties to MAP inference.

Our empirical results are promising, indicating competitive or better results than recent dynamic programming and branch-and-bound approaches to structure learning.

References

Chickering, D. 1996. Learning Bayesian Networks is NP-Complete. *Pages 121–130 of:* Fisher, D., & Lenz, H.J. (eds), *Learning from Data: Artificial Intelligence and Statistics V.* Springer-Verlag.

Chickering, D. 2002. Learning Equivalence Classes of Bayesian-Network Structures. *Journal of Machine Learning Research*, **2**, 445–498.

Chickering, D., Heckerman, D., & Meek, C. 2004. Large-Sample Learning of Bayesian Networks is NP-Hard. *Journal of Machine Learning Research*, **5**, 1287–1330.

Dasgupta, S. 1999. Learning polytrees. *In: Proc. of the 15th Conference on Uncertainty in Artificial Intelligence.*

de Campos, C., Zeng, Z., & Ji, Q. 2009. Structure Learning of Bayesian Networks using Constraints. *In: Proc. of the 26th International Conference on Machine Learning.*

Friedman, N., & Koller, D. 2003. Being Bayesian about Bayesian Network Structure: A Bayesian Approach to Structure Discovery in Bayesian Networks. *Machine Learning*, **50**(1–2), 95–125.

Grötschel, M., Jünger, M., & Reinelt, G. 1985. On the acyclic subgraph polytope. *Math. Prog.*, **33**, 28–42.

Guo, Y., & Schuurmans, D. 2006. Convex Structure Learning for Bayesian Networks: Polynomial Feature Selection and Approximate Ordering. *Pages 208–216 of: Proc. of the 22nd Conference on Uncertainty in Artificial Intelligence.* Arlington, Virginia: AUAI Press.

Heckerman, D., Geiger, D., & Chickering, D. 1995. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, **20**, 197–243.

Koivisto, Mikko, & Sood, Kismat. 2004. Exact Bayesian Structure Discovery in Bayesian Networks. *Journal of Machine Learning Research*, **5**, 549–573.

Lu, J., Getz1, G., Miska, E., Alvarez-Saavedra, E., Lamb, J., Peck, D., Sweet-Cordero, A., Ebert, B., Mak, R., Ferrando, A., Downing, J., Jacks, T., Horvitz, H., & Golub, T. 2005. MicroRNA expression profiles classify human cancers. *Nature*, **435**(June), 834–837.

Magnanti, Thomas L., & Wolsey, Laurence A. 1995. *Handbooks in Operations Research and Management Science.* Vol. Volume 7. Elsevier. Chap. 9, Optimal trees, pages 503–615.

Parviainen, P., & Koivisto, M. 2009. Exact Structure Discovery in Bayesian Networks with Less Space. *In: Proc. of the 25th Conference on Uncertainty in Artificial Intelligence.*

Silander, T., & Myllymäki, P. 2006. A simple approach for finding the globally optimal Bayesian network structure. *In: Proc. of the 22nd Conference on Uncertainty in Artificial Intelligence.*

Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T., & Weiss, Y. 2008. Tightening LP Relaxations for MAP using Message Passing. *In: Proc. of the 24rd Conference on Uncertainty in Artificial Intelligence.*

Spirtes, P., Glymour, C., & Scheines, R. 2001. *Causation, Prediction, and Search, 2nd Edition.* The MIT Press.

Teyssier, M., & Koller, D. 2005. Ordering-based Search: A Simple and Effective Algorithm for Learning Bayesian Networks. *Pages 584–590 of: Proc. of the 21st Conference on Uncertainty in Artificial Intelligence.*