**PREDICTING THE HARDNESS OF LEARNING BAYESIAN NETWORKS**

Brandon Malone, Kustaa Kangas, Matti Järvisalo, Mikko Koivisto, Petri Myllymäki

**INTRODUCTION**

**BAYESIAN NETWORKS**

A Bayesian network is a graphical model on random variables $X_1, \ldots, X_n$. The **structure** of a Bayesian network is a directed acyclic graph (DAG) $G$.

**SCORING FUNCTIONS**

A scoring function $s$ measures how well $G$ fits observed data on the variables. Typical scoring functions decompose into a sum $s(G) = \sum_{i=1}^n s_i(G_i)$, where $G_i$ is the set of parents of $X_i$ in $G$.

Common $s$: penalized likelihood, minimum description length, BDeu, etc.

**STRUCTURE LEARNING PROBLEM**

**Input:** A set $G_i$ of candidate parent sets for each variable $X_i$ and the local scores $s_i(G_i)$ for all $G_i \in G_i$.

**Task:** Find a DAG $G$ such that $G_i \in G_i$ and the score $s(G)$ is maximized. (NP-hard)

$$G = \{G_1, G_2, \ldots, G_n\}$$

**ALGORITHMS**

Various exact algorithms are guaranteed to find an optimal $G$ while avoiding exhaustive search in the space of all DAGs:

- **Dynamic programming** over variable subsets finds an optimal ordering of variables that is compatible with an optimal DAG.
- **Integer linear programming** searches a convex polytope where each vertex is a feasible solution. Cutting planes are added during search to enforce acyclicity.
- **Branch and bound** searches a relaxed space of cyclic graphs and breaks cycles by branching on arcs to remove in best-first order.

**PREDICTORS**

We use REP trees to train two predictors:

- **Predictor A:** Uses the features $n$ and $m$. Blue instances were solved faster by ILP, red ones by A*.
- **Predictor B:** Uses all features.

**PREDICTION**

Although the simple predictor A already admits an efficient portfolio algorithm, predictor B makes more accurate predictions:

Orthogonality between dominant solvers w.r.t. $n$ and $m$. Blue instances were solved faster by ILP, red ones by A*.