Learning to Search in Branch-and-Bound Algorithms

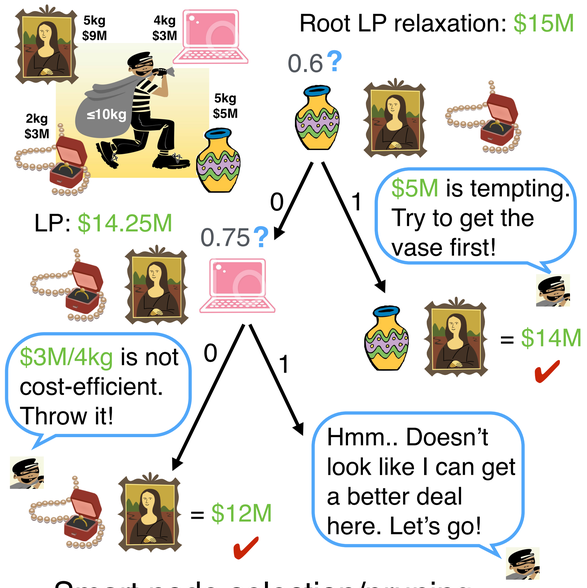
He He, Hal Daumé III — University of Maryland, College Park
Jason Eisner — Johns Hopkins University

Overview

What is the best strategy to traverse the search tree in branch and bound?

- *Best* means to find a near-optimal solution as early as possible
- Different types of problems require different search strategies
- A single strategy usually does not work well throughout the search tree
- Our solution: automatically learns searching strategies that are *adapted* to a family of problems and different solving stages within one problem

Toy example:
knapsack problem formulated as integer linear programming (ILP)

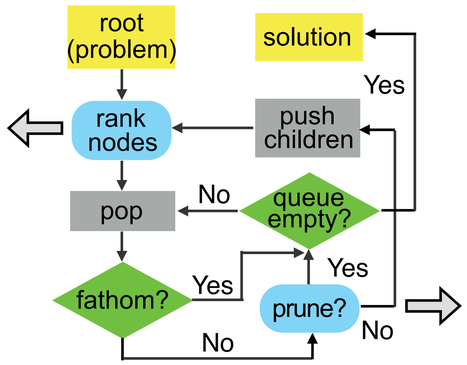


- Smart node selection/pruning speed up the solving process
- Good decisions come from experience — imitation learning

Method

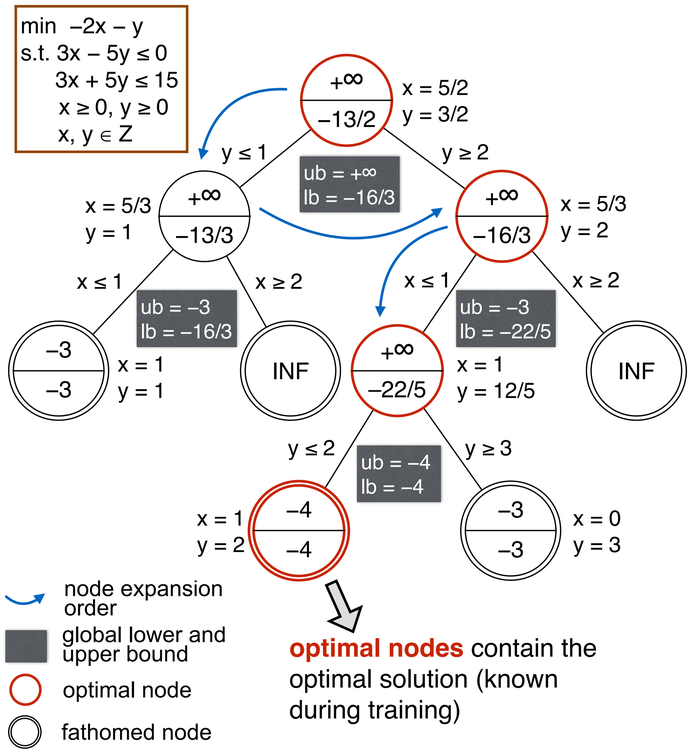
Assumptions:
- A small set of *solved* problems are given at training time
- Problems to be solved at test time are of the same type
- Finding a good feasible solution is enough — no need for proof of optimality

Node selection policy:
- A node comparator
- Node score = weight vector × feature vector
- Higher score means the subtree is more likely to contain the optimal solution
- Always pop the node with the highest score



Node pruning policy:
- A node classifier
- Decide whether to expand the node or cut it off

Both policies are learned by imitation learning (Dataset Aggregation)



optimal nodes contain the optimal solution (known during training)

Oracle:
- Expand optimal nodes first
- Prune all non-optimal nodes
- Provide training labels

Training examples:

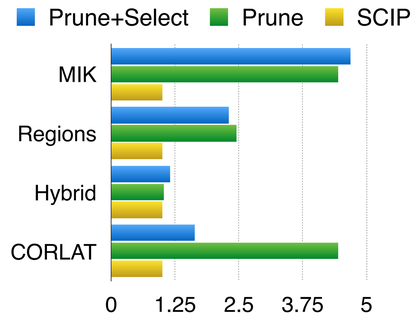
$$w_{\text{select}} \cdot \left(\varphi\left(\frac{+\infty}{-13/3}\right) - \varphi\left(\frac{+\infty}{-16/3}\right) \right) = 0$$

$$w_{\text{prune}} \cdot \psi\left(\frac{+\infty}{-13/3}\right) = 1$$

Policy features (dynamic):
- Node: lower bound, estimated objective, depth, is child/sibling
- Branching: pseudocost, difference between current LP solution and root LP solution/ current bound
- Tree: global bounds, integrality gap, number of solution found

Experiments

- Four Mixed-ILP libraries (# of vars: 300 - 1000; # of constrs: 100 - 500)
- Solver implemented based on SCIP
- Speedup with respect to SCIP



- Optimality gap compared with SCIP (early stop at the same end time) and Gurobi (early stop at the same # of nodes explored)

	P+S	P	SCIP	Gurobi
MIK	**0.04‰**	**0.04‰**	3.02‰	0.45‰
Regions	**7.21‰**	7.68‰	**6.80‰**	21.94‰
Hybrid	**0.00‰**	**0.00‰**	0.79‰	3.97‰
CORLAT	**8.99%**	**8.91%**	fail	fail

- SCIP and Gurobi in their default settings work well on some datasets but not all; while our policy learns to adapt to specific problems
- Cross generalization — apply policies learned on one dataset to another