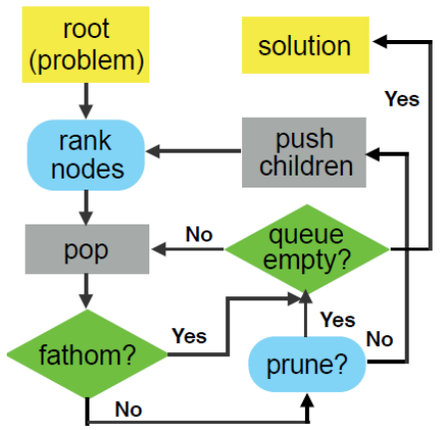
Learning to Search in Branch-and-Bound He HE et. al.

The problem they talcked:

A crucial question in B&B is how to specify the order in which nodes are considered. An effective node ordering strategy guides the search to promising areas in the tree and improves the chance of quickly finding a good incumbent solution

Proposed Algorithms

Method flowchart



Algorithm 1 Policy Learning (π_S^*, π_P^*)

$\pi_P^{(1)} = \pi_P^*, \pi_S^{(1)} = \pi_S^*, \mathcal{D}_S = \{\}, \mathcal{D}_P = \{\}$
for $k = 1$ **to** N **do**
  for Q **in** problem set $\mathcal{Q}$ **do**
    $\mathcal{D}_S^{(Q)}, \mathcal{D}_P^{(Q)} \leftarrow \text{CollectExample}(Q, \pi_P^{(k)}, \pi_S^{(k)})$
    $\mathcal{D}_S \leftarrow \mathcal{D}_S \cup \mathcal{D}_S^{(Q)}, \quad \mathcal{D}_P \leftarrow \mathcal{D}_P \cup \mathcal{D}_P^{(Q)}$
  $\pi_S^{(k+1)}, \pi_P^{(k+1)} \leftarrow$ train classifiers using $\mathcal{D}_S$ and $\mathcal{D}_P$
return Best $\pi_S^{(k)}, \pi_P^{(k)}$ on dev set

Used method : Imitation Learning

- Oracle: Imitation learning requires an oracle at training time to demonstrate the desired behavior
- Node selection policy: repeatedly picks a node from the queue of all unexplored nodes
- Node pruning policy: decides if the popped node is worth expanding

Advantages:

- Non-problem-dependent learning: can be applied to any family of problems solvable by the B&B framework. We use imitation learning to automatically learn the heuristics, free of the trial-and-error tuning and rule design by domain experts in most B&B algorithms
- Dynamic decision-making
  - learns different strategies for different problem types
  - within a problem type, it can evaluate the hardness of a problem instance based on features describing the solving progress
  - adapts the searching strategy to different levels of the B&B tree and makes decisions based on node-specific features
- Easy incorporation of heuristics: multiple heuristics can be simply plugged in as state features for the policy, allowing a hybrid "heuristic" to be learned effectively.

Drawbacks:

- Assume that a small set of solved problems are given at training time and the problems to be solved at test time are of the same type.
- Need a 'ORACLE' that knows beforehand the optimal solution of orignal IP problem

Code for the paper: https://github.com/hhexiy/scip-dagger