# Least Discrepancy Search*

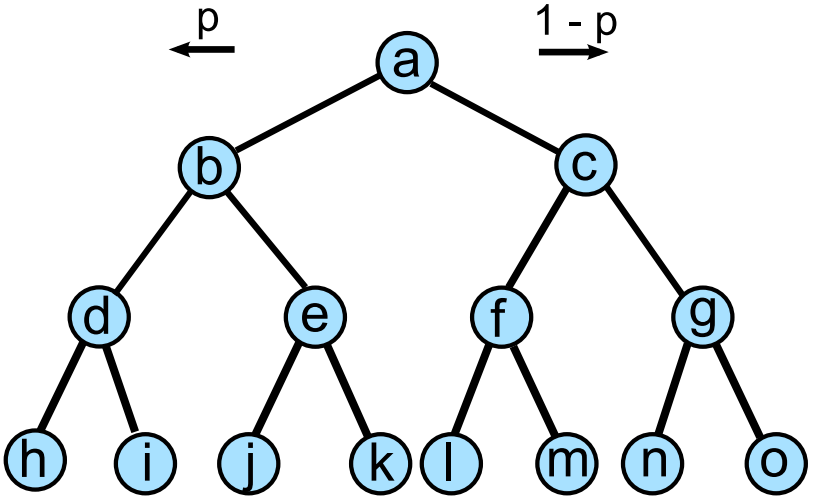
CS 149 Staff

November 17, 2009

When performing a branch and bound search, imagine we had a heuristic that could tell us the probability $p > 0.5$ of finding the optimal node by searching the left branch as opposed to the right branch. Given such a heuristic, it would make sense to look at the node in the search tree farthest to the bottom left, which in a search tree with $d$ decision variables has probability $p^d$ of being the optimal node. After looking at this node, we would naturally then want to look at the nodes with the next highest probability, $(1-p)p^{d-1}$, and so on until we finally reach the node with the lowest probability, $(1-p)^d$.

Once we have the nodes ranked by probability, we then have to break ties between nodes with the same probability. For example, in the following search tree nodes $i, j$ and $l$ all have probability $(1-p)p^2$. We break ties based on the intuition that if our heuristic has made a mistake (that is, it told us to branch left when we should have branched right), we probably made the mistake higher up in the search tree when we had less knowledge about the problem. This gives us the ordering that we will first look at node $l$, then node $j$, and finally node $i$ to break the tie at probability $(1-p)p^2$.

Consider the search tree for a problem with three decision variables ($d = 3$), $x_1$, $x_2$, $x_3 \in \{0, 1\}$, where we first branch on $x_1$ at node $a$ and $x_2$ at nodes $b$ and $c$, and $x_3$ at $d, e, f, g$. Note that the leaves in the tree all correspond to integer solutions, since all of the decision variables take on integer assignments at those nodes.




*a.k.a. Limited Discrepancy Search

The probability of each leaf node having the optimal value is given below:

**Sorted by node**

- **h** $p^3$
- **i** $(1-p)p^2$
- **j** $(1-p)p^2$
- **k** $(1-p)^2p$
- **l** $(1-p)p^2$
- **m** $(1-p)^2p$
- **n** $(1-p)^2p$
- **o** $(1-p)^3$

**Search order**

- **h** $p^3$
- **l** $(1-p)p^2$
- **j** $(1-p)p^2$
- **i** $(1-p)p^2$
- **n** $(1-p)^2p$
- **m** $(1-p)^2p$
- **k** $(1-p)^2p$
- **o** $(1-p)^3$

This idea scales to a search tree of any size, as long as the decision variables are binary.