Check Website for HW1 Examples (added copy method to mazes)

Choosing Heuristics

Cost of search is exponential in search effective branching factor, b. So we want to be able to estimate.

Effective Branching Factor

b* given by

$$N + 1 = 1 + b^* + (b^*)^2 + \ldots + (b^*)^d = \frac{(b^*)^{d-1} - 1}{b^* - 1}$$

N = number of nodes we expanded

Example: if you find a solution of depth 5, and 52 nodes
After solving for b* in the above equation, b* = **1.92**

The book considers 2 heuristics for the 8-puzzle.

h1 = the number of misplaced tiles (underestimate)
h2 = the manhattan distance

Did 1200 random configurations of the 8-puzzle.
They got b* = 1.5 for h1, and b* = 1.3 for h2. Thus, h2 is a better heuristic to use.

We can show that h1 <= h2 for all possible boards. So as both are admissible, h2 will theoretically & empirically give better results.

Notice if weaken rules of 8-puzzle, then both h1 and h2 are the exact cost of a solution to the appropriately weakened problem.

Say h1 and h2 are solutions to a relaxed version of the problem.
Exact Relaxed solutions for a given problem yield admissable heuristics for original problem.

Suppose you have a list of admissible heuristics h1,…hm. Each performs better than all others in some circumstance.
Then h(n) = max{ h1(n), …, hm(n) }
Is also an adimissible heuristic and performs at the best level among these heuristics.

Admissible heuristics can also be generated by solutions to subproblems.

Using this idea can come with disjoint pattern databases, and use them to get a heuristic cost of a solution.

Learning heuristics

How can an agent come up with heuristics.

Maybe try to learn heuristics.

Give a sequence of optimal solutions to a given problem and try to learn h(n).

Could use decision trees, neural nets, etc… to try to learn h(n).

Will look at inductive learning approach.

To do this, break problem into a set of relevant features… TO BE CONTINUED