ECS 170

Xie Zhou

912143385

01/22/2017

HW2

Problem 1: IDS is not efficient when the target node is on the longest path. It's even worse when all other paths we need to go through are deep.

Suppose the left most path is the deepest path in the whole tree and its leaf node is the target node. Using DFS would find the optimal path in the first search but using IDS would go through all the other paths in the tree.

Problem 2: Given $h(x) = h'(x) + C$ where $h'(x)$ is the exact estimation or the underestimation of x. (admissable)

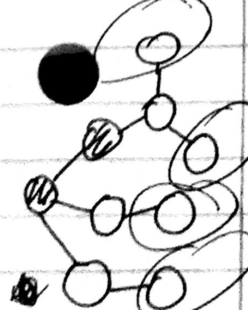
$$f(n) = g(n) + h(n)$$
$$= g(n) + h'(n) + C$$

Because $g(n) + h'(n) \leq f^*$ since $h'(n)$ is admissable

$$f(n) - C \leq f^*$$
$$f(n) - f^* \leq C$$

Problem 3: a) This method will overestimate. For example, if # uncovered nodes is very large, they could still be covered after 1 step is taken. This case would be when many nodes are connected to one central node.

b) Say k is large and we only have covered few nodes. The optimal solution could be found in the next few steps. $k - |S|$ could overestimate the actual # remaining nodes in the optimal path. It's possible that many nodes in S never needs to be added to the set cover.



c) Number of nodes that are not covered, and not directly connected to the current tree. It underestimates the # remaining nodes by excluding the "intermediate" node.

Problem 4:

a) By definition given:

$h_{max}(n) = \max_i h_i(n)$, $h_1, h_2 \ldots h_k$ are admissable

because $h_i(n)$, $1 \le i \le k$, is admissable,

h_{max} is an admissable heuristic

b) Because $h_{max}(n) \geq h_i(n)$ $1 \le i \le k$

and $g_1(n) = g_2(n) = \cdots = g_k(n)$ by assumption

and $f_1(n) = g_1(n) + h_1(n)$, $f_2(n) = g_2(n) + h_2(n)$,

$\ldots\ldots$ $f_i(n) = g_i(n) + h_i(n) \ldots$

and $f_{max}(n) = g_{max}(n) + h_{max}(n)$

$\therefore$ $f_{max}(n) \geq f_i(n)$

$f^* \geq f_{max} \geq f_i$

Therefore, h_{max} dominates all other h_i.

c) So we know that method 1 opens no more nodes than method 2. Method 1 does less work than method 2, and is more informed.

Problem 5:

a) It overestimates because swapping changes a pair of tiles. So 1 swapping of 2 adjacent ~~tien~~ tiles could place them into the correct spots, but the heuristic would give an overestimation of cost 2.

b) (# wrong row + # wrong column)/2

Suppose we can swap between any 2 tiles, increase. # wrong row / wrong column by 1 if a number is not in the correct row/column.

Problem 6:

a) MST heuristic relaxes the condition that the salesman can travel back from an edge. MST only cares about minimizing the edge costs as long as the nodes are all "visited". Minimizing the remaining tree makes the heuristic never overestimate.

b) Let MST heuristic be h_1 and straight line distance be h_2.

h_1 = edge cost to the next chosen node + MST cost starting from that chosen node.

h_2 = edge cost to the next chosen node.

Therefore $h_1 \geq h_2$

since $f_i(n) = g(n) + h_i(n)$

$f^* \geq f_1(n) \geq f_2(n)$

$\therefore$ MST heuristic dominates straight line distance.

Problem 7

a) Yes. The cost function $f = g + h$ depends on Dijkstra's and A* algorithm. In this case the ε_{ij} doesn't change the cost function. It can be substracted from each edge weight w_{ij} for calculation in fact. We will still get the same P.

b) No. ε_{ij} is negative and Dijkstra's algorithm cannot work with negative edges to find the optimal path.
$(w_{ij} + \varepsilon_{ij} < 0)$

c) No. If $\varepsilon_{ij} = w_{ij}$, we are doubling the edge weights. In this case the heuristic function h needs to be changed and P is not guranteed to be the same.