Homework Set 3

Michael Anderson

October 21, 2010

CS531

Prof. Tadepalli

4.2 One way to formulate this problem is to begin with no tracks placed as the initial state. A state consists of the tracks so far placed, where they and connected to one another, and the amount of rotation in each connection. To transition from one state to the next, a track can be added to any tracks already in the current state. The branching factor is based on the number of tracks that can be added, and where each one can be attached, and the amount of rotation (up to 10 degrees) off center that the piece is placed with. The goal is reached when all of the pieces have been placed, and there are no overlapping pieces, or unconnected piece edges.

This problem could be solved with simulated annealing by using a utility function f:

$$f(state) = p - u - o$$

where p is the number of tracks so far placed, u is the number of unconnected edges, and o is the number of overlapping pieces.

4.5 The information that would need to be stored is already given, all of the states so far visited at any given point in the search would need to be stored in a list.

If a state is found to already be visited, there are two cases that need to be handled, the already visited state led to a goal, or it failed. If it led to a goal, then label the subplan that comes out of that goal, and redirect the current state to that label to avoid redundancy. If the already visited state failed to reach a goal, then consider the current state failed as well and do not expand it.

4.7 Let $\{ s_1, s_2 \ldots s_{n-1}, s_n \}$, denote the set of n states contained within some belief state b. Then:

$$h(b) = min(h*(s_1), h*(s_2) \ldots h*(s_{n-1}), h*(s_n))$$

In other words, make $h(b)$ the minimum of the optimal costs of solving any of the states within b. The admissibility of this heuristic is fairly straight- forward intuitively, there can be no solution from b that costs less than the lowest optimum cost from any of the states in b to a goal state, so this $h(b)$ cannot be over-optimistic.
This heuristic performs poorly in the vacuum cleaner example, because as far as the agent knows it may start in a goal state, and at any point it might be in a goal state (the floor may be completely clean). So $h(b)$ always equals 0.

4.8 a) The proof is very simple. By definition, if an action sequence is a solution for a belief state b, then it is a solution for each of the states contained within b. By the definition of subset there are no states contained within any subset of b that are not contained within b itself. Therefore any action sequence that solves b must also solve all of the states within any subset of b, and therefore must solve any subset of b.

The same cannot be said of supersets of b, because they may contain states that are not solvable by an action sequence that solves b. All that can be said is that an action sequence that solves b solves *some* of the states in any superset of b, if $b \neq \emptyset$.

b) The generic GRAPH-SEARCH algorithm expands leaf nodes if they are not already in the explored set or the frontier set. GRAPH-SEARCH should also consider whether a leaf node (a belief set) is a subset of a node already in the explored set (by iteration through the explored set and comparison), and if it is then instead of expanding it simply return that belief set's corresponding solution (or failure).

c) When AND-OR-GRAPH-SEARCH is performed with belief states, before considering the actions that can be taken from a given belief state (which occurs in OR- SEARCH), check to see if the belief state is a subset of belief states that have already been explored. If it is, then end the search and simply return the plan corresponding to the already-explored superset.

4.12 a) In the initial state the agent knows that there are no walls up or to the right of it, because it perceives it can legally make those moves. It is ignorant of where walls are located in the rest of the environment. Each belief state in the initial state would correspond to some combination of walls either occurring or not occuring in the remainder of the environment in between squares in the grid.

There may be up to 2 full columns of 3 walls that partition the 3 columns of the grid, and there may be be up to 2 full rows of walls that partition the 3 rows of the grid. $2 \times 3 + 2 \times 3 = 6 + 6 = 12$ walls. Since the agent has to consider that each of these walls could either exist or not exist, there are $2^{12} = 4096$ possible configurations of the environment from the initial position.

b) Since it is impossible to move down or left no matter what, there are only 2 walls remaining that could exist or not exist, giving $2^2 = 4$ unique percepts from the initial state.

c) As shown in b), there are 4 possible percepts from the initial state. If the agent is completely walled in, then no action can be taken. The agent has 1 possible action if there is one wall, and 2 possible actions if there are no walls. So there are $0 + 1 + 1 + 2 = 4$ branches from the initial node.

If the agent now considers possible situations arising from its position at either (2,1) or (1,2), it sees that it may be able to go in any of 3 directions. Two neighboring walls are possible from either square, giving $2^2 \times 2 = 8$ possible wall configurations. From either square if it is walled in it can only go back to (0,0). If there is one new neighboring wall then it can go in 2 directions, else it can go in 3 directions. So these 4 leaves will expand to $4(1 + 2 \times 2 + 3) = 32$ new leaves at the 3rd level of the tree.