

## 1 Search and Heuristics

As an example: if the agent shown were initially stationary, it might first turn to the east using (*right*), then move one square east using *fast*, then two more squares east using *fast* again. The agent will of course have to *slow* to turn.

- The size of the state space is  $4MN(V_{max} + 1)$ . The state representation is (direction facing,  $x$ ,  $y$ , speed). Note that the speed can take any value in  $\{0, ..., V_{max}\}$ .

- The maximum branching factor is 3, and this happens when the agent is stationary. While stationary it can take the following 3 actions - *fast*, *left*, *right*.

3. Is the Manhattan distance from the agent's location to the exit's location admissible? Why or why not?

No, Manhattan distance is not an admissible heuristic. The agent can move at an average speed of greater than 1 (by first speeding up to  $V_{max}$  and then slowing down to 0 as it reaches the goal), and so can reach the goal in less time steps than there are squares between it and the goal. A specific example: the target is 6 squares away, and the agent's velocity is already 4. By taking only 4 *slow* actions, it reaches the goal with a velocity of 0.

4. State and justify a non-trivial admissible heuristic for this problem which is not the Manhattan distance to the exit.

There are many answers to this question. Here are a few, in order of weakest to strongest:

- (a) The number of turns required for the agent to face the goal.
- (b) Consider a relaxation of the problem where there are no walls, the agent can turn and change speed arbitrarily. In this relaxed problem, the agent would move with  $V_{max}$ , and then suddenly stop at the goal, thus taking  $d_{manhattan}/V_{max}$  time.
- (c) We can improve the above relaxation by accounting for the deceleration dynamics. In this case the agent will have to slow down to 0 when it is about to reach the goal. Note that this heuristic will always return a greater value than the previous one, but is still not an overestimate of the true cost to reach the goal. We can say that this heuristic *dominates* the previous one.

5. If we used an inadmissible heuristic in A\* tree search, could it change the completeness of the search?

No! If the heuristic function is bounded, then A\* tree search would visit all the nodes eventually, and would find a path to the goal state if there exists one.

6. If we used an inadmissible heuristic in A\* tree search, could it change the optimality of the search?

Yes! It can make the good optimal goal look as though it is very far off, and take you to a suboptimal goal.

7. Give a general advantage that an inadmissible heuristic might have over an admissible one.

The time to solve an A\* tree search problem is a function of two factors: the number of nodes expanded, and the time spent per node.

An inadmissible heuristic may be faster to compute, leading to a solution that is obtained faster due to less time spent per node. It can also be a closer estimate to the actual cost function (even though at times it will overestimate!), thus expanding less nodes.

We lose the guarantee of optimality by using an inadmissible heuristic. But sometimes we may be okay with finding a suboptimal solution to a search problem.

## 2 Expanded Nodes

Consider tree search (i.e. no closed set) on an arbitrary search problem with max branching factor  $b$ . Each search node  $n$  has a backward (cumulative) cost of  $g(n)$ , an admissible heuristic of  $h(n)$ , and a depth of  $d(n)$ . Let  $c$  be a minimum-cost goal node, and let  $s$  be a shallowest goal node.

For each of the following, you will give an expression that characterizes the set of nodes that are expanded before the search terminates. For instance, if we asked for the set of nodes with positive heuristic value, you could say  $h(n) \geq 0$ . Don't worry about ties (so you won't need to worry about  $>$  versus  $\geq$ ). If there are no nodes for which the expression is true, you must write "none."

1. Give an expression (i.e. an inequality in terms of the above quantities) for which nodes  $n$  will be expanded in a breadth-first tree search.

$d(n) \leq d(s)$ : BFS expands all nodes which are shallower than the shallowest goal. Recall that our search performs the *goal – test* after popping nodes from the fringe, so we typically expand some nodes at depth  $s$ , before we expand the optimal goal node.

2. Give an expression for which nodes  $n$  will be expanded in a uniform cost search.

$g(n) \leq g(c)$ : Uniform cost search expands all nodes that are closer than the closest goal node. Recall that our search performs the *goal – test* after popping nodes from the fringe (this ensures optimality!), so we might expand some nodes of cost  $g(c)$ , before we expand the optimal goal node.

3. Give an expression for which nodes  $n$  will be expanded in an A\* tree search with heuristic  $h(n)$ .

$g(n) + h(n) \leq g(c)$ : All nodes with a total cost of less than  $g(c)$ , get expanded before the goal node is expanded. This can be proved by induction on the cost  $g(n) + h(n)$ . Consider a node  $n_1$  which satisfies this property. Note that its parent  $n_0$ , will also satisfy this inequality, and by the induction hypothesis,  $n_0$  will be expanded before the goal is expanded, which means that it will put  $n_1$  on the fringe, which will get expanded before the goal node is expanded.

4. Let  $h_1$  and  $h_2$  be two admissible heuristics such that  $\forall n, h_1(n) \geq h_2(n)$ . Give an expression for the nodes which will be expanded in an A\* tree search using  $h_1$  but not when using  $h_2$ .

Let  $S$ , be the set of all the nodes. Using the above part, set of nodes expanded by  $h_1$  is  $N_1 = \{n : g(n) + h_1(n) \leq g(c)\}$ , and, set of nodes expanded by  $h_2$  is  $N_2 = \{n : g(n) + h_2(n) \leq g(c)\}$ . The set of nodes expanded using  $h_1$  but not using  $h_2$ , is  $N_1 \cap (S - N_2)$ . Since,  $h_1 \geq h_2$ ,  $N_1 \subseteq N_2$ , hence  $N_1 \cap (S - N_2) = \phi$ .

5. Give an expression for the nodes which will be expanded in an A\* tree search using  $h_2$  but not when using  $h_1$ .

As above, set of nodes expanded using  $h_2$  but not using  $h_1$ , is  $N_2 \cap (S - N_1) = \{n : g(n) + h_2(n) \leq g(c) \text{ and } g(c) \leq g(n) + h_1(n)\}$ .