ComS 472
Homework 2

Sean Gordon

Sep 25, 2020

- 4.1 -

1) Hill climb search
2) Breadth first search, adding one layer of nodes at a time
3) First Choice Hill climb search, as T=0 effectively removes temperature from the algorithm
4) Random walk, as temperature would cause every successor to have equal probability
5) Random walk, as the only chance for movement would be mutation

---

- 4.7 -

As the initial belief state contains all possible states, and we know the true cost for each state, a good heuristic for an individual belief states is the maximum of the costs of the states it contains. As you can never know which specific state in a belief state you are in, you must assume the worst case (the largest cost state). Because any solution for this largest cost state would also solve the other states in the belief state, this heuristic is admissible.

- 4.8 -

1. A solution for b would pass through the subsets of b, and thus would be a solution from those subsets. Because the solution doesn't necessarily pass through the superstates of b, it is not a solution for the supersets.

2. Graph search can be optimized for this problem by keeping a list of visited nodes. If the current node has already been visited, we can skip it and its children. This will avoid loops that needlessly increase the cost.

- 4.10 -

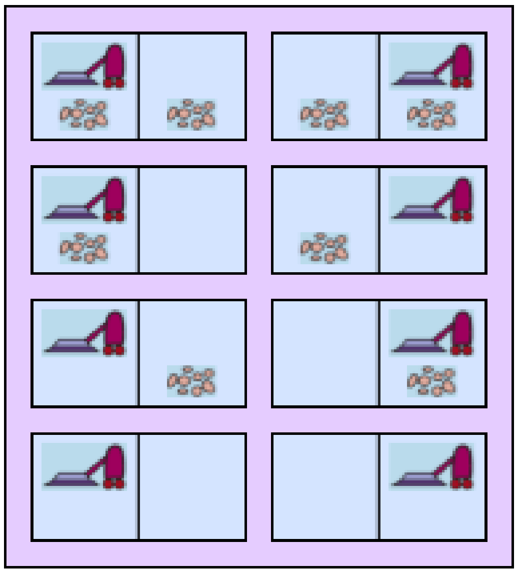
Figure 1: Belief states recheable from initial 8 belief states.

The problem is unsolvable because of the chance of depositing dirt when cleaning an empty square. Because there are no sensors, the agent can't tell if a square is initially clean or not and may end up depositing dirt when sucking, so the possible belief states can never be reduced.

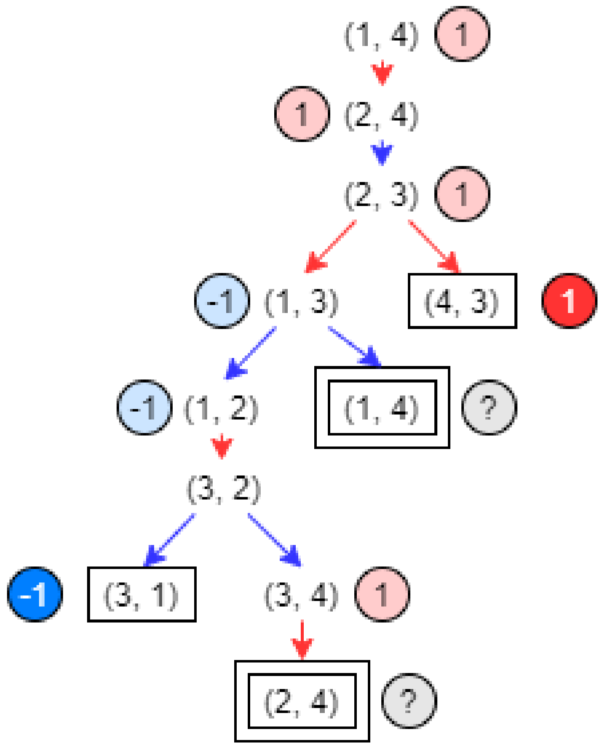

Figure 2: Line Game tree.

Because the ? values corresponded to loop positions that were always higher in the tree, they would be considered anyway as the algorithm propagates upward. Therefore, I just propagated the ? and ignored it when faced with a choice.

Minimax operates using DFS, and would therefore be stuck in an infinite loop upon reaching a loop position. This can be avoided by keeping a record of visited states, and skipping any that have been visited already.

If both players are playing optimally, the game is decided in the center of the box, with the win going to the first player to jump over the other, putting them 1 square closer to goal than the opposition. If n is even, A reaches the center line first and is poised to jump over B. If it is odd, A moves past the center line first, and B is poised to jump over A.

- 5.9 -

There are 255168 unique games possible

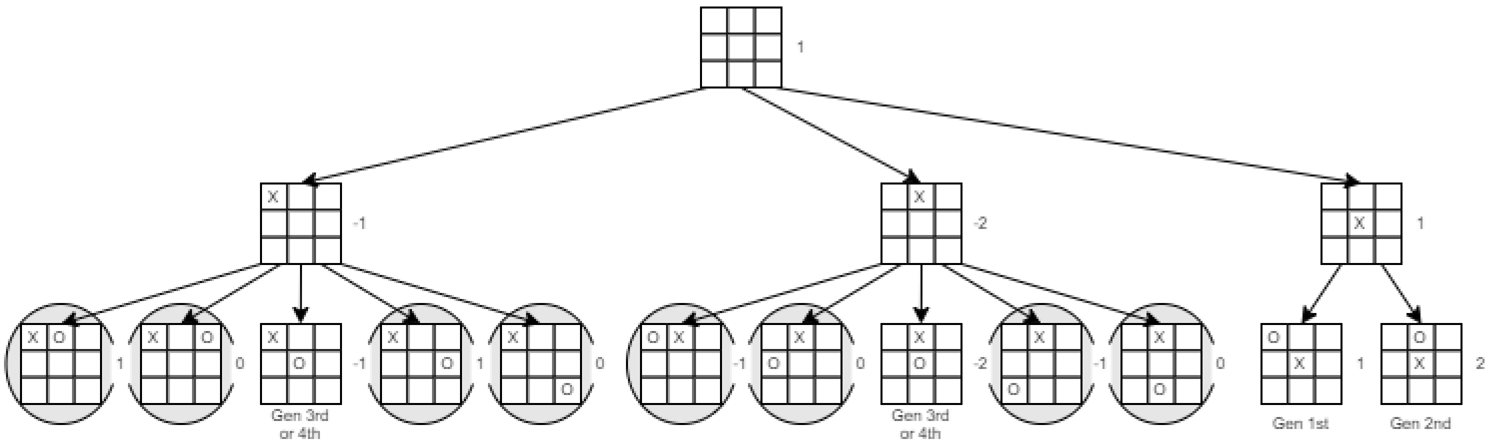

Figure 3: Tic-Tac-Toe tree.

- 5.14 -

1) $n_1 = min(max(n_3, n_{31}, ..., n_{3j_3}), n_{21}, ..., n_{2b_2})$
2) $n_1 = min(l_2, max(l_3, n_3, r_3), r_2)$
3) $n_1 = min(l_2, max(min(min(l_3, l_{31}, ..., l_{3j}), n_3, r_3), r_2)$
n_j must not exceed the upper bound placed on it by all nodes to its left $(l_2, l_4, ..., l_i)$, or it will not be selected higher up in the tree. I have no idea how to reformat this expression to show that without it being an absolute mess...
4) I still don't know how to reformat that expression, but the lower bound for this one is $max(l_3, l_5, ..., l_j)$.

- Extra -

The final value for root is 14.

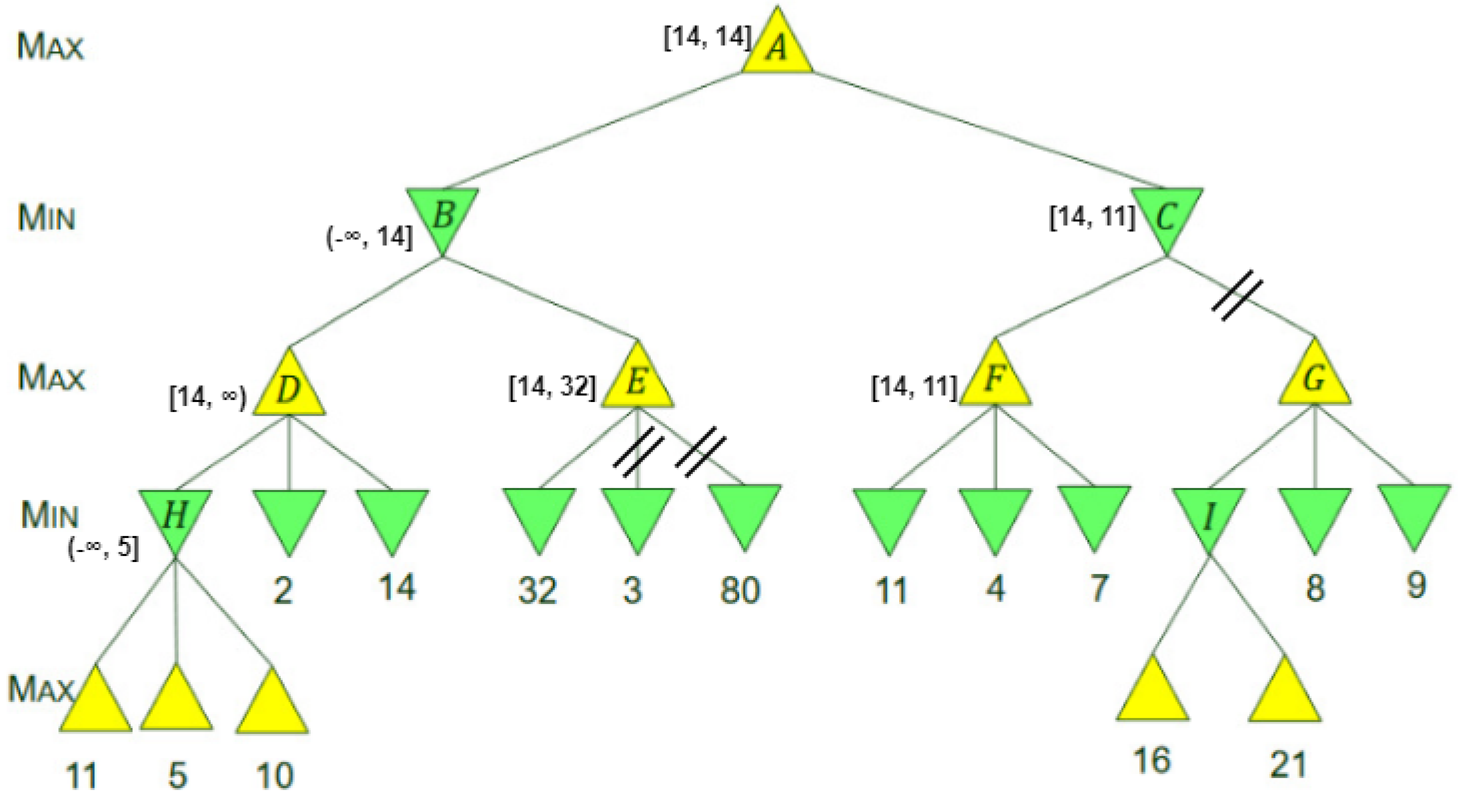

Figure 4: Alpha-Beta tree.