CS 188
Summer 2022

Final Review Games

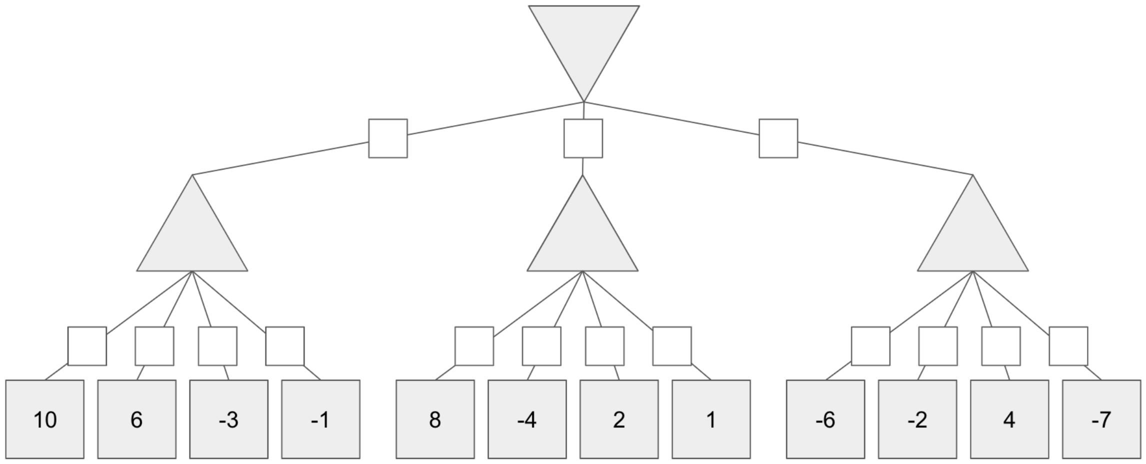
Q1. Coin Stars

In a new online game called Coin Stars, all players are walking around an M x N grid to collect **hidden coins, which only appear when you're on top of them**. There are also power pellets scattered across the board, which are visible to all players. If you walk onto a square with a power pellet, your power level goes up by 1, and the power pellet disappears. Players will also attack each other if one player enters a square occupied by another player. In an attack, the player with a higher power level will steal all the coins from the other player. If they have equal power levels, nothing happens. Each turn, players go in order to move in one of the following directions: {N, S, E, W}.

In this problem, you and your friend Amy are playing Coin Stars against each other. You are player 1, and your opponent Amy is player 2. Our state space representation includes the locations of the power pellets (x_{p_j}, y_{p_j}) and the following player information:

- Each player's location (x_i, y_i)
- Each player's power level l_i
- Each player's coin count c_i

(a) **(i)** Suppose a player wins by collecting more coins at the end of a number of rounds, so we can formulate this as a minimax problem with the value of the node being $c_1 - c_2$. Consider the following game tree where you are the maximizing player (maximizing the your net advantage, as seen above) and the opponent is the minimizer. Assuming both players act optimally, if a branch can be pruned, fill in its square completely, otherwise leave the square unmarked.



○ None of the above can be pruned

(ii) Suppose that instead of the player with more coins winning, every player receives payout equal to the number of coins they've collected. Can we still use a multi-layer minimax tree (like the one above) to find the optimal action?

○ Yes, because the update in payout policy does not affect the minimax structure of the game.

◯ Yes, but not for the reason above

◯ No, because we can no longer model the game under the updated payout policy with a game tree.

◯ No, but not for the reason above

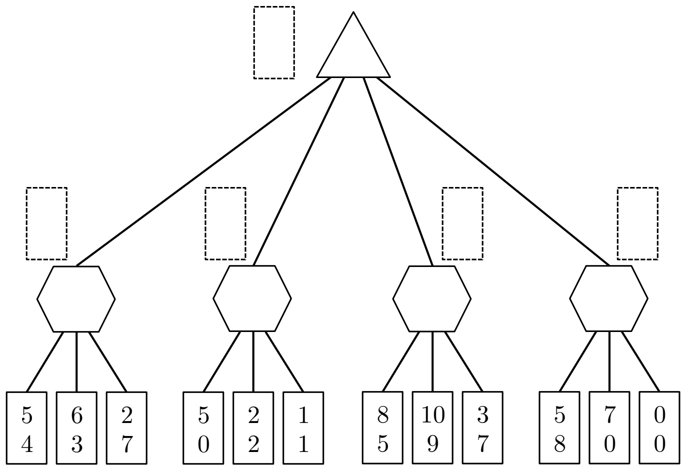
Q2. Game Trees and Pruning

You and one of the 188 robots are playing a game where you both have your own score.

- The maximum possible score for either player is 10.
- You are trying to maximize your score, and you do not care what score the robot gets.
- The robot is trying to minimize the absolute difference between the two scores. In the case of a tie, the robot prefers a lower score. For example, the robot prefers (5,3) to (6,3); it prefers (5,3) to (0,3); and it prefers (3,3) to (5,5).

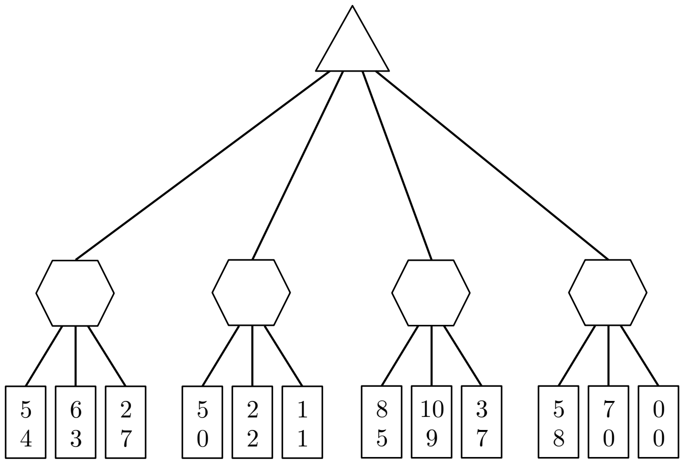
The figure below shows the game tree of your max node followed by the robots nodes for your four different actions. The scores are shown at the leaf nodes with your score always on top and the robots score on the bottom.

(a) Fill in the dashed rectangles with the pair of scores preferred by each node of the game tree.



(b) You can save computation time by using pruning in your game tree search. On the game tree above, put an 'X' on line of branches that do not need to be explored. Assume that branches are explored from left to right.

(c) You now have access to an oracle that tells you the order of branches to explore that maximizes pruning. On the copy of the game tree below, put an 'X' on line of branches that do not need to be explored given this new information from the oracle.


5
4
6
3
2
7
5
0
2
2
1
1
8
5
10
9
3
7
5
8
7
0
0
0