CS 188 Spring 2021 Introduction to Artificial Intelligence

Final Review Games

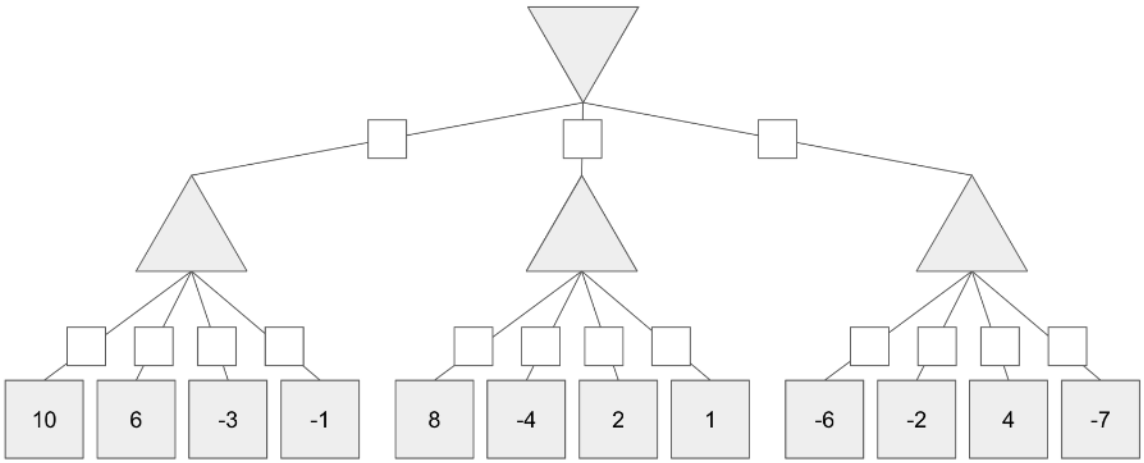
Q1. Coin Stars

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. Search & Games

(a) In this question, we will be formulating flying between two locations as a search problem. You've designed a cool new app that will find the fastest way to fly between two cities. For every flight in the world you know the departure time and location, and the arrival time and location. The app does not contain pricing information, it only tries to minimise time spent traveling. Between flights you assume people just wait in their current city. You may also assume that each city has only one airport. Formulate this as a search problem with a minimal state space:

What are the states for this search problem?

◯ Current city
◯ Current city and amount of time spent traveling so far
◯ Current city and current time
◯ Current city, current time, and amount of time spent traveling so far

What is the successor function for this search problem?

◯ Action: Take a flight
Successor: Update city
Cost: Time length of the flight plus time until takeoff

◯ Action: Take a flight
Successor: Update city and current time
Cost: Time length of the flight plus time until takeoff

◯ Action: Take a flight and increase the amount of time so far
Successor: Update city and time spent traveling
Cost: Time length of the flight plus time until takeoff

◯ Action: Take a flight and increase the amount of time so far
Successor: Update city, time spent traveling, and current time
Cost: Time length of the flight plus time until takeoff

What is the start state?

◯ Oakland
◯ Oakland, 0 minutes traveling
◯ Oakland, 8pm
◯ Oakland, 8pm, 0 minutes traveling

What is the goal test?

◯ Check if the current city is in Australia
◯ Check if the amount of time spent traveling is minimal
◯ Check if the current city is in Australia and if the amount of time spent traveling is minimal
◯ Check if the current city is in Australia, the amount of time spent traveling is minimal, and if the current time is minimal

(b) In this question, you are writing a program to compete in a chess tournament. Because it is your first time in the competition, you have been given access to your opponent's program and are allowed to do anything with it. (They are experienced and so do not get access to your program.) You adjust your minimax: instead of considering all actions at a min node, you run your opponent's code and consider only the action it would take. Time spent running your opponent's code does not count towards your own time limit. Select all of the statements below that are true. If none are true, select none of them.

◯ Your minimax search tree is equivalent to a game tree constructed with only max nodes
◯ Running your code for the same amount of time as in regular minimax, you are able to search deeper in the game tree than you could with regular minimax
◯ If you search to the same depth as you could with regular minimax you will always return the same answer as regular minimax

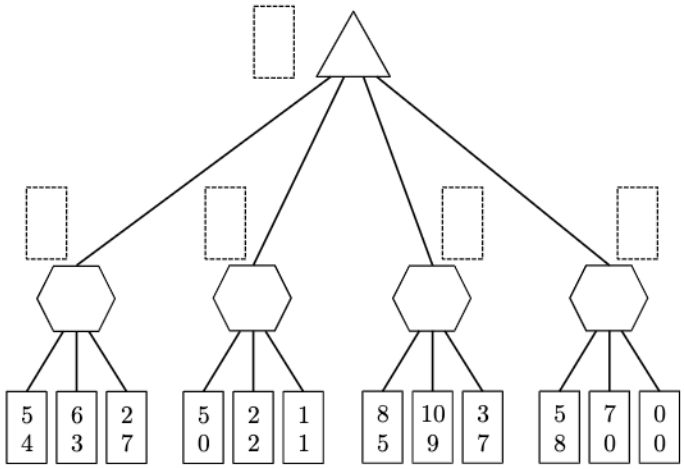
Q3. Game Trees and Pruning

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).

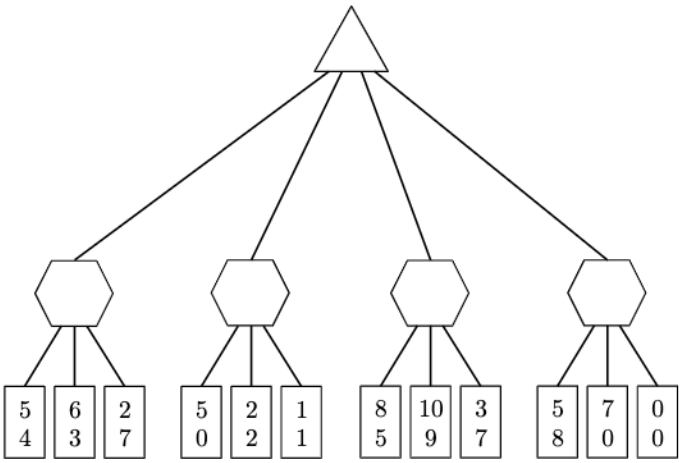
The figure below shows the game tree of your max node followed by the robots nodes for your four different actions. The scores are shown at the leaf nodes with your score always on top and the robots score on the bottom.

(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.


54
63
27
50
22
11
85
109
37
58
70
00