CS 6364-002 Homework 5

October 6, 2022

Requirements:

- Deadline for the first submission: **Oct-19-2022**.
- All assignments **MUST** have your name, student ID, course name/number at the beginning of your documents.
- The following youtube video shows how to implement Tic Tac Toe AI with the minimax algorithm. I posted this link here in case you find it useful.

  `https://www.youtube.com/watch?v=trKjYdBASyQ`
- Please write all the codes in one Jupyter notebook and run the codes to display the results in the notebook before you save the notebook (ipynb file). Pls zip the notebook and related files (if there are) into one file and submitted the zipped file.

If you have any questions, please contact me.

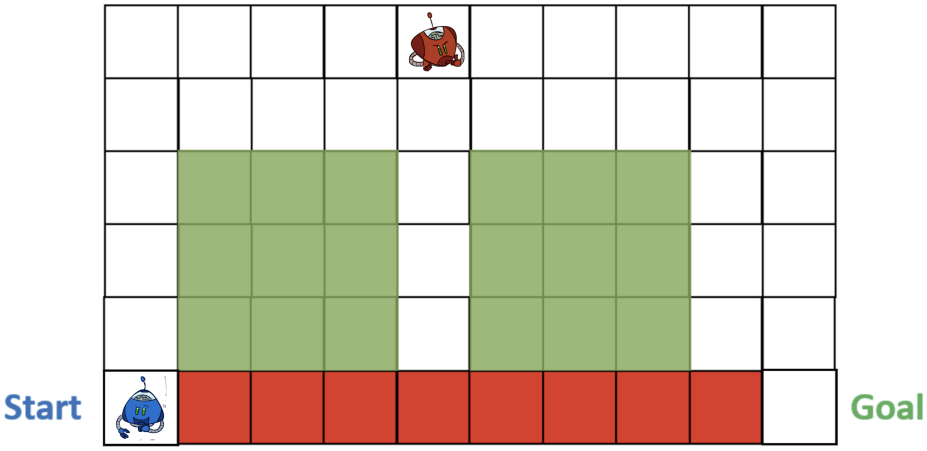
In this question, we aim to solve the "cliff walking" zero-sum game using the **minimax search method with alpha-beta pruning with the depth limited to six layers (d=6)** as introduced in Lecture 4-C. There are two players (the maximizing player and the minimizing player) who will control the movements of two agents respectively. There are one blue agent controlled by the maximizing player and red agent controlled by the minimizing player. The depth limit "d=6" means that each of the two agents can make three steps.

The size of the grid is 6×10 (see below). The blue agent starts at the leftmost cell in the bottom, that is, $(6, 1)$. The goal cell is the rightmost cell in the bottom (blue), that is, $(6, 10)$. All the cells with the green color refer to the regions with water. For the blue agent, the cost of each movement (action) in the white-colored cells (non-water region) is one, and the cost of each movement in the green-colored cells (water region) has the cost 5. All the cells between $(6, 2)$ and $(6, 9)$ is the cliff (red). If the blue agent enters the cliff, which means the agent falls into the cliff, then the agent will die. The red agent starts at the cell (1, 5). The red agent does not consider costs for its movements. Its task is to block the direction of the blue agent, such that the blue agent will select a path with the largest cost to the goal cell. The red agent will not enter the cliff and will be always alive.

Both blue and red agents can move only one cell at a time to the neighboring cell, that is, up, down, right and left, unless the agent touches the border or the other agent. When the agent touches the border or the other agent, the action that makes the agent cross the border is not performed but it must remain stopped at the point waiting until the next action. For example, if the agent is at (1, 3) and the action is to up, then agent remains at that point, and if the next action is to right, then it moves to (1, 4), or when the next action is down then agent moves to (2, 3). In this question, please finish the following two parts:

1. (a) Implement the **minimax search method (without alpha-beta pruning) with the depth limited to six layers (d=6)** and print out the returned paths of blue and red agents and the cost of the path for the blue agent. Note that, you will need to decide and implement an evaluation function to approximate the utilities of the "leaf nodes" of the search tree with depth equal to 6. The following information may (or may not) be useful for you to design your evaluation function: (1) the min-cost path returned by $A^\star$ search from the current cell to the goal cell and the min-length path returned by $A^\star$ search from the current cell of the red agent to the current cell of the blue agent.

2. (b) Extend your implementation in part (a) and add alpha-beta pruning. Print out the returned paths of blue and red agents and the cost of the path for the blue agent.


Start
Goal