CSIS, BITS Pilani K. K. Birla Goa Campus

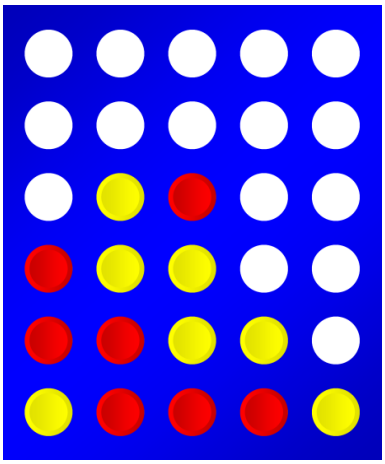
Artificial Intelligence (CS F407)

Programming Assignment 2
Total Marks: 24
Submission Deadline: 9 PM on ~~21/11/2021~~ 02/12/2021 (Thursday)

Each student must individually do this programming assignment. Your program must be written in Python and should run (without errors) on Python 3.6 or later.
Any form of plagiarism will result in -5 marks being awarded to everyone involved. There will be no differentiation between minor and major plagiarism.
Note that the deadline is **9 PM** and not midnight. Five marks per day will be deducted for submissions after the deadline. It will be your responsibility to submit the assignment well in advance and avoid unforeseen problems like power failures etc.

Question 1 (24 marks)

Consider a smaller version (5 columns $\times$ 6 rows) of Connect 4 game shown below. More details about the rules of Connect 4 game can be found here:
https://en.wikipedia.org/wiki/Connect_Four.



(a) Implement two versions of Monte Carlo Tree Search (MCTS) algorithm for playing the Connect 4 game shown above. Version MC_{40} uses 40 simulations (playouts) before picking an action, and version MC_{200} uses 200 simulations before picking an action. (Instead of total number of wins, you can use total rewards to estimate the value of a state in MCTS.) You can also change parameter C in the action selection policy determined by UCT. Whatever changes in parameters you make must be made for both the versions of the MCTS algorithm. Make the two versions of MCTS play against each other for 100 games with each version being the first player (Player 1) for 50 games. (The two versions should maintain separate Game trees. Before choosing the first action, the algorithms can construct a game tree till

depth four depending on the n paths that were simulated, where n is the number of simulations/playouts. Thereafter, the algorithms can expand the nodes in the game tree as and when needed.) What choice of parameters give a clear advantage for MC_{200} algorithm in terms of number of wins in 100 games? What choice of parameters reduces the advantage that MC_{200} algorithm has? Give reasons for the results that you observe. You must submit the program for part (a) for evaluation as explained below.

(b) Implement the Q-learning algorithm for playing the simplified Connect-4 game (see NOTE below) . Choose appropriate rewards for the various terminal states. Will the concept of *afterstates* be useful? Train the Q-learning algorithm by allowing it to play the game against MC_n algorithm, where n can vary between 0 and 25 ($n = 0$ means that MC_0 always picks an action (uniformly) randomly without any playouts) . Use a good choice of parameters for MC_n based on your observations in part (a). Let MC_n always be Player 1 and Q-learning always be Player 2. (This will reduce the number of values that Q-learning needs to estimate.) For what values of parameters does Q-learning converge to optimal values in a fast manner. Show relevant graphs to justify your answer.

NOTE: For parts (b) and part (c), you can simplify the Connect-4 game to have 5 columns and r rows. The value of r can be between 2 and 4 (both inclusive). (When r is less than 4, arranging 4 coins diagonally will not be possible.) You can choose the maximum value of r for which Q-learning converges. If you had tried other approaches (for larger values of r and n) before this modification, then you can explain that in the report.

(c) Train the Q-learning algorithm such that it will be able win games against MC_n algorithm, where n can vary between 0 and 25 ($n = 0$ means that MC_0 always picks an action (uniformly) randomly without any playouts) . The Q-learning algorithm should try to win using minimum number of moves. For the MC_n algorithm, you can choose a good set of parameter values based on your observations in part (a). Let MC_n always be the first player and the Q-learning algorithm always be the second player. Describe the procedure that you followed to ensure that Q-learning will work well against a range of MC_n algorithms. You must submit the algorithms in part (c) for evaluation as explained below. You can save the estimates of various states found by the Q-learning algorithm in a data file (use .dat extension).

NOTE: As mentioned above, you can simplify the Connect-4 game to have 5 columns and r rows. The value of r can be between 2 and 4 (both inclusive). You can find the maximum value of r for which Q-learning performs well. If you had tried other approaches (for larger values of n and r) before this modification, then you can explain that too in the report.

Instructions for submission

- You must submit a single program file with the name "ROLLXYZ_FIRSTNAME.py". Your program should ask the user whether to show the output for part (a) or part (c)

of this assignment. For part (a), MC_{200} should be the first player and MC_{40} should be the second player. The output should be for the unsimplified Connect-4 game (i.e. 5 columns $\times$ 6 rows). For part (c), MC_n should be the first player and Q-learning algorithm should be the second player. The output should be for the simplified Connect-4 game (i.e. 5 columns $\times$ r rows). The program that you submit must print the maximum value of n and r for which Q-learning could converge. In the submitted program, the Q-learning algorithm should make a **greedy move** (i.e. $\epsilon = 0$) at each step. The estimates found by the Q-learning algorithm can be stored as a separate "ROLLXYZ_FIRSTNAME.dat" file. You should submit the data file as well. For both part (a) and part (c), the submitted program should play **one** game between Player 1 and Player 2. For each move, the output of your program should be similar to that shown in "ROLLXYZ_FIRSTNAME.py" (run the program and see the output).

- Try to minimize the size of the data file by carefully choosing an appropriate representation for game states. In the report, describe the state representation that was used. Use the gzip library to compress your data file. In this case, you can submit the "ROLLXYZ_FIRSTNAME.dat.gz" file. Your submitted program should read the estimates from the "ROLLXYZ_FIRSTNAME.dat.gz" file. See the link below for examples on how to use the gzip library:
`https://docs.python.org/3/library/gzip.html`

- Your report must be named "ROLLXYZ_FIRSTNAME.pdf". The report must contains details of the choices you made for parts (a) to (c). Also, the answers to parts (a) to (c) must include appropriate graphs wherever required. In the report, you can also include other ideas that you explored. Use **1.5 linespacing** in your report.

- Please use only capital letters for the three file names. Eg. 2020H1030999G_ADARSH.py, 2020H1030999G_ADARSH.dat and 2020H1030999G_ADARSH.pdf.

- Submit **only** the three files mentioned above. **Don't** zip the files. The assignment submission will be through quanta.