CS814 AI for Autonomous Systems – Coursework 2

Guidelines

Aim of the assignment

To provide deeper understanding of Game Playing algorithms (Minimax, alpha-beta pruning and Monte Carl Tree search algorithms).

Learning outcomes

Program in Python, with the goal being to implement key AI algorithms and build AI systems; Define and understand the problem of Artificial Intelligence as it relates to autonomous systems; Apply search techniques to enable autonomous systems to choose actions that are appropriate to their goals; Apply key techniques to adversarial problems, such as Mini-Max, alpha-Beta pruning, and Monte Carl Tree search.

Team

Students can work on the coursework and submit it individually. However, group work is allowed, and a group can be formed of maximum 2 students. Please use the activity "Coursework 2 Group Selection" on MyPlace to select your group, even if you are going to work alone. Further information is available on MyPlace.

Submission

Only one member of the group should submit two files:

1. The first file consists of the answers on the first question. Please make sure that the file is well written and well structured. One should be able to clearly read the solution and identify the final answer to each and every part. The submitted file should be in pdf format.
2. The second file consists of the answers on the second question or the coding. Please make sure that the file is well written and well documented. The submitted file should be in py or ipynb extension.

Both files must be submitted using MyPlace; no submission will be accepted in any different way. Any extension should be requested in advance of the submission deadline, with a valid reason. Assessments submitted after the deadline without an approved extension will receive a mark of zero.

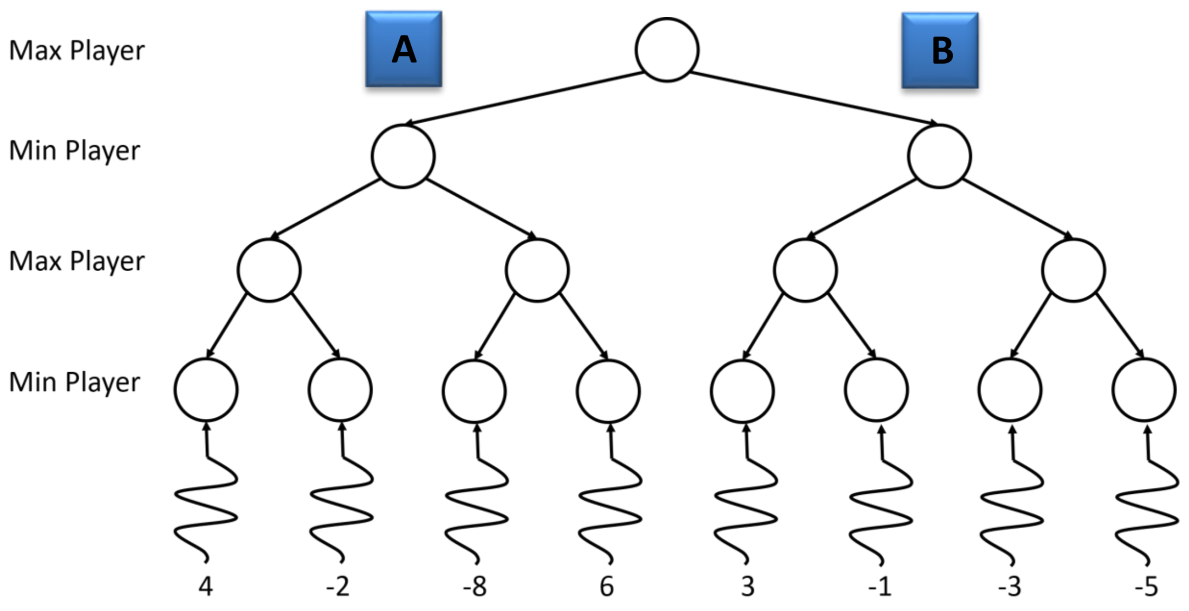
Good Luck!!

DUE: 12:00 noon, Monday November 22nd, 2021

CS814 AI for Autonomous Systems – Coursework 2

Part I (35% of the mark)

Let us consider the game tree shown below. For the avoidance of doubt: MAX plays first (top node), then MIN, then MAX. After these three moves, the max depth will be reached and will return an evaluation represented by the score shown at the bottom of the tree.



Your solution document on the two questions below should be a 1-page or 2-pages file in portrait PDF format. Any submissions not meeting this formatting requirement will receive a mark of zero.

Question 1 (15%)

Apply the Minimax algorithm on this game. Show the chosen value inside the nodes where these are calculated. Then indicates the action (A or B) that the Max player should play.

Question 2 (20%)

Perform minimax with alpha-beta pruning on this game. Show all v, α and β values next to the nodes where these are calculated. Show any prunes that are made by drawing a double line across the arc.

DUE: 12:00 noon, Monday November 22nd, 2021

Part II (65% of the mark)

Question 3

English draughts (British English) or checkers (American English), also called American checkers or straight checkers, is a form of the strategy board game draughts. It is played on an 8×8 chequered board with 12 pieces per side. The pieces move and capture diagonally forward, until they reach the opposite end of the board, when they are crowned and can thereafter move and capture both backward and forward.
Please check the following link on Wikipedia to have more information about the game: https://en.wikipedia.org/wiki/English_draughts

In this question, we are going to write a Python application to create an AI agent capable of playing the basic game (with no rule variations), and definitely aiming to win against a human player. The Rules of the basic game can be checked from the link mentioned above.

1. Choose a formalism to represent all possible states, including to check a winning state (and obviously identify the winner) (20%)
2. The search tree may be huge for this game. Choose an evaluation function for a non-terminal state (15%)
3. Implement the alpha-beta algorithm, and use the evaluation function when a certain depth is reached (20%)
4. Implement the main function of the algorithm that will allow a human being to play against the computer using the keyboard. (10%)
5. This question is optional / Bonus: Implement the Monte-Carlo Tree Search algorithm, and create a function allowing the two agents to play against each other and identify the winner (15%)