Artificial Intelligence Spring 2019 Homework 2: Adversarial Search

Please follow this format for submission. You may reproduce it in LaTeX or word if you wish

Name: Vikram Ho

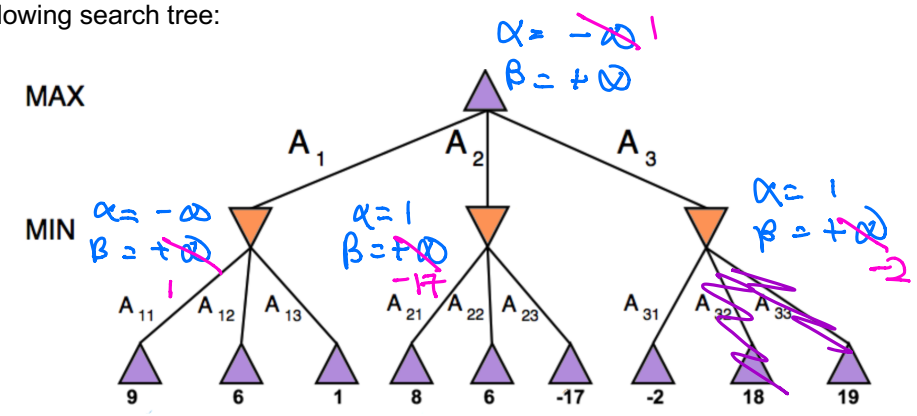
UNI: vwh2107

WRITTEN

Please justify all your answers to receive full credit (unless stated otherwise). There are 3 questions.

Question 1: Minimax and Alpha-Beta Pruning

Consider the following search tree:



a. Using minimax, which of the three possible moves should MAX take at the root node? What is the value of Max at the root?

MAX should take the move A1
Value of MAX at the root is 1

b. Using minimax with alpha-beta pruning, compute the value of alpha and beta at each node (show your work on the tree). Which branches are pruned?

Alpha and beta values are on the tree above.
Since α and β cross at A_{31},
we pruned A_{32} and A_{33}

Question 2: Iterative Deepening in Adversarial Search

Provide at least two reasons why **Iterative Depth Search** (also called **Depth First Iterative Deepening DFID**) is useful in solving adversarial two-player games like chess.

Suggested reading: Section 7 of Depth-First Iterative Deepening Korf 1985, found in you : https://courseworks2.columbia.edu/courses/76402/files/folder/READING?preview=2364236

Iterative depth search is useful for solving adversarial two-player games like chess for 2 main reasons
The first reason is IDS selects the maximum depth per iteration and computes the utility at that depth and sends that utility up the tree. Thus computation takes less time as searching through the entire depth of a tree before returning the utility “upwards” IDS minimizes at least asymptotically, time and space for any given search depth

The second reason is information in the previous iteration can be useful to decide the next possible move in the search space since we have the utility at each iteration, we can effectively prune the tree more efficiently. This saves time and space for searching through all possible search spaces.

Question 3: Checkers is solved!

Describe in the space below how checkers were solved. Explain what methods were used and how. You can provide a general algorithm or specific explanations. Use your best judgement to provide the key elements.
https://courseworks2.columbia.edu/courses/76402/files/folder/READING?preview=4056973

Checkers was solved using a combination of forward and backward search
For forward search, there were end game databases that provide starting positions for backward search
A tree traversal managers that keeps track of next and previous states to be search as with forward search
A proof solver that calculates the viability of forward search gives the position provided by the tree traversal manager

Using a combination of those strategies mentioned above, the algorithm evaluates the utility of a win, a lose or a draw of all game stages when the game is over, using the technique called retro grade analysis working backwards towards by enumerating all one piece, two piece and three piece positions and determining their value each time, towards the beginning.
At the same time, forward search uses alpha-beta pruning. The forward search consists of 2 elements, the tree manager and the proof solver. The tree manager hold a master copy of the proof and the solvers gets a position to evaluate from the manager. The solver uses two search programs to evaluate a position. The first is Chinook to determine a heuristic value for the position with alpha beta algorithm being the main algorithm.
Alpha-beta search algorithm pass the heuristic bounds on the value of a position by minimize the value of one side and maximize the value that the opponent can limit the side to move to.