CSCI 5511
Artificial Intelligence
FINAL PROJECT REPORT

Sri Krishna Vamsi Koneru
koner033@umn.edu
5881358

Group: I am doing a solo project by myself

Description:

I have focused on the Connect five(also known as gomoku) game. I used adversarial search algorithms Minimax and with Alpha Beta pruning to write multiple adversarial agents and then observed their performance. I made use of a random adversarial agent which took a random move everytime for testing purposes of my higher level agents.

I also used two different utility functions where one was partially complete while the other utility function checked for more parameters and scored accordingly. I have built the game interface from scratch based on the github aima code and randothello assignment. I implemented the command line output format similar to the randothello output and also took reference for the class templates. Then, I wrote the two utility functions found in the classes to check the impact of the various heuristics.

Initially, I focused only on the lengths of the connections(like sizes,4,3,2) but when I tried executing, each move was taking a lot of time due to the large state space of the problem. But, then I incorporated two other heuristics, center and corner heuristics which gives higher priority to cells being placed at center or corners of the board because based on what I read about the game, these two situations increase the chances of winning. One more thing I considered was the fact that there are too many possibilities in the initial states because simply the branching factor is very high in the initial states hence, I made a list of actions which considered empty center or corner cells and then a randomized choice was made among those legal actions.

I have written two utility functions, evalmove and eval1move where eval1move considers the center and corner scores as well along with all the heuristics in the evalmove function where I took into consideration the open ended 4 lines,3 lines, 2lines and also checked for the number of 1s available.

Review of Supporting Literature:
I implemented the minimax, alpha beta algorithms with the help of the textbook and I implemented the output similar to rand othello assignment. I also went through the following

publications and tried to implement specific concepts mentioned by me under the respective sources from which I gained the specific insights.

References:

Nasa, Rijul, Rishabh Didwania, Shubhranil Maji, and Vipul Kumar. "Alpha-beta pruning in mini-max algorithm–an optimized approach for a connect-4 game." *Int. Res. J. Eng. Technol* (2018): 1637-1641.

It had a theoretical approach of how to tackle the connect four game using minimax, alpha beta pruning. It also showed the vast number of iterations it took for connect four itself. It only meant that the number of iterations in connect five were on a very higher scale to that in the paper. It helped me in deciding the heuristics.

https://canvas.umn.edu/courses/390946/assignments/3438719

I have based the structure of my code on the RandOthello assignment
class structure and the output is also of similar format in the terminal which I implemented based on the rand othello assignment

https://github.com/aimacode and Russell, S. and Norvig, P. (2020). Artificial Intelligence. 4th Edition. [Insert Publisher Location]: Pearson Education (US).

I followed the book algorithm for the execution of the mini max algorithm and then alpha beta pruning for minimax.

Liao, Han. "New heuristic algorithm to improve the Minimax for Gomoku artificial intelligence." (2019).

Found some inspiration regarding the type of heuristics to be included in the utility function calculations

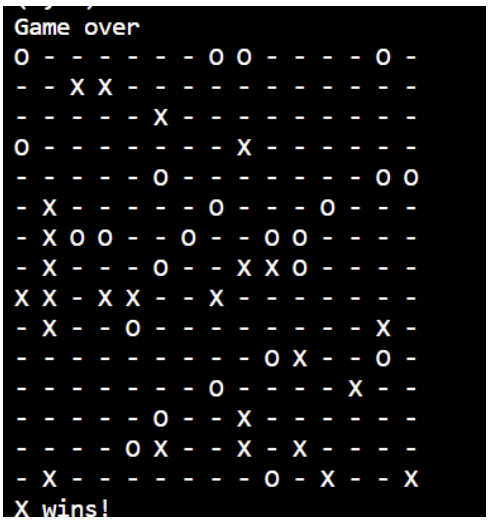
Software Needed:

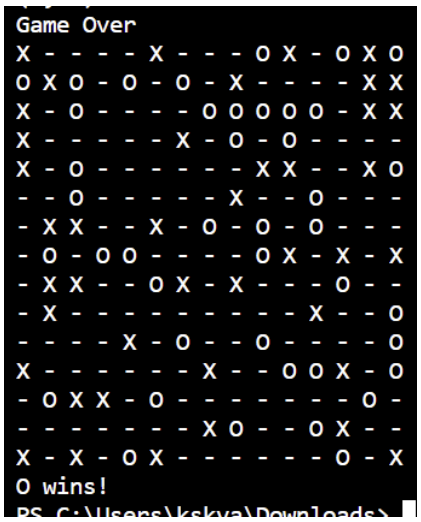
Python should be enough as everything is in a single .py file

Instructions:

Execute the .py file through the terminal.
In the play_game() method can pass the various class players as parameters.

Results:





The results represent the output of the code.

However, the minimax and alphabeta player classes are taking too long to execute even the first step. This might be due to the size of the state space in the initial steps. Hence, I am writing the following sections:

Expected Results:

Minimax performs definitely better than random choice because of the heuristics which never overestimate and also the algorithm in general.

With alpha beta pruning is expected to be way faster than the minimax implementation because of the size of the statespace and the parts of the statespace which do not need to be explored as opposed to mini max.

Walking through my thought process:

While, I do not have the proper results to vouch for it, I wanted to let you know the thought process behind the implementation of the project to express my understanding of the application of the learnings in the course via this project.

First, comparing the two evaluation functions, I implemented center and corner cells detection in one of the utility functions which is expected to perform better due to better scoring mechanisms. However, I feel I have not encountered the issue of state space size properly thus resulting in the issue at hand.

Coming to the **solution choice**, for connect 5 adversarial search is the right choice because in the game it is not just about playing for your own win but also to ensure the opponent's chances are also lowered. Hence, minimax is the perfect algorithm to ensure this and alpha beta pruning further increases the efficiency of minimax. However, I havent done any move ordering which might result in even better efficiency.

I also wanted to reiterate that I have implemented the minimax and alpha beta pruning algorithms based on the text book and thus believe there is nothing fundamentally wrong in the algorithm functioning. But rather the issue lies within the evalfunction.

I tried but I could not figure out how to reduce the state space without doing any hard coding of the values. It looks like I have underestimated the challenge of the task at hand as I felt it would be achievable once I had the interface running.