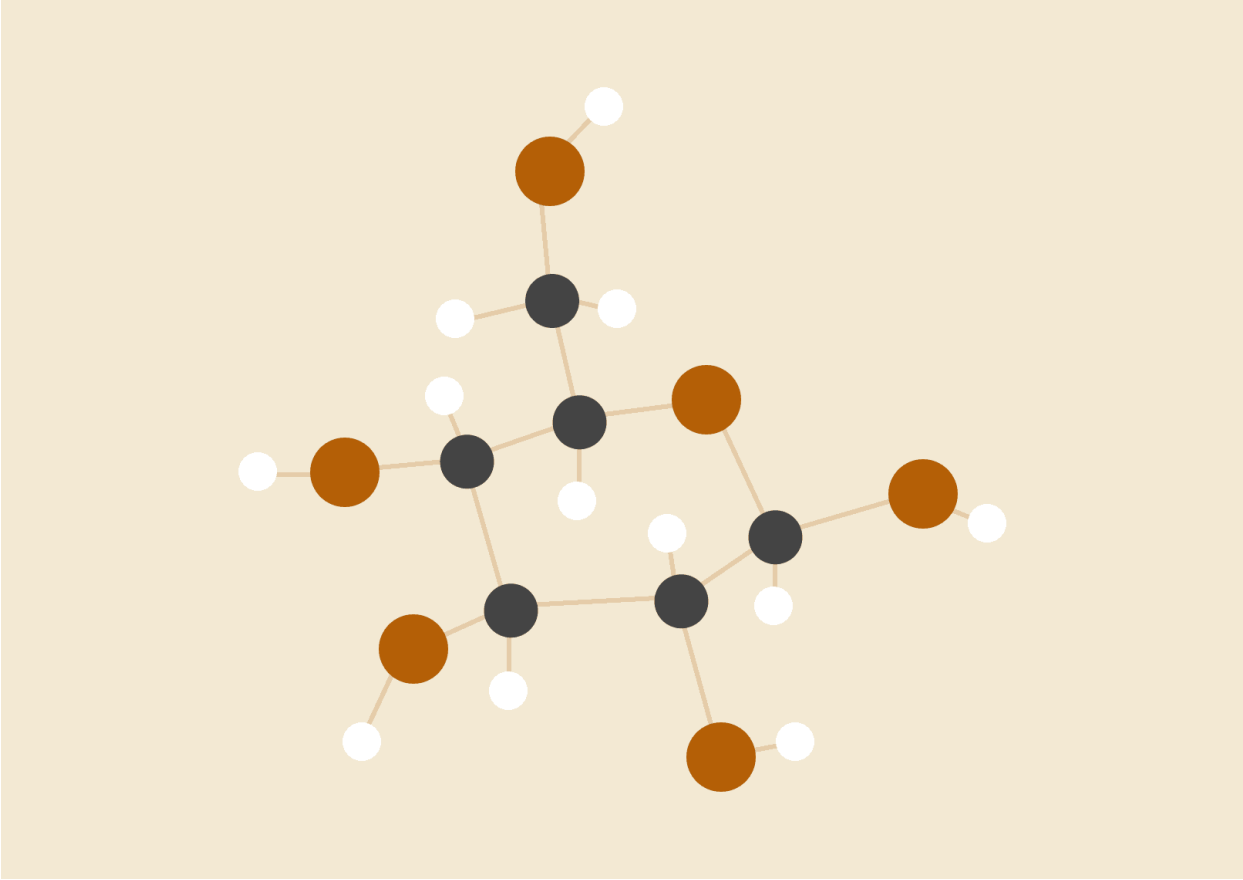
CSE 240 - AI Assignment 2

Alpha-beta and expectimax search algorithms



Héctor Carrión

01.31.2022

1926122

INTRODUCTION

In this assignment, I have implemented an agent to play the game of Connect-Four. My AI agent uses the alpha-beta or the expectimax search algorithms depending on the type of opponent it is facing. When it is playing against another AI or against a human it will utilize alpha-beta, when playing against a random player it will use expectimax. This report covers the questions on the assignment handout as well as discusses results data.

FILES

1. ConnectFour.py (starter, untouched code)
2. Player.py (custom implementations for Board class and moves functions)

USAGE

Hardware is a HP Chromebook with a 2.4 Ghz intel processor running a Linux VM. To play the game you need to run “python ConnectFour.py arg1 arg2” where arg1 and arg2 are one of AI, random, or human. For example if you wanted to play against a random player you would run “python ConnectFour.py human random”. If you wanted your AI to play itself you would run “python ConnectFour.py ai ai” ConnectFour.py takes one optional argument --time that is an integer. It is the value used to limit the amount of time in seconds to wait for the AI player to make a move. The default value is 60 seconds.

RESULTS - AI vs AI

The time shown here is the max recorded time for any move to the depth value. Algorithm is **alpha-beta search with pruning**. AI 1 goes first, AI 2 goes second. 3 games.

Depth	Time	Winner
6	< 100s	AI 2 x3
5	< 10s	AI 1 x3
4	< 3s	AI 2 x3

RESULTS - AI vs HUMAN

The time shown here is the max recorded time for any move to the depth value. The results are of **alpha-beta search with pruning**. The human is going first to give us the best possible advantage. 3 games ran per row

Depth	Time	Winner
6	< 100s	AI x3
5	< 10s	AI x3
4	< 3s	AI x3

RESULTS - AI vs RANDOM

The time shown here is the max recorded time for any move to the depth value. The results are from **expectimax search**. The random player is going first to give it the best possible advantage. 3 games ran per row.

Depth	Time	Winner
6	< 100s	AI x3
5	< 10s	AI x3
4	< 3s	AI x3

DISCUSSION

HEURISTIC FUNCTION

The heuristic used here considers ‘connected components’. Say for example the board has 2 discs of the same color placed next to each other, this would value the board at 1 point. For 3 discs the value is 10 and 4 discs is 100. The negative of these scores is true

when considering the value of the opponent's board. The function observes this connection for horizontal, vertical and diagonally placed discs.

This heuristic was chosen since this closely matches how humans would value a Connect-Four board. It could be improved upon by not valuing states in which 3 discs can be connected but 4 cannot (say because the edge of the board), for example.

PERFORMANCE

The amount of time constraints how deep we may search the tree. As you can see above, allowing for 100 seconds gives us a max depth of 6. This is the depth at which I saw the best decisions (starting disc placed on column 3, for example) though I did not test larger depths. The time requirement grows exponentially with the depth as we can see above.

I personally cannot beat the algorithm at depths greater than 4. In fact I also challenged friends who were unable to beat it.

FIRST ADVANTAGE

Connect-Four is technically a ‘solved’ game, meaning that the player that goes first can always win within 41 moves, regardless of what the opponent does. However, when playing AI vs AI this advantage didn't seem so clear. This is probably due to the fact that these agents are not optimal. As I mentioned before, the heuristic function could be improved in order to help the agents make optimal decisions.

REFERENCES

1. A Knowledge-based Approach of Connect-Four, Victor Allis 1988
2. Artificial Intelligence A Modern Approach, Fourth Edition