McGill University

Artificial Intelligence
ECSE 526
Dynamic Connect-4

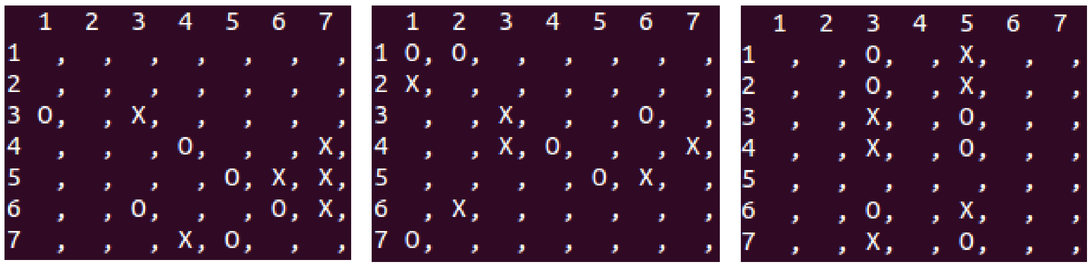
Russell Buchanan

September 30, 2015

1 Introduction

The purpose of this assignment was to design an agent to play a modified connect four game. The agent should first be designed to implement a Minimax algorithm and then an Alpha-Beta Pruning algorithm. I began by using the SimpleAI library by Juan Pedro Fisanotti in Python. However I found that it would be too difficult to modify the limited_depth_first search algorithm to work as a Minimax or Alpha-Beta Pruning algorithm. So I wrote my own algorithms based on the pseudo code available on Wikipedia here and here.

There are three Python scripts available: connect4.py which is the main game playing agent intended for playing against an opponent over the game server; interface.py which is intended for human interface with game server to play against an agent; and questionSolver.py which implements the exact same algorithm as connect4.py (minus the first layer, see 3. The Heuristic) and was used for answering the questions.



```
  1  2  3  4  5  6  7
1  ,  ,  ,  ,  ,  ,  ,
2  ,  ,  ,  ,  ,  ,  ,
3 O,  , X,  ,  ,  ,  ,
4  ,  ,  , O,  ,  , X,
5  ,  ,  ,  , O, X, X,
6  ,  , O,  ,  , O, X,
7  ,  ,  , X, O,  ,  ,
```

```
  1  2  3  4  5  6  7
1 O, O,  ,  ,  ,  ,  ,
2 X,  ,  ,  ,  ,  ,  ,
3  ,  , X,  ,  , O,  ,
4  ,  , X, O,  ,  , X,
5  ,  ,  ,  , O, X,  ,
6  , X,  ,  ,  ,  ,  ,
7 O,  ,  ,  ,  ,  ,  ,
```

```
  1  2  3  4  5  6  7
1  ,  , O,  , X,  ,  ,
2  ,  , O,  , X,  ,  ,
3  ,  , X,  , O,  ,  ,
4  ,  , X,  , O,  ,  ,
5  ,  ,  ,  ,  ,  ,  ,
6  ,  , O,  , X,  ,  ,
7  ,  , X,  , O,  ,  ,
```

Figure 2.1: From left to right: Quesiton A,B and C

2 Minimax vs Alpha-Beta

I ran both my Minimax and Alpha-Beta programs on the following start states. I tested with depths of 3,4,5 and 6. In the table you can see the results.

Configuration	Depth 3	Depth 4	Depth 5	Depth 6
A (Minimax)	5,212 States	82,409 States	1,295,917 States	20,343,008 States
A (Alpha/Beta)	547 States	1,569 States	10,204 States	35,308 States
B(Minimax)	2,874 States	40,449 States	579,330 States	8,443,491 States
B(Alpha/Beta)	354 States	3,982 States	21,251 States	57,609 States
C(Minimax)	3,567 States	56,732 States	914,085 States	14,881,853 States
C(Alpha/Beta)	975 States	5,972 States	48,594 States	271,373 States

3 Order of search and number of states

Switching the direction of state exploration should have no effect on the Minimax algorithm. This is because the Minimax algorithm will always search every possible state regardless of the heuristic. Searching left to right or right to left makes no difference. This is supported by the fact that when I reverse the direction of my Minimax function for configuration A and depth 4, I search the same number of states: 82,409.

Alpha-Beta Pruning on the other hand can be affected by direction of search. Since the algorithm is always trying to avoid looking at new states by comparing them to past states, different branches will be pruned depending on the direction of search. When I reverse my function for configuration A, depth 4 I find it explores more states: 2,453.

4 The Heuristic

The heuristic I used has two levels and was optimized for playing a full game from standard starting positions. The first level is inspired by popular chess solving agents and essentially hard codes the first three turns of the game. Since the opening moves are too far from the end game I recruited some friends to act as researchers and play a mock up of the connect 4 game. They played several matches and evaluated the best opening moves for both colours. From their feedback I came up with the following opening strategy:

1. White's first move should be one of the center pieces with respect to row
2. Black's first move should be one of the center pieces and on the same side as White's move
3. White's second move should be one of the center pieces and on the opposite side of it's first move

Figure 4.1: "Researchers" evaluating opening move strategies

The second layer of the heuristic is a traditional heuristic evaluation. I have prioritized getting four-in-a-row and three-in-a-row by evaluating them at +/-1000 and +/-50 points respectively. Two-in-a-row and positioning a piece in the center column are valued equally at +/-20. This is because I wanted to encourage the agent to move pieces towards the center during the early game even if they could get two pieces beside each other on one side of the board. Because of the starting layout if two-in-a-row is too valuable, the agent will try to make several two-in-a-row combinations at each end of the board.

At first I tried creating a heuristic for each player but found the agent performed better when it was penalized for allowing the opponent to make good moves. This means it will actively attempt to block its opponent from making a winning move.

Finally, I also evaluated +/-10 and +/-15 points for moving pieces 1 and 2 steps closer to the center. This gets most of the pieces into play and allows for more four-in-a-row combinations.

5 Computational Trade-off

My heuristic evaluation is fairly simple, evaluating two different qualities: how close pieces are to the center and how many are in-a-row. This is fairly computationally taxing however as it must go through 4 for-loops and evaluate every possible combination of multiple pieces in-a-row. However, it is necessary to prioritize a wining state. A simpler heuristic based only on closeness to the center might never end the game.

Because of this complexity it becomes harder to search at greater depths as evaluating each move will take much longer. Searching at greater depths does provide an advantage because you can plan more moves ahead, but if the quality of the heuristic is too simple, an agent wont be able to exploit this advantage. A compromise must be made between sophisticated heuristics and search depth.

6 Collaboration with other Students

I discussed high-level heuristic strategies with a friend who had taken this course in a previous year.

I found a checkers board and covered one row and one column. I then went to several friends, asking them to play the game in real life and develop human level strategies. Based on their feedback I implemented an opening move set.