Homework 5

Due Mar 20 by 11:59pm **Points** 14 **Available** Mar 8 at 8am - Mar 23 at 11:59pm 16 days

Your task this week is to implement an AI agent who plays a perfect (or near perfect) game of an arbitrary sized Tic Tac Toe.

The mechanics of the game have been implemented for you. The starter code is available in **HW5.zip**.

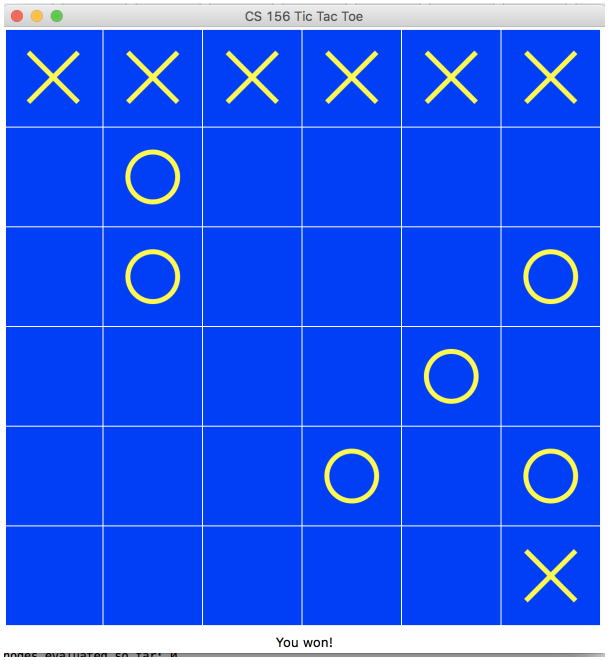
Once you download the code, you can play and win a game (of any size) against a non-optimal AI agent who picks moves randomly:

python tictactoe.py 6 rand

Note that if you have both Python 2 and Python 3 installed, you may need to type:

python**3** tictactoe.py 6 rand

As you can see below, this is not a challenging game to play.



Your task is to implement an AI agent who will play a better game.

1. Implement the minimax algorithm by filling in your code in the following 4 functions in adversarial_search.py: *minimax*, *value*, *max_value* and *min_value.*

   You can test your minimax code on a 3x3 Tic Tac Toe as follows:
   python tictactoe.py 3 minimax
   Even though your agent plays a perfect game and cannot be defeated, it searches the entire tree from the current game state to find the best move. The program prints the number of nodes evaluated with each move:
   Number of nodes evaluated so far: 59,704
   Number of nodes evaluated so far: 60,756
   Number of nodes evaluated so far: 60,802
   Number of nodes evaluated so far: 60,806
   Total number of nodes evaluated: 60,806
   Note that the number of nodes will be different for each game - but of the same order of magnitude (around 60,000).
   You can try your minimax agent with a 4x4 Tic Tac Toe but you'll see that it takes too long to decide on a single move. We'll have to do better.
2. Implement the minimax algorithm with Alpha Beta pruning by filling in your code in the following 4 functions in adversarial_search.py: alphabeta, abvalue, abmax_value and abmin_value.

   You can test your Alpha Beta code on a 3x3 Tic Tac Toe as follows:
   python tictactoe.py 3 alphabeta
   You will see that the game is much faster now and the number of nodes expanded has been significantly reduced.
   Number of nodes evaluated so far: 3,808
   Number of nodes evaluated so far: 4,172
   Number of nodes evaluated so far: 4,213
   Number of nodes evaluated so far: 4,217
   Total number of nodes evaluated: 4,217
   However we are still unable to play a 4x4 Tic Tac Toe.
3. Implement a depth limited alpha beta algorithm by filling in your code in the following 4 functions in adversarial_search.py: abdl, abvalue_dl, abmax_value_dl and abmin_value_dl. You can use the evaluation function that has been implemented for you in tictactoe.py (the eval method

in GameState class) to evaluate non terminal nodes. However for the terminal nodes (win and lose) you will have to decide on values that are consistent with the values retuned by eval. Make sure you understand what eval is doing first.

You can test your implementation code on a 4x4 Tic Tac Toe and with a depth of 6 as follows:

python tictactoe.py 4 abdl 6

Number of nodes evaluated so far: 288,767
Number of nodes evaluated so far: 408,462
Number of nodes evaluated so far: 445,792
Number of nodes evaluated so far: 453,185
Number of nodes evaluated so far: 455,072
Number of nodes evaluated so far: 455,279
Number of nodes evaluated so far: 455,290
Number of nodes evaluated so far: 455,291
Total number of nodes evaluated: 455,291

You will find that your agent is still unbeatable. You can even test your implementation on a 7x7 Tic Tac Toe with a depth of 3 as follows:

python tictactoe.py 7 abdl 3

What depth do you have to go down to to beat yor abdl agent?

Homework 5 Grading Rubric

Criteria	Ratings		Pts
Minimax *The 4 functions are implemented correctly*	4.0 pts Full Marks	0.0 pts No Marks	4.0 pts
Alpha Beta *The 4 functions are implemented correctly*	4.0 pts Full Marks	0.0 pts No Marks	4.0 pts
Depth Limited Alpha Beta *The 4 functions are implemented correctly. The value of the terminal states are consistent with the evaluation function. The depth is handled correctly.*	6.0 pts Full Marks	0.0 pts No Marks	6.0 pts
			Total Points: 14.0