AI - Assignment 6

Patrick Nagel: patrick.nagel@h-brs.de
Amirhossein Pakdaman: amirhossein.pakdaman@smail.inf.h-brs.de

Due Date: 15.12.2019

0.1 Programming Assignment: Connect-4 game

1. Implement a simple Connect 4 game to demonstrate the use of adversarial search for deterministic, fully observable, two player turn-taking zero-sum games. For more details read here https://en.wikipedia.org/wiki/Connect_Four

2. For the adversarial search problem, implement a minmax with alpha-beta pruning agent that allows you to play against the computer.

3. Compare these two approaches (with and without alpha-beta pruning) based on their search time, space requirement and other information that you think is important.

0.2 Implementation Details

- The Connect4 class in helper.py is partially implemented for you. Please implement *check_for_horizontal_streak*, *check_for_vertical_streak*, *check_for_diagonal_streak* in order to check how many consecutive coins per player are available on the board.
- Please implement *get_evaluation_func* to obtain the utility value per player based on **all the streak values**. For instance if an evaluation is made based on the vertical streak alone, then a player having 3 consecutive coins in the vertical streak should be given a higher value by the evaluation_func as compared to a player with 2 consecutive coins. For our case, the evaluation_func must consider all the 3 streaks while evaluating the players.
- The switching between players alternatively, checking for end of game, etc. are some tasks that should be implemented under *play_game* function.
- The minimax and alpha-beta pruning algorithm should be implemented in *minimax* and *alpha_beta_pruning* functions respectively.
- Please note that your algorithm should work for boards of any size. The default size in the given code is 6×7. In order to change the board size, please refer to the travis.yml for command line arguments.
- **You need not implement a nice GUI. A simple interactive console interface is sufficient.** Alternatively, you can use the printing board snippet from the main function to display the board.
- Since the game is a simulation of a real Connect4 game, **it must follow the laws of gravity**. A coin when dropped into the board falls into the bottom-most row, if empty, otherwise, on to next non-empty row. https://en.wikipedia.org/wiki/Connect_Four shows an animation of how the coins are stacked one on top of the other.
- Please add comments explaining your code as and when required. Also fill in the docstrings for the implemented functions.
- You may add additional functions if required and modify the parameters of the existing functions.

- Any extra functions required as per your implementation should be included in helper.py. Please add the necessary docstrings and keep your code as clean as possible.
- Please refer to the travis.yml for command line options to run your code.