Donghyun Park

Prof. Sable

Artificial Intelligence

11/5/17

Project #1: Othello Game AI

The game Othello was recreated in C++, printing out to the shell a square ASCII board. Compiling of the program can be accomplished by calling the Makefile attached with the rest of the components, which are split mainly into board.cpp and game.cpp. The board.cpp mainly handles the initialization of the game board through a class implementation (class Board), along with other things such as searching for legal moves (discoverMoves), applying chosen legal move, and flipping pieces (makeMove) accordingly. It also facilitates the alternation of players (nextPlayer), skipping either the player or the AI’s turn if there are no legal moves to be had. There are two private methods in the Board class, which are (checkOnBoard) and (iterate). As the name suggests, the prior checks whether a given location is within the board state, while the latter iterates through the surrounding locations of a piece to see whether a location is free or not. The end state of the game is checked by the method (checkEnd), and at every turn, the board’s current state is shown by (print).

With this board.cpp framework, game.cpp combines the facilitating of the actual game of Othello and the AI that plays against the player. There are only two public methods, which set up and operate Othello in a loop, while the private methods implement the AI which runs on a minimax alpha-beta search algorithm with iterative deepening. The (heuristic) method calculates the estimated cost of a move, while the (alphaBeta) method implements the minimax alpha-beta search algorithm via recursive iterative deepening. These two files are combined in the main.cpp, which take in the options of the user from the command line. The (setGame) method sets up the board state, and depending on whether the player imports a pre-set board, it initializes the game accordingly. The (playGame) method puts the game in a looping state for alternating turns.