Assignment 2: Othello

CS6380 : Artificial Intelligence

Due: Trial Round - 8 November 2020

Problem Statement:

In this assignment, you will code a bot to play and win the game of Othello. Given a board configuration and a turn, your bot will return a valid move. The game ends when neither of the players can make a valid move. The player with maximum number of coins is the winner.

Specifications:

The following programming language and system(OS) has to be used for completing the assignment.

- **Programming Language**: C++
- **System**: 64-bit Linux distribution.

Instructions:

Setting up the framework:

A framework (**Desdemona.zip**) has been provided to you that lets two bots compete against each other.

1. Extract the contents of **Desdemona.zip** into a suitable directory.

2. Set up the framework by executing a **make clean** command in the root of this directory which removes the older bin,lib and obj directories and then execute a **make** command to generate the executables and libraries needed for coding your bot.

Coding the bot:

1. You will modify **MyBot.cpp** to return a valid move whenever the function **play** is called. The source file is located at (**./bots/MyBot**).
2. All other source files are to be left untouched.
3. The **Makefile** is also provided at this location. Use it to generate a **.so** file by executing the command **make** from the location **./bots/MyBot**.
4. You can test your bot against another bot by executing the command "**./bin/Desdemona ./<path to bot1.so> ./<path to bot2.so>**"
5. By convention, the first bot is **BLACK** and the second **RED**.
6. A random bot (**./bots/RandomBot**) has been provided for testing.
7. At the end of the game, a **game.log** file is created that contains the sequence of moves made.
8. The bots being submitted must have **NO** print statements.
9. If a bot returns an invalid move, it will be disqualified.

Helper Functions:

The following functions have already been written to assist you:

- **bool OthelloBoard::validateMove(Turn turn, int x, int y)** Returns true if the passed move (x,y) is valid for the passed turn, false otherwise.
- **bool OthelloBoard::validateMove(Turn turn, Move move)** Returns true if the passed move is valid for the passed turn, false otherwise.

- **void OthelloBoard::makeMove(Turn turn, int x, int y)**
  Updates the board configuration by making the move (x,y); throws an exception if the move is not valid.
- **void OthelloBoard::makeMove(Turn turn, Move move)**
  Updates the board configuration by making the specified move; throws an exception if the move is not valid.
- **list<Move> OthelloBoard::getValidMoves(Turn turn)**
  Returns a list of valid moves that can be made given the turn.
- **int OthelloBoard::getBlackCount()**
  Returns the number of black coins on the board.
- **int OthelloBoard::getRedCount()**
  Returns the number of red coins on the board.
- **void OthelloBoard::print(Turn turn)**
  Prints the turn, the board configuration, and the number of black and red coins. 'X' is **BLACK**, 'O' is **RED**, and unfilled locations are blank.

Time constraints:

Each bot can take at most 2 seconds to return a move. If this time limit is exceeded, the bot causing the timeout will be disqualified.

Tournament Details:

Bots submitted by all groups/individuals will be contestants. In both the trial round and final round, each bot will play against every other bot twice; once as the first player (black), and once as the second (red). At the end of the tournament, each bot will be given a rank based on the total points scored.

Evaluation:

Based on the outcome of the game, an individual/group will score the points per game as mentioned below:

Win/Lose

- **Winner:** 64 + (Number of coins placed on the board)
- **Loser:** Number of coins placed on the board.

Draw

- Both the teams will get points equal to number of coins placed on the board.

Disqualification

- Disqualifying team: -64 + (Number of coins placed on the board)
- Opponent team: 64 + (Number of coins placed on the board)

Submissions:

- **Trial Round:**
  Upload a single .so file with the name <**GROUPNO**>**.so** or <**ROLLNO**>**.so** (eg: 23.so or CS17B001.so)
- **Final Round:**
  Upload a zip file containing your source code(i.e MyBot.cpp), report as well as the <**GROUPNO**>**.so** file. The zip file should be named as <**GROUPNO**>**.zip** or <**ROLLNO**>**.zip** (e.g 23.zip or CS17B001.zip)

Deadlines

- **Trial Round - 23:59 8 November, 2020**
- **Final Round - 23:59 15 November, 2020**