Mini Project #3

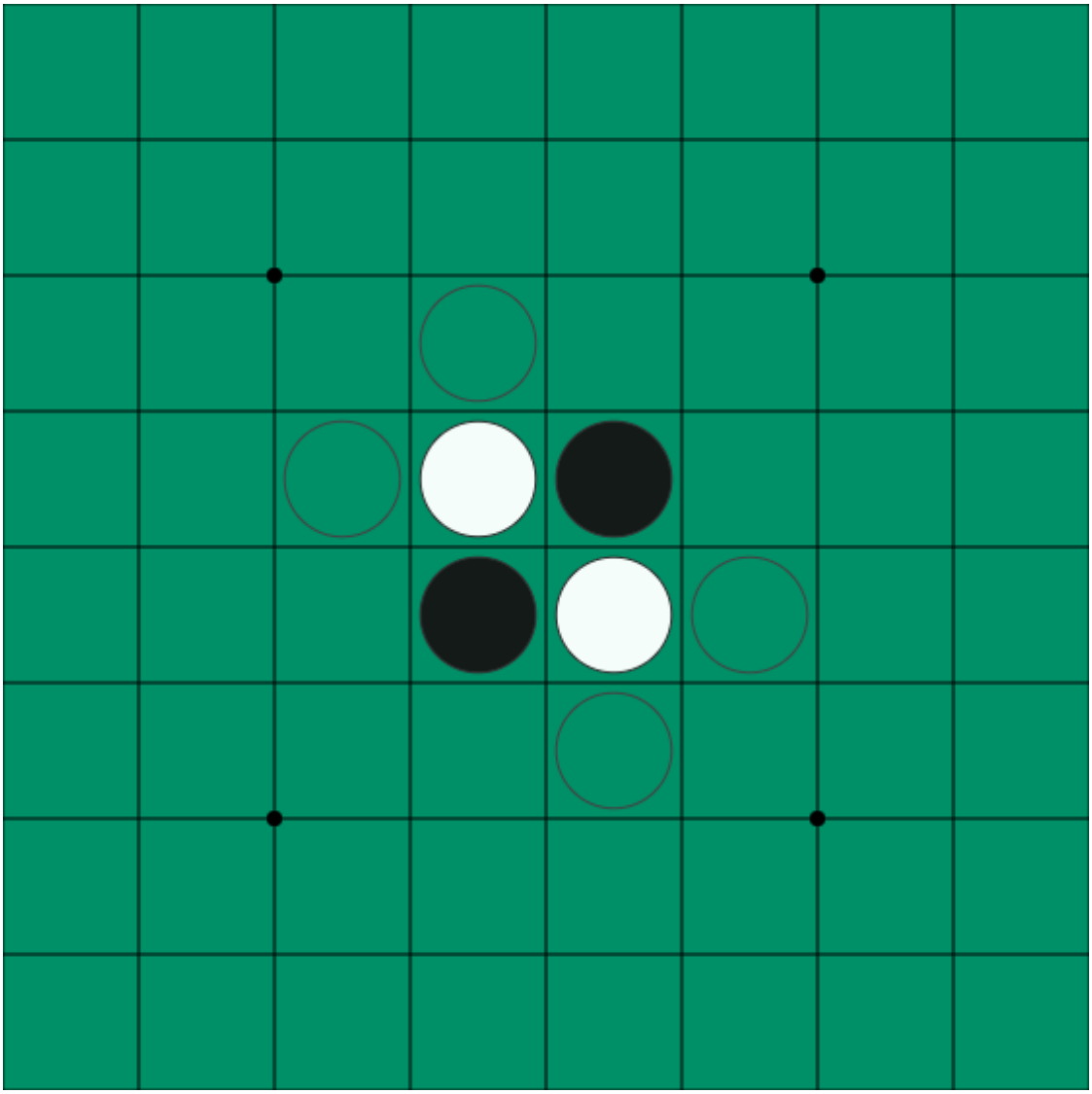
Othello A.I.

Due: 2021/06/28 (Mon.) ~~23:59~~ 11:59 am

Problem Description

Design and implement an A.I. that can play the boardgame Othello. (Reversi)

If you are not familiar with the rules of Othello, you can learn it online at EOthello.



Requirements

You must enhance your A.I. policy through the following methods:

1. Tree Search (e.g. Minimax, Negamax, Alpha-Beta Pruning)
2. State Value Function design (or State-Action Value Function design)

If you are not satisfied by the approaches above, you can try out more advanced methods to defeat Strong A.I. baselines. Some nice references:

1. Monte Carlo Tree Search
2. Value & Policy Networks
3. Silver et al., 2017, Mastering the game of Go without human knowledge, Nature'17.
4. Silver et al., 2017, Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm
5. And more…

For the brief tutorial on Tree Search, Minimax, Pruning, please refer to the Introduction Slides.

Rules

1. If your program outputs invalid moves, you lose and the game ends immediately.
2. Time limit for each move is 10 seconds, and the memory limit is 4 GB. (Note that the 10 second duration may vary depending on different processors.)
3. You can keep overriding your move during the time limit. Only the last successful output move is selected.
4. You are not allowed to use the following:
    a. External API/libraries (e.g. Boostm system calls)
    b. Inline assembly (e.g. asm)
    c. Vectorization instructions (e.g. SSE, AVX)
    d. Multi-threading (e.g. OpenMP, pthread)
    e. GPU acceleration (e.g. CUDA code)
    f. Networking (e.g. Sockets, MPI)

Program Submission

1. Please use C++ and write your program in a single source file.
2. Your source file must be named as "`<Student_ID>_project3.cpp`" and please make sure that all characters of the filename are in lower case. For example, if your student id is `109062000`, the name of your program file should be `109062000_project3.cpp`.
3. Your program will be compiled in a GNU/Linux environment with:
   `g++ -O2 -std=c++14 -Wall -Wextra <Student_ID>_project3.cpp`
4. The source file must be uploaded directly, without compressing the file.

5. 0 points will be given to Plagiarism. NEVER SHOW YOUR CODE to others and you must write your code by yourself. If the codes are similar to other people's and you can't explain your code properly, you will be identified as plagiarism.
6. 0 points will be given if you violate the rules above.

Report

1. Elaborate on how you design your Othello A.I. (e.g. tree search, state value function, …) The report is not graded directly, but indirectly through TA demo. So, make sure you have enough figures to help you explain your method to TA.
2. ~~You can only refer to your report during the demo. Referring to your code is not allowed, so make sure you have taken a screenshot of the important code snippets.~~
3. ~~The report filename must be "`<Student_ID>_project3.pdf`" and please make sure that all characters of the filename are in lower case~~.

Grading Policy

1. The project accounts for 9 points of your total grade.
2. You must submit both your source code and report. Remember the submission rules mentioned above, or you will be punished on your grade. No late submission is allowed.
3. Your code will be tested against the A.I. Baselines after the submission deadline.
4. For Tree Search, State Value Function Design, and Alpha-beta Pruning, make sure to include a screenshot of your code in your report, and elaborate on them during demo. The scores are given if you have implemented them correctly.
5. For the Version Control bonus, register an account on GitHub and learn how to use it. The scores are given if you include a screenshot with more than 3 commits. (We strongly recommend trying this out, since this bonus score is pretty easy to get. Furthermore, using version control tools (e.g. Git) is necessary for becoming a good programmer.)
6. The bonus score can exceed the total 9 points.

- A.I. Baselines +5 points
- Tree Search (Minimax) +2 points
- State Value Function Design +1 point
- Alpha-beta Pruning +1 point
- Version Control (Bonus) +1 point
- Class Ranking (Bonus) +2 points