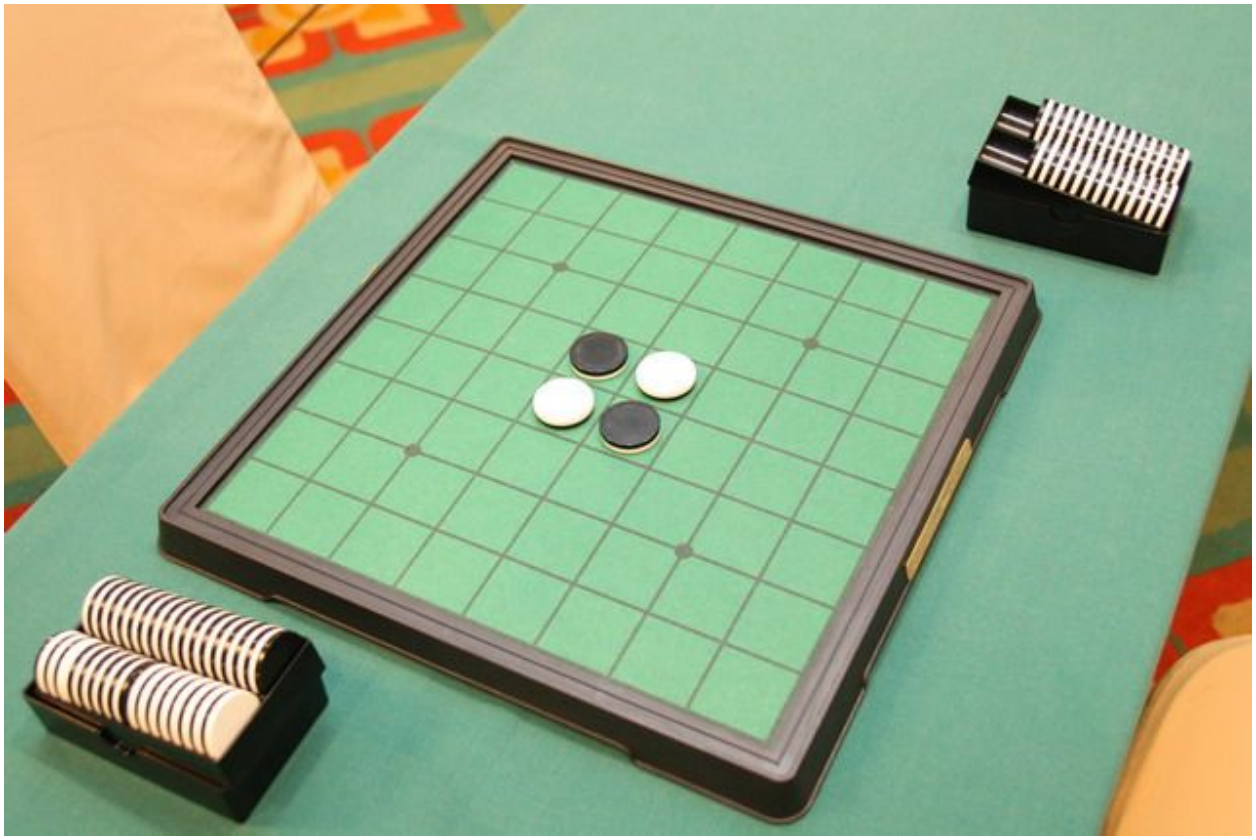

The picture comes from https://en.wikipedia.org/wiki/Reversi

Better AI for Reversi

1. **Project participants:**

Xiaocheng Zhang | Hang Guo.

In order to balance the workload, we separated the preparation into three parts:

1. We did brainstorm and outlined a list of topics that we are interested in and acceptable (23rd February). But some topics that attracted us don't have a relation specified a correspond to Artificial Intelligence, we couldn't pick up the topic directly.
2. In order to start a project proposal ASAP. We tried to connect with Professors and TAs. Hang posted questions about picking topics on Piazza, and Xiaocheng emailed to the professor.
3. After filtering out some inappropriate topics, we discussed and picked up a topic about doing researches about using multi-implementations in a zero-sum game: Reversi (2nd March).

2. **Problem description:**

In this project, we focus on a kind of chess game, Reversi, also known as Othello. It's a simple zero-sum game with two players. And the object is to have the majority of disks turned to display a player's color. In our project, we are

going to compare some existing AI programs which are used to solve this problem in perspective of algorithms, time complexity, space complexity as well as success rate.

What's more, we will try to do some improvements based on current programs or even come up with new methods. For instance, it seems that most of the existing programs are using Markov Decision Process(MDP). Can we develop some new heuristic methods? Moreover, is it possible that we introduce Reinforcement Learning(RL) into this topic and develop an algorithm? How can we make it? What're the advantages and drawbacks of using RL? Can we make it and beat those MDPs agents?

To be specific, the input of our programs will be an MDP with states S, actions A(s), transition model P(s'| s, a) and rewards R(s, a, s'). The output will be a policy guiding the agent on how to move.

3. **Algorithms:**

During the implementation part, we planned to build a Reinforcement Learning Agent, which treated the total board and opponent's reflection as part of the environment, and this opponent is an MDP Agent. We will try to implement the Minimax algorithm, Alpha-Beta pruning, Expectiminimax algorithm for the regular MDP Agent with different heuristic methods to train the Reinforcement Learning Agent.

For the MDP Agent, the Minimax algorithm takes more time with more accuracy. Using this algorithm is a good way to test the ability of the RL Agent.

Alpha-Beta pruning and Expectiminimax is efficient but not optimal enough. So they can be good opponents which will train the RL Agent.

4. **Libraries and tools:**

As we are going to compare the existing implementations of the Reversi problem and develop something new, tons of articles are required. In the preparation stage, we come up with the rough idea after reading these essays:

Armanto, H., Santoso, J., Giovanni, D., Kurniawan, F., & Yudianto, R. (2012). Evolutionary Neural Network for Othello Game. *Procedia-Social and Behavioral Sciences, 57*, 419-425.

Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W. M., Perolat, J., Jaderberg, M., & Graepel, T. (2019). Open-ended learning in symmetric zero-sum games. *arXiv preprint arXiv:1901.08106.*

Zhang, Z., & Chen, Y. Searching Algorithms in Playing Othello.

Zwer H.(2016) AI Design of Reversi. Retrieved from http://hzwer.com/7865.html

Also, we have searched some existing implementations on Github.

Arminkz, Reversi, 2018, Retrieved from https://github.com/arminkz/Reversi

Zolomon, reversi-ai, 2018, Retrieved from https://github.com/Zolomon/reversi-ai

In future research, we'll definitely take more articles as reference and learn from platforms like Google scholar and Github to make a better comparison of algorithms.

5. **Results:**

What is the ideal outcome of the project? What results do you expect to show? What comparisons will you do? Are there risks for not getting all the results? If so, what will you do about it?

After a brief search, we find that Othello is already solved on 4x4 board, 6x6 board. And although not yet proven by mathematics, Othello 8x8 can be solved with "top programs on fast parallel hardware". Thus, it's not a problem if we want to come up with a method that can solve 4x4 or 6x6 Othello efficiently. And about 8x8 Othello, we may just stay in a theoretical aspect and discuss the pros and cons of different algorithms rather than truly implement them on the device.

Alos, after looking up essays and codes on Google Scholar, we found most AI agents for this type of game (turn-based, zero-sum game). According to our design, because most existing algorithms are about MDP, so the ideal outcome is that we generated a RL Agent that can beat all three types of MDP Agents in a win-rate higher than 50%. If a Reinforcement Learning Agent failed to beat those algorithms, we will try to improve those existing algorithms according to the data that we collected from doing research about the win-rate and efficiency of different MDP Agents prior.