Computational Physics Lab Final Project Final Report
Ising-Model-Based Algorithm for Othello Strategy

Jiun-Cheng Jiang and Wen-Ning Wan
Department of Physics National Tsing Hua University, Taiwan
(Dated: January 15, 2023)

I. INTRODUCTION

A. Othello

1. *Rule*

Othello is a two-player strategy board game played on an 8×8 board. Each player has pieces, which are black and white. The game starts with four pieces in the middle of the board, two black and two white. The players take turns placing pieces on the board with their assigned color facing up.

A piece can only be placed on an empty square if it can flip at least one of the opponent's pieces. To flip the opponent's pieces, the new piece must be placed so that an straight (horizontal, vertical, or diagonal) line connects it with at least one of the opponent's pieces. All of the opponent's pieces between the new piece and the connected piece are then flipped to the new piece's color. The object of the game is to have the majority of pieces turned to display your color when the last playable empty square is filled.

The game ends when neither player has a legal move. The winner is the player with the majority of pieces on the board. If both players have the same number of pieces, the game is a draw.

2. *Strategy*

The strategy of Othello is made around the concept of mobility and stability. Mobility is the number of legal moves a player can make. Stability is the number of pieces that cannot be flipped by the opponent.

a. Grab corner Since there is no way to flip the corner pieces, the player who has the corner pieces has a great advantage. Therefore, the corner has the highest stability and has the highest priority to be grabbed.

Additionally, pieces next to the corner piece are also cannot be flipped by the opponent. Therefore, these pieces has the second highest stability.

b. X square The X square is the square surrounding the corner. The player should avoid placing pieces on the X square since it is the most vulnerable square. The opponent can have the chance to occupy the corner if the player places a piece on the X square. Therefore the X square has the lowest stability.

c. X edge The X edge is the edge surrounding the X square. Keeping X edge may make the opponent places pieces on the X square, which help the player to occupy the corner. Therefore, the X edge has high tactical value but low stability.

d. Danger zone The danger zone is the next rows in from the four edge rows. We should avoid placing pieces on the danger zone since it would help the opponent occupy the edge which has high stability. Therefore, the danger zone has low stability.

e. Mobility Mobility is the number of moves the player is currently able to make, which has significant weight in the opening of game, but diminishes to zero towards the endgame.

3. *$\mathbb{Z}2$ Gauge Theory*

Othello can be viewed as discretization of $\mathbb{Z}2$ gauge, since there are some common characteristics.

Gauge invariant states of $\mathbb{Z}2$ gauge theory is equivalent to the state of Ising model on the board. The flipping on a board corresponds to acting the unit strength magnetic flux on the board. Note that flipping twice makes it back to white due to $\mathbb{Z}2$. [1]

B. Ising Model

The Ising model has played an important role in the evolution of ideas in statistical physics and quantum field theory.

In order to define Ising model, consider a two dimensional square lattice, on each site or node of the lattice we have an atom or a magnet of spin s_i. In the Ising model, spins have two possible values, up or down which we map to the numerical values $+1$ or -1. The Hamiltonian of the system is given by

$$\hat{H} = -J \sum_{\langle i,j \rangle} s_i s_j - \mu \sum_i s_i \tag{1}$$

The first term is the spin-spin interaction and for J > 0 the system is ferromagnetic. The minimum energy E_0 is obtained for the ground state, which is the unique state in which all spins point in the direction of μ. The minimum energy E_0 is obtained for the ground state, which is the unique state in which all spins point in the direction of B. This is equal to

$$E_0 = -(2J + \mu)N \tag{2}$$

where N is the number of spins in the system.

The partition function is then given by

$$Z = \sum_{\{s_i\}} e^{-\beta H_{s_i}} \tag{3}$$

where $\{s_i\} \equiv \{s_1, s_2, \ldots, s_N\}$ is a spin configuration of the system.

C. Ising-Model-Based Algorithm for Othello Strategy

1. *Concepts*

Since the Othello can be viewed as discretization of $\mathbb{Z}2$ gauge, we can use the Ising model to describe the configuration of Othello game.

2. *Ansatz of the Hamiltonian of Ising Model for Othello*

We make an ansatz of the Hamiltonian of Ising model for Othello, which is

$$\hat{H} = -\sum_{i,j} J(i,j) s_i s_j - \mu(t) M \tag{4}$$

where the first term determined the stability while the second term determined the mobility, and $J(i,j)$ is the interaction between i-piece and j-piece which is

$$J(i,j) = \sum_p r_t s_p^{(ij)} \tag{5}$$

$s_p^{(ij)}$ is the pieces between i-piece and j-piece. r_t is the relevant weight the tactical value of each piece while the middle-16-pieces is set to 1. From Sec. I A 2, we can obtain the following relation: $r_{corner} \geq r_{fixed} \geq r_{Xedge} \geq r_{edge} \geq r_{dangerzone} \geq r_{Xsquare}$. The equal sign only happens in endgame and at the time $r_t = 1$. M is the number of moves the player is currently able to make, and $\mu(t)$ is a time dependent coefficient used to weight the first term and second term, which should be a decay function.

3. *Evolution Method*

Every time the player makes a move, there would be a state transition. We can just calculate the energy change instead of calculating the total energy each time.

As a result, for each move what we need to consider is the interaction between placing piece and placed pieces and the the change of mobility. And then all we have to do is to find the move which make it to the lowest energy, which is the most stable, in the end.

II. GAME ALGORITHM

We knew the composition of Hamiltonian for Othello, and then, we are going to find out the value of interaction terms and weighting term.

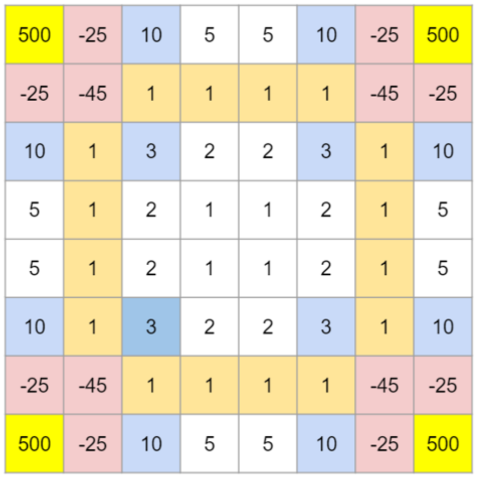
Our program can be categorized into main, algorithm, and Othello. ”main.py” is used to holding the main process, including player to player, player to Ising-Model-Based Algorithm, and other strategies we design to test our algorithm, which will be introduce later, versus Ising-Model-Based Algorithm. Algorithm part handles all the information and calculation of that the algorithm need to make one move. The process starts from training the model, getting the Hamiltonian to predicting the moves of opponent.

A. Implementing Ising Model into Model

Since the Hamiltonian implies the stability of the configuration in Othello, both sides, black and white, have to try to reach the extrema to win the game. However, this may make both sides take the move that helps opponent to reach the minimum of Hamiltonian. To distinguish the strategy for two sides, we make a new definition of Hamiltonian for white which take the opposite number of original Hamiltonian, and therefore, white side have to reach the maximum of Hamiltonian while black side reaching the minimum.

B. Obtaining Each Term in Hamiltonian

Recalling Eqn. (4), $-\sum_{i,j} J(i,j) s_i s_j$. $s_i s_j$ is the summation of the product of one piece and every other pieces. J is the interaction between two pieces, which can be taken as the value of strategy. Here we use a value map, shown in Fig. 1, as the initial value of J. Besides, μM



500	-25	10	5	5	10	-25	500
-25	-45	1	1	1	1	-45	-25
10	1	3	2	2	3	1	10
5	1	2	1	1	2	1	5
5	1	2	1	1	2	1	5
10	1	3	2	2	3	1	10
-25	-45	1	1	1	1	-45	-25
500	-25	10	5	5	10	-25	500

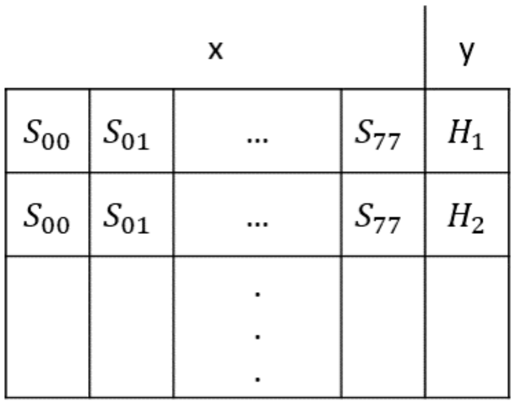
FIG. 1. The initial value map of interaction terms.

is the mobility of next round. Therefore, it should be subtracted to Hamiltonian.

1. *Machine Learning*

In the part of machine learning, we need to know features and target. We make features to be the product of $s_i s_j$ for each grid and adding $mobility \times weight$. To regress all the data, we use *KNeighborRegressor* model in scikit-learn module. The cross validation is set to 0.2, and $k(neighbors) = 4$. The trained model outputs into a pickle file. However, *KNeighborRegressor* cannot calculate feature importance of the model.

During the game, on the black's turn, the algorithm import the model and return a target, or to say a predicted Hamiltonian. This would help the black side to decide which spot to place his piece.



x				y
S_{00}	S_{01}	...	S_{77}	H_1
S_{00}	S_{01}	...	S_{77}	H_2
		⋮		

FIG. 2. Feature and target.

2. *Minimax Algorithm*

This is the last step of the game algorithm. The key idea of minimax search is to run through the opponent's possible move, and how deep it predicts will affect the result of the algorithm. *Depth* refers to the distance. For instance, a depth of 3 for black player means that the black places a piece, the white places another, and then the black places the last one. The decision it makes is to take steps of the minimum or maximum value in the end. The decision tree is similar to Fig. 3.

According to our previous works, the black would seek for the node with the smallest value, while white player will seek for the node with the greatest value.

C. Mock Opponent

To evaluate the quality of our model, we introduce two kind of Greedybot who take monotonously steps to make

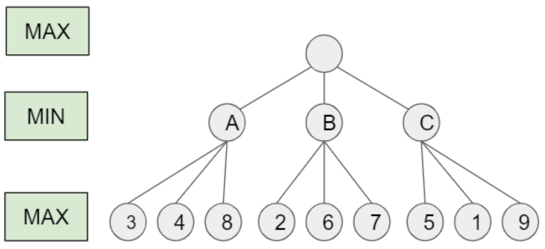


FIG. 3. The decision tree.

some greedy decision.

The first one, labeled as Greedybot(1), is taking the step that make the number of flipping is the most. The second one, labeled as Greedybot(2), is taking the step that make the opponent have the fewest places to draw, which is reducing the mobility of opponent.

By experimenting some mock matches, we will show the performance of our model.

III. RESULTS

A. Match Results and the Value of Hamiltonian

We made some mock tests between our model (IMBA_ML Model) and Greedybot. The results include the winner and the value of Hamiltonian each round along whole match.

The controllable parameters for IMBA_ML Model are *mobility* and *Depth*. We will see how these parameters can affect the performance in a match.

1. *IMBA_ML Model vs Greedybot(1)*

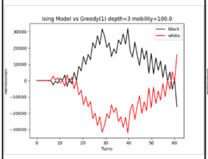
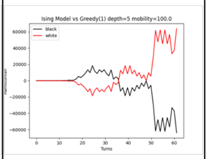
In Fig. 4, the black is presented in black line, and the white is presented in red line.

We can see obvious value change in certain turns, and that is where one player gets a corner. Since we the number is relatively large in the corners, the order of magnitude would be up to from three to five. On the other hand, to consider the contribution of mobility, is would be relatively smaller than the interaction between each piece. And, what's more, since greedy bots has fixed strategy in the way to play, the curves of testing under the same depth is quite similar to one another. But if we look into the values, there is sightly difference due to different *mobility* weight.

2. *IMBA_ML Model vs Greedybot(2)*

Since the mobility doesn't make the huge difference to the plot, we only compared the influence of the depth.

From Fig. 5, firstly, we can see that it's pretty much different with Fig. 4. And secondly, our model can take

mobility = 100		
Depth	3	5
Winner	IMBA_ML	IMBA_ML
Hamiltonian Plot		

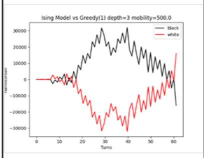
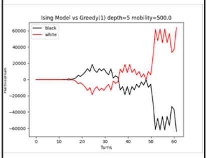
mobility = 500		
Depth	3	5
Winner	IMBA_ML	IMBA_ML
Hamiltonian Plot		

FIG. 4. The value of Hamiltonian plots of IMBA_ML vs Greedybot(1) varying with depth and mobility.

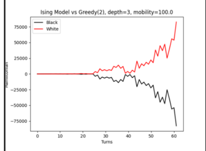
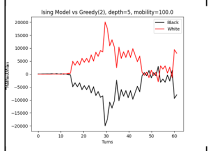
mobility = 100		
Depth	3	5
Winner	IMBA_ML	IMBA_ML
Hamiltonian Plot		

FIG. 5. The value of Hamiltonian plots of IMBA_ML vs Greedybot(2) varying with depth.

advantage of searching that would make it has more stability to win the match.

B. Conclusion

Our model can beat the mock opponent in the cases of different mobility and depth.

And we also showed that at the end of the match, the winner would have the higher stability, for the black is lower value of Hamiltonian, which is satisfied our Ansatz. However, in some case, for one side have higher stability in early moves, there still have some chance that the other would take over the pieces and make win the game with higher stability in the end.

IV. OUTLOOK

By now, we are using the fixed value of mobility weighting and it doesn't make too much contribution to the result. We may use some other machine learning method to realize the dynamic weighting through the match going.

In minimax search, we may use the Numba to speed up the calculating and alpha-beta pruning to reduce the amount of calculating. By speeding up, we can apply more depth and predict much more future steps.

Still, there are many things that can improve the algorithm and the method of machine learning.

To sum up, by now, we built a model that can beat the mock opponent, we hope that, by one day, we can beat the top human player through the optimization of the program and better algorithm.

Appendix A: Contribution

Jiang	Theory Research, Othello, Program Maintenance
Wan	Machine Learning, Minimax Algorithm, Model Testing

TABLE I. Team members' main contributions.

[1] K. Hashimoto, N. Iizuka, and S. Sugishita, Phys. Rev. D **96**, 126001 (2017).
[2] K.-P. Chan, 2013 othello world cup game records.