CS5100 Final Project Othello Game

Xiang Fan
Vaishnavi Hire
Northeastern University
fan.x@husky.neu.edu, hire.v@husky.neu.edu

Abstract

Game playing, as one of the most challenging fields of artificial intelligence has received a lot of attention.
This paper implements and examines different heuristics, in an attempt to make observations about the interplay between the heuristics, and how well each heuristic contributes as a whole.

Introduction

Othello, the game is mostly attractive because of the simplicity of its rules. One may start playing after a minute of introduction, and still the game has enough complexity to leave a dedicated person with years to master.

Game playing, as one of the well-admired components of artificial intelligence research, has captured tremendous amounts of attention. Computers, looking ahead beyond the next move and judiciously rationing out the next best move, mimic human intelligence, and at times, better than it. Computers have, to a good extent, captured the essence of game playing along with its intricate complexities. The importance of game playing arises out of the competition that exists between the human race and machines.

Most game-playing programs use evaluation functions to estimate the players' winning chances at the leaves of game-trees. Normally, these functions combine features that measure specific properties of the position which are correlated with winning. We implemented the game with Alpha-Beta pruning algorithm and combined with a strong evaluation function. This combination was able to deliver human level performance and we will evaluate our program later in the result section.

Related Works

In artificial intelligence, game playing has become a well-admired and well-explored domain as it demonstrates the capability of computers to look ahead beyond the next move and judiciously play the next best move, that is, mimic.

The use of Othello to determine and demonstrate a computer`s game playing capabilities is not new. The game of Othello was introduced in 1975.
Othello`s rules severely constrain the next possible moves, hence resulting in a very low branching factor, making it feasible for a computer search.

Othello game playing strategies can be improved using following approaches-
i) improvements in search strategies used
ii) form better heuristic functions
iii) use learning methods and train computer against massive data sets.

Research into Othello's game playing strategies was started by Rosenbloom in 1982 which essentially used the alpha-beta algorithm [1]. Modifications in minimax search also resulted in a comprehensive strategy to play Othello [2]. Later research of the game playing strategies include machine learning techniques. Methods that used a neural network to

evaluate Othello boards using temporal learning were also researched[3]. [5] uses Bayesian learning techniques for the feature combinations .[6] uses supervised learning to improve minimax searching. Advanced research work also focused on using genetic algorithms to evolve neural networks to assess Othello board configurations [4]. Genetic algorithms are also used to create evaluation function to determine which nodes to expand[7].

Although there is a lot of prior work in developing the heuristics for the Othello game, interaction between these heuristics to optimize Othello moves is unclear. In our project, we try to implement and examine various heuristics, their interaction with each other and their combined contribution towards the AI agent's decision to play the next move.Identifying heuristics that contribute immensely to Othello game-play implies that more processor cycles could be allocated in that direction to enhance the quality of play. We tried to cover all possible aspects of the game while developing our heuristics. The features we consider include difference in colored discs, ability of the discs to have maximum valid moves i.e mobility, heuristics for corners and stability of a disc. Our work concentrates on identifying high-yielding heuristic components and analyzing their behavior. To aid us in this process of analysis, various experiments were conducted.

Approach

Our Othello game's feature fall into the following cases:

1. Minimax Tree Search with Alpha-Beta Pruning
2. Evaluation function
    a. Color difference
    b. Mobility
    c. Stability
    d. Corner

Search Heuristic

Assume for the time being that we have an evaluation function at our disposal. We then can use a simple technique called "MiniMax Search" to look ahead. MiniMax search is called thusly because when we try to determine the best move, we always consider the best moves of each player. At some point, we have to stop looking ahead - simply because even in a simple game like Reversi, there are far too many possibilities to consider. We just can't look ahead from the first move until the end of the game. Therefore, we need to limit the search depth of our look-ahead. When we have reached the last level, we can't look beyond it, we use the evaluation function to find out whether the position we arrived at is good.

Alpha-Beta search is a refinement of MiniMax search that allows us to prune the game tree: if we find that it isn't worth continuing examining moves in a certain position, we backtrack immediately.

Evaluation Function

a. Color difference

This is the basic evaluation function we implemented at the very beginning. It just use the number of pieces of the color players play on the board as an indicator for the quality of a position. This greedy evaluation function is derived from the final goal of the game: the player who has the most pieces wins. But this only makes sense if you really can look ahead until the end of the game - which is only feasible in the last 10 to 18 moves (depending on the performance of your computer). If you can't do that, this greedy function will mislead your program. If you have a lot of pieces in any given position except at the end, this doesn't mean anything: a few moves by your opponent, and in the end, the situation may well be totally reversed. So we only use it as a basic evaluation function and used to compare with our improved heuristics later.

b. Mobility

A better criterion for determining a position is to count the number of moves a player can make, the mobility play an important role in Othello. The strategy is to restrict your opponent's mobility and to mobilize yourself. If you only have a few moves, it's likely that your opponent can play such that you are forced to make bad moves. Therefore, if your mobility is high, the position is better for you than if it's low. Mobility is calculated by examining the board and counting the number of legal moves for the player.

c. Stability

Stability of coins is another key factor in Othello. The stability measure of a coin is a quantitative representation of how vulnerable it is to being flanked. Stable coins are coins which cannot be flanked at any point of time in the game from the given state. Unstable coins are those that could be flanked in the very next move. Corners are always stable in nature, and as you build upon corners, more coins become stable in the region. We assigned different weights for stable and unstable coins.

d. Corner

Corners are the four squares a1, a8, h1, and h8. The specialty of these squares is that once captured, they cannot be flanked by the opponent. They also allow a player to build coins around them and provide stability to the player's coins in the environment. Capturing these corners would ensure stability in the region, and stability is what determines the final outcome to quite a large extent. There is a high correlation between the number of corners captured by a player and the player winning the game. Of course, it is not true that capturing a majority of the corners would lead to victory, since that clearly need not hold. But capturing a majority of the corners, allows for greater stability to be built. We assigned a very high weight for corner and our program will always pick up a corner if it's available.

Total Score

To calculate the total score of a position we used a linear function to add up the scores we got from the four parts above.

Total Score = a*color_difference + b*mobility_score + c*stability_score + d*corner_score

Evaluation

To evaluate our program, the results of the following scenarios are noted and compared for evaluation.

- Our program play against itself but with different heuristic functions.
- Our program play against online heuristic based Othello game[8].

Results

We have several comparison groups for the first scenario, Basic AI vs. Mobility AI, Basic AI vs. Final AI and Mobility AI vs. Final AI. Basic AI is only implemented with the color_difference evaluation function, Mobility AI is implemented with mobility function based on the basic AI, Final AI is the final version of our agent and it's implemented with everything we have. Each group played against another heuristic for 50 games and we recorded the results as shown in figure 1. Our Basic AI can't get 1 win against our Final AI, and our Mobility AI can only get 3 wins against our Final AI.

For the second scenario, we still have 3 comparison groups, Basic AI vs. online Othello, Mobility AI vs. online Othello and Final AI vs. online Othello. Each group played 50 games, and we recorded the result as shown in figure 2. Our Basic AI can not win a single game against the online Othello. The Mobility AI is doing slightly better and got 6 wins against the online Othello. Our Final AI is definitely improved a lot and got 45 wins against the online Othello. This proves that we successfully implemented the Othello game and we were making progress during the development of this game.

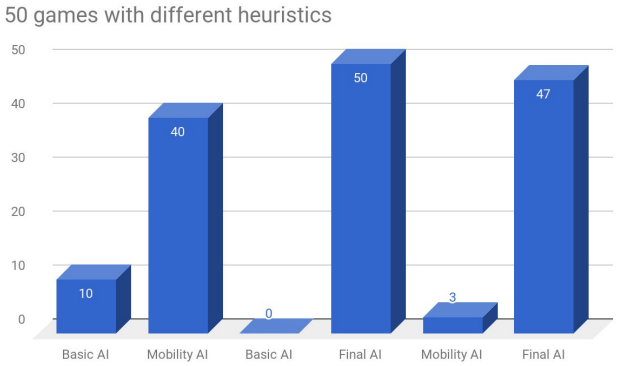

figure 1

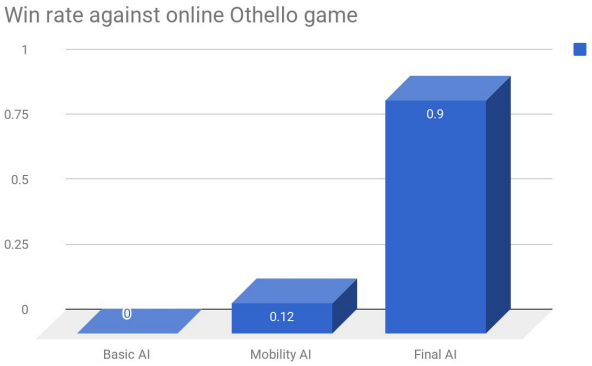

figure 2

Acknowledgments

We would like to thank Prof. Stacy Marsella for his guidance on the topics of Search algorithms and heuristic evaluations. All the code for the game was written by us. The Tkinter module of Python was used for frontend of Othello. Implementation of heuristics, research on related works and report was done by Xiang Fan. Implementation of game environment, alpha-beta pruning and report was done by Vaishnavi Hire.

References

[1] P. Rosenbloom,"*World-championship-level Othello program"*, Artificial Intelligence, Vol. 19, No. 3, 1982

[2] Campbell, Murray S. and Marsland T.A. *A Comparison of Minimax Tree Search Algorithms Artificial Intelligence 20*

[3] A. Leouski, *Learning of Position Value in the Game of Othello,* Masters Dissertation, Department of Computer Science, University of Massachusetts, 1995.

[4] D. E. Moriarty and R. Miikkulainen, *"Discovering complex Othello strategies through evolutionary neural networks",* Connection Science, Vol. 7

[5] Lee, K.; Mahajan, S.: *The Development of a World Class Othello Program, Artificial Intelligence*

[6] Buro, M., *Improving heuristic minimax search by supervised learning, Artificial Intelligence*

[7] A. Benbassat, Elyasaf, A. and M. Sipper, *"More or less? two approaches to evolving game-playing strategies",* Genetic Programming Theory and Practice X, 2013.

[8] Online Othello Game, http://www.ultraboardgames.com/othello/game.php