PFP Project Proposal: Othello and Minimax

Bryanna Geiger: bg2603

November 24, 2020

Abstract

My final project proposal for Parallel Functional Programming is to implement the board game, Othello, with the Minimax algorithm and potentially alpha-beta pruning in Haskell. Othello is a modern day board game involving two players trying to optimize their ”points” or the number of tiles of their color by the end of the game. The Minimax algorithm utilizes a tree structure and is a backtracking algorithm commonly used in games such as Othello, 2048, Chess, Checkers, and others. Alpha-beta pruning would help to optimize the efficiency of the implemented minimax algorithm. I am hoping to draw from my experience implementing 2048 with the minimax algorithm and alpha-beta pruning in Python to translate that to the board game Othello in Haskell.

1 What is Othello?

Othello is a board game derived from the original game Reversi. The game board itself is an 8x8 square grid. Each player has 32 disk-like black and white pieces - each playing choosing a color to start the game. The initial starting position includes 4 pieces in the middle 4 squares of the board - 2 black and 2 white pieces.

The aim of the game is to end with more pieces of your color on the board than the opposing player. Player alternate turns and when a valid move is not possible, that player’s turn is skipped for the current round. A player may not voluntarity skip their turn when a valid move exists. When a single disk or a row of disks is surrounded at the ends by the opposing color, the surrounded disks will be ”flipped” to the opposite color, which is how each player earns points.

The game ends when there are no valid moves remaining for either player or both players are out of disks, thus not being able to make a valid move. The player with more disks of their color wins. If there is an even amount of both color, the game will end in a tie.

2 What is Minimax?

The general idea is that minimax is a backtracking algorithm which is a recursive algorithm. As a searching algorithm, minimax is commonly used in games such as 2048, chess, othello, checkers, and others. It utilizes a tree structure in order to determine the optimal move for each player to make.

In this case, there are two players: "maximizer" and the "minimizer" where each player will either try to maximize the value of a move while the other will minimize the value of a move. Different sorts of heuristics can be implemented in order to adjust the values of the board.

Additionally, a potential expansion on this idea could be to also implement alpha beta pruning. This technique can help to optimize the efficiency of the minimax algorithm by reducing the number of paths or nodes that the minimax algorithm would normally explore. Ideally, implementing alpha beta pruning in addition to just the pure minimax algorithm is ideal, but if time does not allow, just implementing minimax for the Othello game also works.

3 Why Am I Interested?

Othello is one of the board games that I grew up playing with my family so I think it would be an engaging project to be able to implement this particular board game. Additionally, when I took COMS 4701 Artificial Intelligence, we had implemented the 2048 game with minimax algorithm and alpha beta pruning. Being that I've implemented the minimax algorithm before in Python, I should have a comfortable understanding of how the algorithm itself works. While Othello and 2048 are two different games, having implemented 2048 and these concepts in Python, making the jump to Othello seems to be within the scope of this project. I am interested to see the implementation of Othello in Haskell as opposed to in Python, as well as how 2048 and Othello differ in implementation, especially keeping in mind how parallel programming can impact the performance of this game.

Figure 1: An example of the Othello board game