

Applications of Artificial Intelligence and Machine Learning in Othello

Jack Chen

TJHSST Computer Systems Lab 2009-2010

Abstract

The purpose of this project is to explore Artificial Intelligence techniques in the board game Othello. The project focuses on the creation and evaluation of AI players for Othello. It explores several techniques used in strong AI players, including improvements to minimax game-tree search algorithms and higher-quality evaluation functions. It also investigates machine learning methods to enable AI players to improve the quality and speed of play automatically based on training and experience.

Introduction and Background

The primary aspect of most AI players is the search algorithm, which is used to evaluate a board state based on a prediction of future moves from that state. The standard basic game-tree search algorithm is minimax with alpha-beta pruning. I plan to implement several improvements on minimax search. NegaScout and MTD(f) are two minimax search algorithms that can reduce the number of nodes that must be searched compared to alpha-beta pruning by searching with a null-window, where alpha and beta are almost equal. Null-window searches produce many more cutoffs than alpha-beta pruning.

An important way to improve search speed is to cache information about board states that have already been evaluated in a transposition table, which allows the player to avoid repeated searches. Selective search algorithms can further enhance game-tree search by pruning parts of the game tree that probably will not affect the overall minimax value. This allows the player to search much deeper in the relevant parts of the game tree. I will also investigate other search techniques, such as quiescence search.

I will also investigate opening books, which allow much better and faster play in the early game by storing previously computed information about early game board states. I will explore parallelization of the search algorithms, as well as the machine learning algorithms used to train the AI.

Development

The first part of this project was the implementation of an Othello referee program to run the game. The referee keeps track of the board state and allow two players to play games against each other. The referee supports AI player programs written in multiple languages as well as human players. I have also implemented a graphical user interface for the referee. The GUI displays the board, animates the players' moves, and allows a human to play easily by clicking on the board.

The primary focus of the project was on Othello AI players. I have investigated several AI techniques, including improved minimax search algorithms such as NegaScout and MTD(f).

I store boards as bitboards, which allow great speed improvements in certain operations, such as finding all possible moves or counting frontier squares, with bit manipulation techniques.

I have explored Machine Learning techniques for the optimization of evaluation function parameters. My evaluation function takes a linear combination of a set of features, including the number of pieces on different types of squares (in particular, on corners and squares adjacent to corners), the number of possible moves, frontier squares, and parity. The weights for these features are trained using a batch gradient descent method. The game is divided into one stage for each of the possible number of pieces on the board. A separate set of weights is trained for each stage, starting from the last stage. The target value of each board for the stage is determined with a minimax search based on the already trained weights for later stages. The weights are then converged to minimized error with a batch gradient descent algorithm using a modified line search.

The preliminary results are displayed in the chart below. Corners and x-squares are extremely important, especially early in the game, while mobility becomes increasingly important near the end of the game.

