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.

The board evaluation function is another important aspect of AI players. In Othello, evaluation functions are often based on several “complex” features, such as mobility and stability. However, using a collection of “simple” features, which evaluate patterns in a small number of disks, can improve board evaluation. I will investigate various board evaluation methods such as these.

I will explore the use of machine learning to train an evaluation function by automatically optimizing the relative feature weights. There are several machine learning techniques that can be applied to this problem, including stochastic gradient descent, genetic algorithms, and artificial neural networks. I will also explore other ways to enable an AI player to improve the quality and speed of play based on experience.

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 implemented basic AI players in both Python and C++, in order to compare their performance, but most of the AI player development is in C++. First, I wrote players using minimax and minimax with alpha-beta pruning. These are used as a baseline for comparison and for testing improved minimax search algorithms. I have implemented bitboard optimizations, which allow certain board operations, such as finding the legal moves and counting frontier squares, to be performed much faster. I have investigated several AI techniques, including improved minimax search algorithms such as NegaScout and MTD(f). I have also explored Machine Learning techniques for the optimization of evaluation function parameters.

Results and Conclusions