CS 182 Final Project

Shayn Lozano, Larry Zhang, Benjamin Zheng

October 2016

1 Introduction

While designing this final project, we were interested in tackling subproblems presented in the game of chess using the techniques we have learned in class. One particular subproblem that stood out to us was the question of whether an aggressive style of chess produces better results than a passive style of chess. We define an aggressive style of chess to be one in which the player is more predisposed to trading pieces regardless of whether there is a clear advantage gained, which contrasts with the passive style's aversion to trading pieces when there is no clear advantage being gained. In doing this, we hope to gain perspective on which style works better for artificial intelligence, and delve into why that is the case.

In addition to the above, we would like to take a look at end game situations, which have a significantly smaller state space and a much clearer evaluation function, allowing algorithms learned in this class to be better applied and to be run to deeper depths to these situations. Specifically, we would like to examine situations where there are only pawns and kings left on the board, as they situations can become complicated due to factors like board positioning sometimes being much more important than the value of the pieces on the board due to a lack of mobility, the 'winning' player's general aversion to drawing, and navigating the state space to produce checkmate if a pawn is promoted.

2 Algorithms

We intend to approach the first problem by applying the genetic algorithm presented in class instead of the more traditional search algorithms. Much of the reasoning behind this decision can be found in our first cited paper, but genetic algorithms will allow us to theoretically generate high-quality solutions by doing an initial search for solutions, "growing" high-quality solutions, and then searching for even better solutions through mimicking processes like mutation and crossover. Because we already limit our search to sets of moves that follow an aggressive or passive paradigm and we are looking at the chess board at large, genetic algorithms lend themselves very well to these limited situations

that allow multiple areas of play on the board (as there can be many avenues of attack, for example).

In solving the pawn-endgame problem, we intend to use alpha-beta pruning, focusing our attention on building a complete and optimal algorithm that can take all the variables described in the introduction and identify the best solution to the situation inputted into the chess program we have constructed *as quickly as possible*. This will involve the design of clever heuristics to help direct the AI agent in the right direction without taking too much time exploring the state space, and the ability to identify situations when a draw from repetition is invevitable based on optimal play from both parties.

In both of the above, our algorithms will assume the opponent uses optimal play by default. Should we have extra time at the end, we may add functionality allowing us to input other policies for the opponent or attempt to learn the opponent's policy.

3 Expected Behavior

With the pawn end-game problem, a proper implementation of a solution will be able to take any game board and, if a complete solution exists given the opponent's policy and defaulting to optimal play, identify the winner and the moves it will take until checkmate. If no complete solution exists (i.e. there could be a draw, win, or loss depending on the opponent's moves), the algorithm will return the optimal moves and the outcome they lead to.

The aggressive vs passive policy problem should yield more interesting results, although it is a less tractable problem. It is hard to predict exactly what will happen, as one could reasonably say trading aggressively will shrink the state space and allow for the ability to delve deeper into the possible states, whereas one might also say the aggressive agent will be prone to making trades that compromise position or other intangibles, and thus will be worse than the passive agent.

4 Issues

Many of the issues we expect to focus on have been described above. For the aggressive vs. passive policy problem, outside of designing the algorithms to play as optimally as possible within these policies, we will face the task of assessing which style of play is objectively better. For the pawn end-game solver, some of the issues we will need to account for include dealing with draws by repetitions, the promotion of pieces (sometimes a piece other than a queen will be advantageous), the change need to account for board positioning sometimes being preferred to raw point value on the board, and the need to transition all of these factors as pieces become promoted on the board.

5 Resources

While we were researching, we found several resources that will be helpful in completing our project. We list them below, with the top article being the most helpful.

https://www.cs.purdue.edu/homes/dgleich/publications/Gleich%202003%20-%20Machine%20Learning%20in%20Computer%20Chess.pdf

http://www.genetic-programming.org/hc2014/David-Paper.pdf

https://www.uio.no/studier/emner/matnat/ifi/INF4130/h12/undervisningsmateriale/chess-algorithms-theory-and-practice-ver2012.pdf

http://www.cs.cornell.edu/boom/2004sp/projectarch/chess/algorithms.html