Brandon Palomino
CS 420

4 in a line Project Analysis

Approach

The 4 in a line project was essentially a bigger version of tic tac toe from my point of view. This viewpoint also plays a part in how I implemented my version of the alpha beta pruning algorithm and the evaluation function along with it. The project turned out to be a small but enjoyable game of big tic tac toe against a CPU which uses a-b pruning to estimate what the next best move will be for itself throughout the game play.

My game starts off by prompting who will go first and how long the CPU has to determine its moves with a timer applied on the a-b pruning algorithm. After the initial settings the game starts with the user selecting their plays and the CPU playing back and this goes back and forth until a 4 in a line is achieved.

The frame of the game is straightforward using lists, strings, and integers to pass information around the game and core components to the program is the alpha-beta pruning algorithm and evaluation function that works alongside it. My implementation uses recursion to recursively call itself to generate "scores" which are evaluations of a board state. It first starts with an initial board state and generates all possible successors of that board to calculate a score for. These scores update the alpha and beta values which are passed along, and this finishes off when either it has a calculated a terminal node or reaches the time limit provided initially and returns the calculated nonterminal. The most important part of this algorithm is the pruning that happens when it checks the alpha and beta values throughout the loop of successors if the condition where the alpha value is greater than or equal to the beta value then it prunes that branch and continues to the next successor. This achieves great optimization with generation the best possible move and memory efficiency. My evaluation function works to go throughout the board checking each row and column for a specific patterns. For 4 in a row plus 1000 points for the CPU minus 1000 for the other player, 3 in a row plus 100 for the CPU and minus 100 for other player, and so on until 1 in a row with 3 free spaces. The entire board is evaluated for these patterns in the row and columns adding a subtracting points based on the situation the players are in.

Analysis and Findings

In terms of time the algorithm runs smoothly as it runs the best alpha value it can achieve for itself in the time limit it has. And it has maximum depth of 4 so it does not struggle to much to calculate a good decision. I found that at depth 5 and higher it took longer than 30 seconds for the algorithm to actually achieve a terminal node. Also that when introducing killer moves into game would prove to be very challenging for CPU to make a good play against. Nonetheless it plays adequately in all games.