Clay Coleman

Eric Fortney

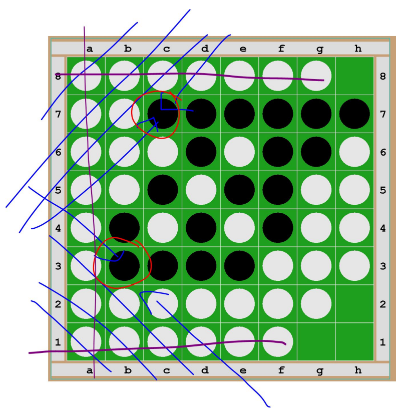
CS 470

Reversi Final Lab Report

The first implementation of this lab explored the trees of possible moves from a given state determining the best via Alpha-Beta Pruning (ABP). We assigned each move a weight given a heuristic function which factored in the following features: total number of tokens gained, the set weights of the board positions, if the move would lead to an early win/loss by consuming all opponent tokens (aggression/self-preservation), and the number of 'safe tokens' the move would generate. Following ABP, the full tree of potential moves is progressively explored with each node trying to maximize the score of the parent's opponent. With our heuristic evaluation function we can generate upper and lower boards for scores (Alpha and Beta) to consider and prune off entire branches of the search tree as the heuristic function determines them redundant.

1	5	3	3	3	3	5	1
5	5	4	4	4	4	5	5
3	4	2	2	2	2	4	3
3	4	2			2	4	3
3	4	2			2	4	3
3	4	2	2	2	2	4	3
5	5	4	4	4	4	5	5
1	5	3	3	3	3	5	1

The various positions of the board are partitioned and assigned values in a method analogous to the image to the left. Through trial and error, we determined appropriate weights for each space. In this illustration, lower number are considered more favorable over higher numbers. The initial AI had a continuous light heuristic weight to maximize the total number of tokens. If a player possessed a corner it opened up the possibility to create a region of unflippable tokens. When two adjacent corners were controlled broad bands of safe space could be created between the corners progressing inward. When just one corner was controlled, 45-45-90 triangles of safe spaces could be seized stretching inward from the corner. A secondary algorithm would identify these safe-spaces by progressively tracing lines of identical tokens until an opponent token is found, a visualization of said algorithm is seen to the lower right.



All of the previous features remained in place in our final iteration of the AI although many optimizations and tweaks were made. First of all, dynamic depth was added. Due to the exponential explosion of the search space towards the middle of the game, only a relatively modest depth of 6 nodes could be reached on our machines before we encountered series bottlenecks in performance. As we enter the more consequential moves of the endgame though, the breadth of each level of the tree is considerably smaller, allowing us to consider a larger depth of nodes. Simply put, as the endgame approaches we progressively raise the depth of the ABP algorithm. In our experience we could comfortably model to the end of the game from 52 tokens onward. We applied similar logic to how the game should optimize for maximizing tokens. The most successful players tend to avoid moves that net a large number of tokens towards the start of the game as they generally open up several avenues of play for the opponent. In our updated algorithm we disincentivize capturing general token positions until the 'late game' is reached (36 total tokens onward). Finally, in the same vein as our 'guaranteed safe positions' we implemented a small algorithm to find if a move provides the player all tokens along an edge, a generally 'safe' and favorable arrangement on the board.