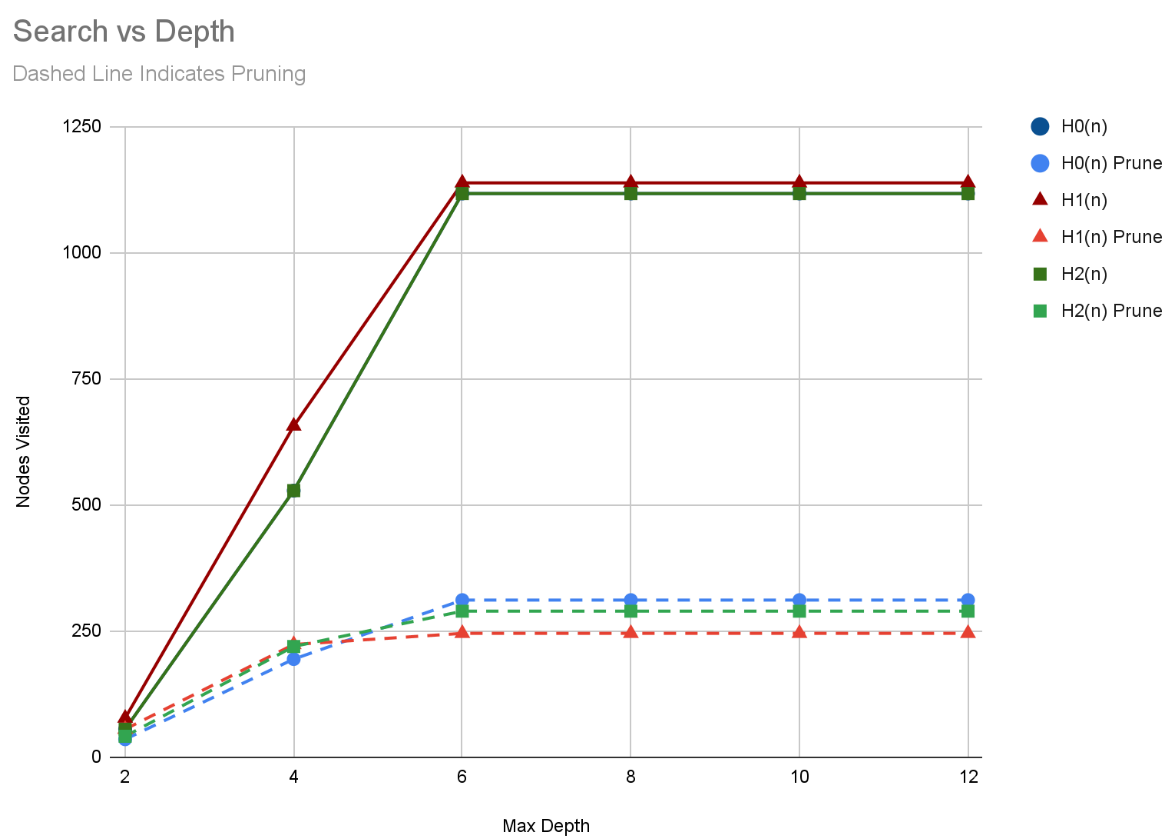
James Smith
Joshua Tumlinson

Othello -- CS 331H

To start, a brief description of the heuristic functions will prove useful:

H0(n)	Number of your pieces - number of opponents pieces
H1(n)	Number of your legal move - number of opponents legal moves
H2(n)	Number of your pieces - number of opponents pieces, +10 for every corner you can legally play or have a piece placed

Search vs Depth:



Heuristic Quality:

4x4 Grid	Max Depth			
Heuristics	2	4	6	8
H0(O) vs H1(X)	P2 Wins	P2 Wins	P2 Wins	P2 Wins
H0(O) vs H2(X)	P2 Wins	P2 Wins	P2 Wins	P2 Wins
H1(O) vs H0(X)	P2 Wins	P2 Wins	P2 Wins	P2 Wins
H1(O) vs H2(X)	P2 Wins	P2 Wins	P2 Wins	P2 Wins
H2(O) vs H0(X)	P2 Wins	P2 Wins	P2 Wins	P2 Wins
H2(O) vs H1(X)	P2 Wins	P2 Wins	P2 Wins	P2 Wins

6x6 Grid	Max Depth				Statistics	
Heuristics	2	4	6	8	P1 Win %	P1 Heuristic Avg
H0(O) vs H1(X)	P2 Wins	P2 Wins	P1 Wins	P1 Wins	50%	50%
H0(O) vs H2(X)	P2 Wins	P2 Wins	P1 Wins	P1 Wins	50%	
H1(O) vs H0(X)	P1 Wins	P1 Wins	P1 Wins	P2 Wins	75%	75%
H1(O) vs H2(X)	P1 Wins	P1 Wins	P1 Wins	P2 Wins	75%	
H2(O) vs H0(X)	P1 Wins	P1 Wins	P1 Wins	P1 Wins	100%	75%
H2(O) vs H1(X)	P2 Wins	P2 Wins	P1 Wins	P1 Wins	50%	

Heuristic Performance:

Looking at our *Search vs Depth* graph, we can see a very clear pattern: pruning substantially decreases the number of nodes visited, thus decreasing the time it takes to run the algorithm. One thing that might be hard to see in the graph is the line for H0(n). When no pruning is applied, H0(n) and H2(n) result in an identical number of nodes visited, as they share the same underlying principle.

We can also observe that both H0(n) and H2(n) visit less nodes than H1(n) without pruning. This is the exact opposite when we incorporate pruning, as both H0(n) and H2(n) visit more than H1(n), with H0(n) visiting slightly more than H2(n), showing that H2(n), being based on the same underlying principle, is an improvement upon H0(n).

It's also important to point out that H2(n) is actually worse compared to H0(n) when max depth is set to the lower numbers (2 and 4). H1(n) is also worse than both other heuristics here, showing that without a substantial amount of foresite, these heuristics don't perform to the best of their ability.

We also see a very interesting occurrence when we reach max depths of 6, 8, 10 and 12. The number of nodes visited do not change between each of these depths. With a 4x4 grid, there are only so many moves that can be made, thus only so far down a path you can go. In the end, these max depths result in the same or similar games being played, resulting in the same paths being visited, leading to the same number of nodes being visited.

A note that is far less exciting is the quality of each heuristic in a 4x4 grid. With a 4x4 grid, player 2 has a substantial advantage over player 1, causing player 2 to win in one hundred percent of matches, regardless of the heuristic it uses or the max depth it can visit. When we expand the grid to 6x6, we see that player 1 is now able to win as well as some heuristics performing better than others. H0(n) is the worst model, winning only 50% of the time for player 1. H2(n) is better, winning 100% of its matches as player 1 against H0(n), while only winning 50% against H1(n). H1(n) performed the best, winning 75% of its matches as player 1 against both other heuristics, meaning it is a more stable and consistent heuristic compared to H2(n).