Robert Brajcich
Damon George
De Palma
CPSC 427: Artificial Intelligence

Final Project - Othello AI

Overview

In our final project we implemented the alpha-beta pruning algorithm, along with a complex heuristic to determine the value of different board states. The final demonstration uses a depth limit of 7 in order to balance timing requirements with the highest depth limit achievable. The alpha-beta pruning code can be seen in the file **OthelloAI.java**, in the function alphaBetaSearch. Furthermore, we implemented a heuristic value hashing algorithm so that the heuristic for any given board state only needed to be calculated once and could then be saved and recalled quickly using a generated hash of the associated board state.

Heuristic

The heuristic function we used involved a weighting of seven different characteristics about the state of the board which will be described in detail below. These were as follows:

Heuristic Characteristic	Associated Weight
Count	0.05
Actual Mobility	0.15
Stable	0.3
Unstable	-0.1
Semi-Stable	0.1
Corners	0.3
Potential Corners	0.2

Each of these quantities is associated with a countable number of pieces that fulfill each characteristic. The count is determined for both the player and his opponent. The value of each characteristic (which will then be multiplied by the associated weight) is then computed as a weighted difference in the form:

(# for player - # for opponent) / (# for player + # for opponent)

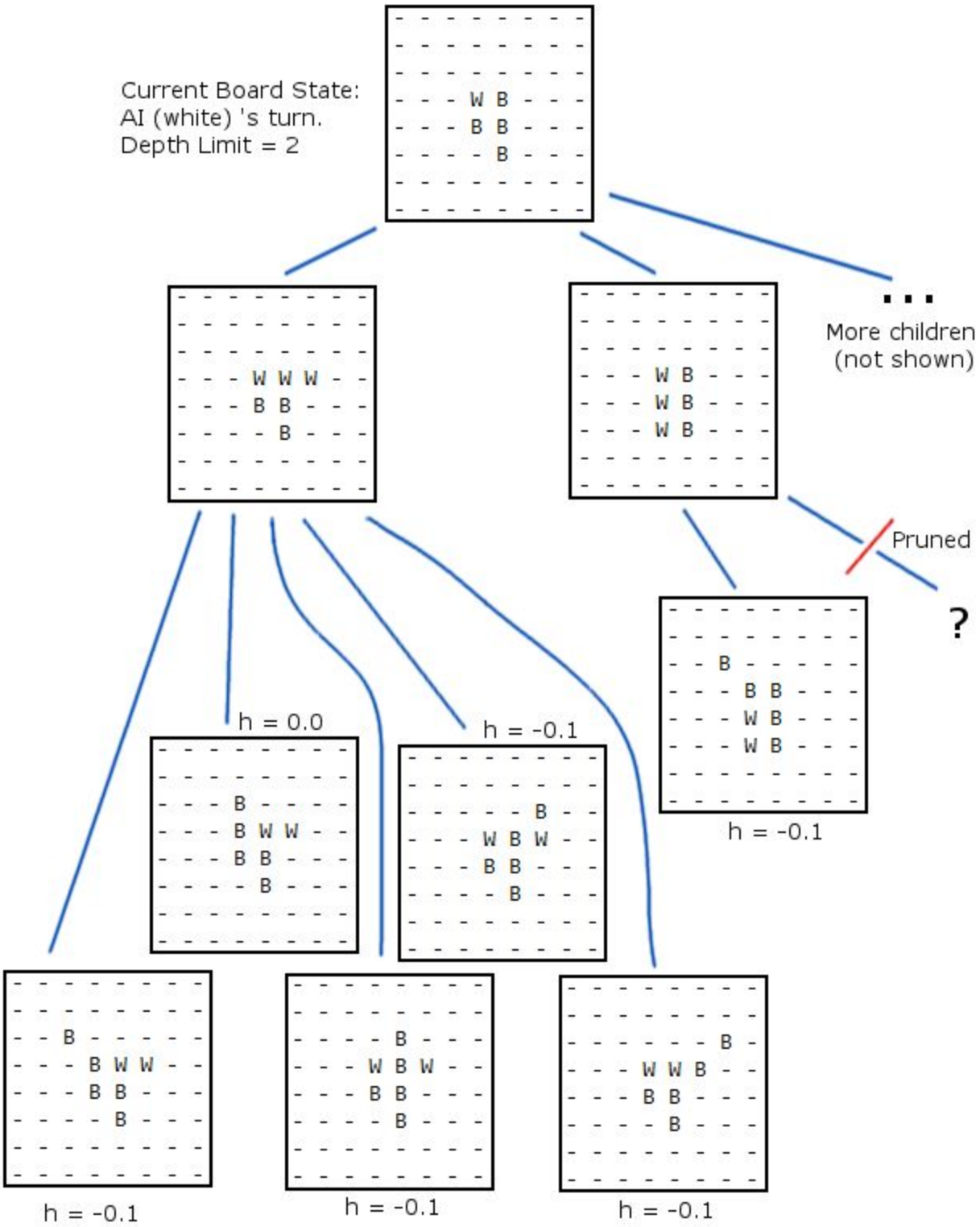
The count characteristic is a simple count of the total number of pieces each player has on the board. The actual mobility is the number of legal moves available to each player in the given board state. The stable, semi-stable, and unstable characteristics calculate the number of

pieces that can be categorized in each of these states. A stable piece is one that can never be flipped from its current value. A semi-stable piece is one that is not currently flippable but could possibly be flipped in the future. An unstable piece is one that could be flipped on the next move by the opposing player. The corners characteristic counts the number of pieces each player has in one of the corners of the board. Finally, the potential corners characteristic counts the number of corners that each player could take on their next move.

One final note is that for a board state determined to be a leaf node, the heuristic is instead calculated as (player score - opponent score) * 1000 in order to ensure a winning or losing game state has a heuristic of very high magnitude compared with other heuristic assessments.

Alpha-Beta Example

The example on the next page uses actual board states and heuristic values generated while running our program, but uses a depth limit of only 2 in order to demonstrate the process without overcomplicating the algorithm.


Current Board State:
AI (white) 's turn.
Depth Limit = 2
More children
(not shown)
Pruned
?
h = 0.0
h = -0.1
h = -0.1
h = -0.1
h = -0.1
h = -0.1