ECE457a Assignment 1

Lucas Wojciechowksi, Ariel Weingarten, Alexander Macguire, Austin Dobrik, Dane Carr

June 2, 2014

1 Question 1

1.1 Part A

The game board consists of 7 squares, indexed from 0 to 6.

We can define every state within the game as a 7-tuple containing

- 3 black coins, denoted ■
- 2 white coins, denoted □
- 2 emptys, denoted −

The inital state is $s_0 = (- - ■□■□■)$

Coins are moved as adjacent pairs. We define a pair of coins as p_i, where i is the lowest-indexed square on which the coins rest. In the initial state, the first pair of coins is p_2.

Empty squares on the board always exist as a single adjacent pair, which we will also define by its lowest-indexed square. On the initial game board, the empty pair is at p_0.

Lets define a function, empty_pair(s) which, given a state s, returns the empty pair on the board. For example. empty_pair$(s_0) = p_0$.
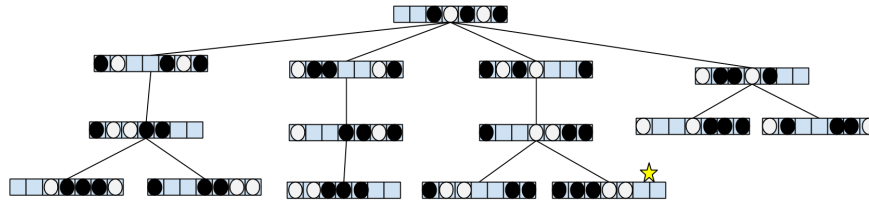
In any game state, s, the valid pairs are

$$\{p_i | i \in [0, 5] \backslash [\text{empty\_pair}(s) - 1, \text{empty\_pair}(s) + 1]\}$$

Each of these pairs corresponds with exactly one valid move
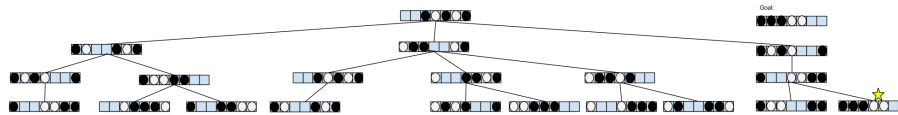
$$p_i \rightarrow \text{empty\_pair}(s)$$

1.2 Part B

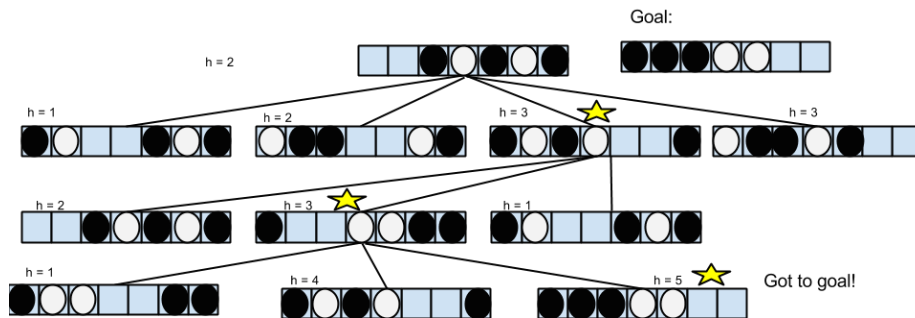### 1.2.1 Breadth First Search

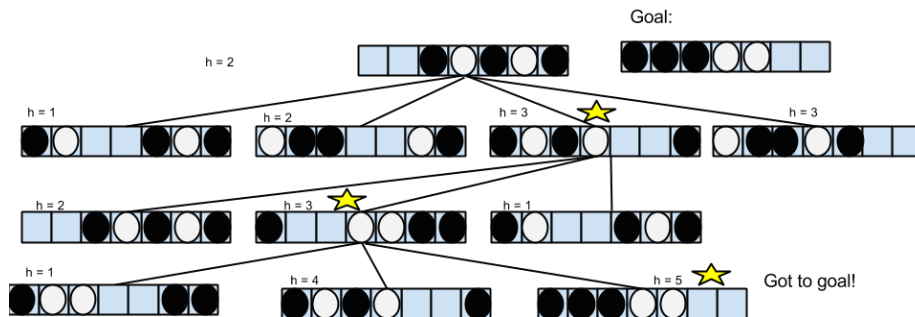### 1.2.2 Depth First Search (with depth limit of 3)

1.2.3 Hill Climbing Search

We used a simple and intuitive heuristic, where the value of each node is the number of squares that have the same coin as the goal state. For example, if the goal state is $s_f = (\blacksquare\blacksquare\blacksquare\square\square - -)$ then $s_0 = (- - \blacksquare\square\blacksquare\square\blacksquare)$ has value 2.

1.2.4 Best First Search (using heuristic developed in previous answer

1.2.5 Compare the performance of the above search strategies in terms of the number of nodes generated

Breadth-first search generated 15 nodes before finding the solution, depth-first search generated 20, and hill climbing and best first generated 11. Hill climbing and best first search followed the exact same path in this case, so their performance was the same. However it may be different with a different heuristic or a different initial state. Note that the depth limit of 3 on depth first search forced the search to generate more nodes. If it had been allowed to go to a depth of 4, depth-first search would have found the solution after only 6 nodes.

In this case, hill climbing and best first search are tied for best performance.

2 Question 2

2.1 Part A

The root node minimax value is 8.
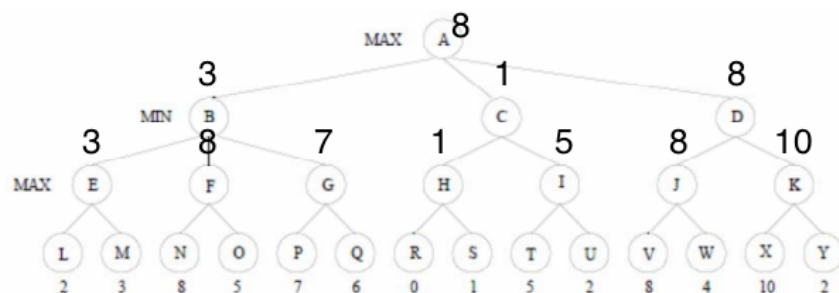
2.2 Part B

Max should choose node D.

2.3 Part C

Nodes O, Q, U, Y are not visited when α-β pruning is used.

2.4 Part D

Swap B and D, giving a final order of D C B on the second level of the tree. This results in ten nodes being pruned: Y, I, T, U, F, N, O, G, P, Q.