Preliminary Report

CMPT355 - Project 1 - Jayden Laturnus, Andy Yip, Graham Cooper

AB puzzle solver preliminary report in which the search algorithm, evalulation function, state space representation, and decided programming language are discussed.

Search Algorithm

After reviewing various heuristic searching algorithms our team opted to implement the A* graph-search algorithm. This was chosen largely because memory was not a constraining factor with regards to the complexity of the puzzle. Furthermore, A* is an optimally efficient algorithm given a consistent heuristic which we intend to use Manhatten Distance as our defining heuristic.

Evaluation Function

Some ideas our team came up with in regards to the evaluation function were:

1. Evaluate the next possible move, and simply choose the one with the least amount of cost associated with it.

2. Pre-play the puzzle, meaning search through and evaluate possible game states for a defined number of moves. Then simply choose the path leading with the least amount of cost associated with it.

State Space Representation

The state space representation of the puzzle problem will be represented with both the larger and smaller disks in an integer array. With a node represented in a struct that contains values used by A* search in order to find the optimal solution. The contents of said struct are outlined below:

- *initial state*
- *goal state*
- *heuristic (estimated cost from n to reach goal)*
- *goal state + heuristic (estimated total cost of path through n to goal)*
- *parent node*
- *children node(s)*
- *number of children*

Programming Language

C was the decided programming language to implement our AB Puzzle Solver. It was chosen for the purpose of speed, manual memory management, and was generally the preferred language amongst our team.