COMP30024 Project Part A Report

Zheping Liu, 683781
Bohan Yang, 814642

1 Formulation of the game as a search problem

The states of the game is structured as a dictionary with elements current positions of all pieces, previous action to get to this state, current positions of all blocks, previous state, colour of the pieces. The initial state is the constructed using the given file, with no previous action and previous state. The actions include JUMP, MOVE, and EXIT. The goal test is achieved by testing if all pieces are removed from the game board. As the final goal requires an extra step from the EXIT position, the sub goal is for any piece to get to any of the EXIT positions. The path cost for each action is simply assumed as 1 in this program.

2 Search Algorithm

The search algorithm used by the program is A*, with the modified Manhattan distance for hexagonal grids. This algorithm is chosen for the optimality and relatively short time consumption. The time complexity for A* search is exponential in (relative error in the heuristic x length of solution). This algorithm is also complete, unless there are infinitely many nodes with f ¡= f(G), which does not appear in this program. The space required for A* search is to keep all expanded nodes (closed list) and all nodes to be expanded (open list) in memory. The modified Manhattan distance heuristic is admissible as it is the sum of the distance from all pieces to their closest EXIT positions respectively. The states to be visited are stored in a priority

queue, which takes the shortest heuristic distance as their priority, the nodes with shorter heuristics will be opened (expanded) first.

3 Time and Space Complexity

The space complexity for this program is very high due to the structure of states in our program. States is a dictionary with one of the elements being previous state. Previous state is a nested attribute that stores all previous states before the current state, therefore, accumulates magnificently as more actions are taken. The time complexity of this program will increase a lot as the number of pieces increase in the game board. This is due to there are many possible successors can be generated at each state. Also, since our heuristic considers the total distance by summing up all shortest distances from every piece to sub-goal, many states will have same heuristics which makes the program will not always expand the optimal node.