1 Introduction

The problem : The 8 puzzle is a game invented by Sam Loyd in the 1870s. It is played on a 3-by-3 grid with 8 square tiles labeled 1 through 8 and a blank square. The goal is to rearrange the tiles so that they are in order. We are permitted to slide tiles horizontally or vertically (but not diagonally) into the blank square. Below is a sample initial state and the goal state that we will be following in this assignment.

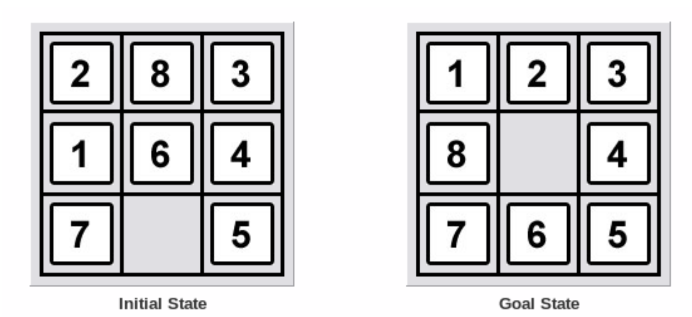

Figure 1: States for 8-puzzle problem

The 8-puzzle is a square board with 9 positions, filled by 8 numbered tiles and one gap. At any point, a tile adjacent to the gap can be moved into the gap, creating a new gap position. In other words the gap can be swapped with an adjacent (horizontally and vertically) tile. The objective in the game is to begin with an arbitrary configuration of tiles, and move them so as to get the numbered tiles arranged in ascending order either running around the perimeter of the board or ordered from left to right, with 1 in the top left-hand position.

1.1 Remarks

Johnson & Story (1879) used a parity argument to show that half of the starting positions for the 8-puzzle are impossible to resolve, no matter how many moves are made. Thus, given that a solution exists, our implementation will show the path to the goal state. In case the input is not *solvable*, our code goes into an infinite loop.

2 A* Algorithm

2.1 Heuristic selection

The A* algorithm is a modified Best First Search with the nodes in its priority queue ordered according to the increasing values of the function f which is describes as below :

$$f(n) = g(n) + h(n) \tag{1}$$

where g(n) is the depth function and h(n) is the chosen heuristic function. For the 8-puzzle problem, the A* algorithm may be applied by inserting the starting board position into a priority queue. Then, delete the board position with the minimum priority from the queue. Insert each neighboring board position onto the queue. Repeat this procedure until one of the board positions dequeued represents a winning configuration.

The success of this method hinges on the choice of priority function. A natural priority function for a given board position is the number of tiles in the wrong position plus the number of moves made to get to this board position. Intutively, board positions with low priority correspond to solutions near the target board position, and we prefer board positions that have been reached using a small number of moves.

Another choice of heuristic for this problem is **Manhattan distance**. The Manhattan distance between a tile and its desired position is the minimum number of moves it would take to move the tile to its desired position assuming you don't have to worry about other tiles. The Manhattan distance priority of a given board position is the sum of the Manhattan distances between tiles and their desired positions, plus the number of moves made to get to this board position. As before, if we solve the puzzle from a given board position on the queue, the total number of moves we need to make is at least its priority. This ensures that we find a solution to the 8 puzzle that uses the minimum number of moves.