CSCI 48700

Programming Assignment 1

Kelsen Hazelwood

Running the Code:

The code is in two different directories. One is for the breadth first search, and the other performs AStar. Both have individual makefiles and can be compiled and ran by the command “make run“ from their individual directories.

Breadth First Search:

This algorithm can take quite a while to run. The 8-tile problem has 9!8!7!6!5!4!3!2!1! possible configurations or 362,880. In an example run, the algorithm checks 151,619 nodes before coming to the solution. This is almost 42 % of possible board configurations. However, the solution that is found is optimal because the breadth search algorithm is checking all of the possibilities row-by-row so the solution that is found is at the lowest level that the solution exists. This may not be practical in applications where a quick solution is necessary, but preferred when you need the best solution and have the time to wait.

A\*:

The A\* algorithm runs much quicker. In an example run of this algorithm, it only checks 1211 nodes before finding a solution. However, as mentioned earlier, the solutions given by the A\* algorithm are not optimal as it does not check all the possibilities for every level, but jumps further down the tree until a solution is found so another solution could exist at a lower level. This is not useful in cases where the amount of steps taken to solve the problem is important such as looking for directions from one place to another.

Printed Results A\*:

287 Nodes

8 0 3

2 1 5

7 6 4

8 1 3

2 0 5

7 6 4

8 1 3

0 2 5

7 6 4

0 1 3

8 2 5

7 6 4

1 0 3

8 2 5

7 6 4

1 2 3

8 0 5

7 6 4

1 2 3

0 8 5

7 6 4

1 2 3

7 8 5

0 6 4

1 2 3

7 8 5

6 0 4

1 2 3

7 8 5

6 4 0

1 2 3

7 8 0

6 4 5

1 2 3

7 0 8

6 4 5

1 2 3

7 4 8

6 0 5

1 2 3

7 4 8

0 6 5

1 2 3

0 4 8

7 6 5

1 2 3

4 0 8

7 6 5

1 2 3

4 8 0

7 6 5

1 2 3

4 8 5

7 6 0

1 2 3

4 8 5

7 0 6

1 2 3

4 0 5

7 8 6

1 2 3

4 5 0

7 8 6

1 2 3

4 5 6

7 8 0

Printed Results Breadth First:

Nodes checked: 109921

1 4 7

6 5 2

8 3 0

1 4 7

6 5 2

8 0 3

1 4 7

6 0 2

8 5 3

1 0 7

6 4 2

8 5 3

1 7 0

6 4 2

8 5 3

1 7 2

6 4 0

8 5 3

1 7 2

6 4 3

8 5 0

1 7 2

6 4 3

8 0 5

1 7 2

6 0 3

8 4 5

1 7 2

0 6 3

8 4 5

1 7 2

8 6 3

0 4 5

1 7 2

8 6 3

4 0 5

1 7 2

8 0 3

4 6 5

1 0 2

8 7 3

4 6 5

1 2 0

8 7 3

4 6 5

1 2 3

8 7 0

4 6 5

1 2 3

8 7 5

4 6 0

1 2 3

8 7 5

4 0 6

1 2 3

8 0 5

4 7 6

1 2 3

0 8 5

4 7 6

1 2 3

4 8 5

0 7 6

1 2 3

4 8 5

7 0 6

1 2 3

4 0 5

7 8 6

1 2 3

4 5 0

7 8 6

1 2 3

4 5 6

7 8 0