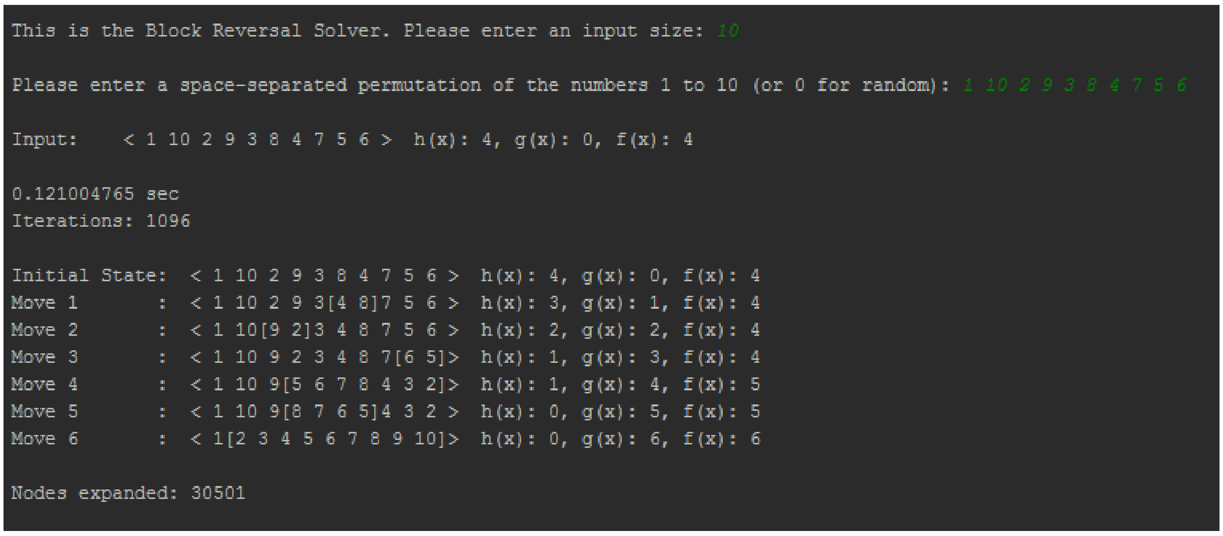
Kyle Janssen,
Aman Gill
CS 480
Spring 2014

Project #1 – A* search

This project was done using Java. Run the program, BlockReversalSolver, on a Java 1.7 equipped machine with “java -jar BlockReversalSolver.jar”. It will ask the size of the permutation you would like to use, then either take one from manual input or create one randomly.

It will then find an optimal path using the A* algorithm, and print out the amount of time the algorithm took, the number of iterations the algorithm took, the path it found from the initial state to the goal state, with the moves it took marked at each step, and the number of expanded nodes.

Test case 1:

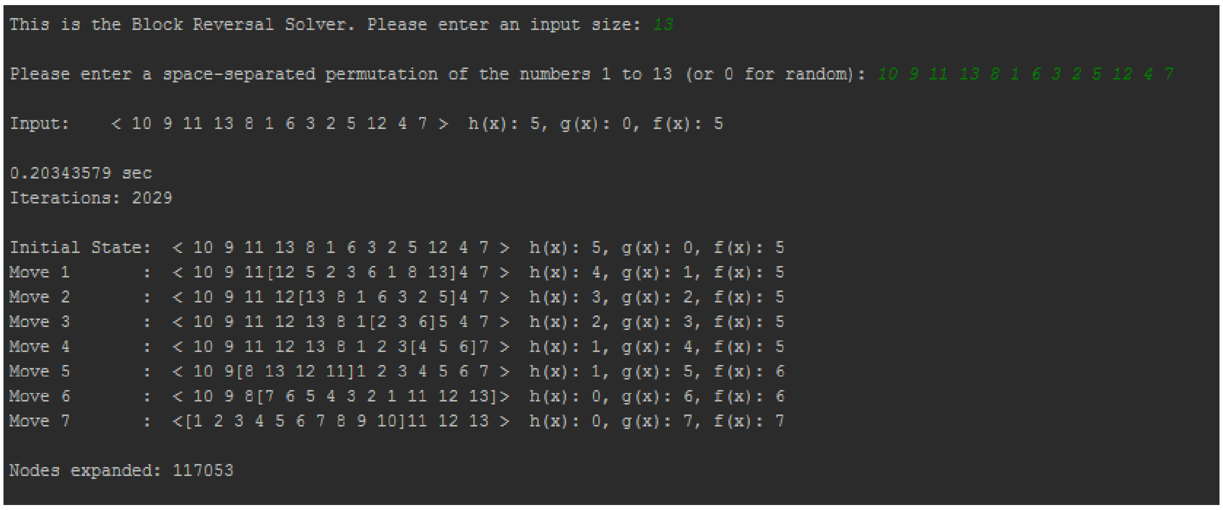


```
This is the Block Reversal Solver. Please enter an input size: 10

Please enter a space-separated permutation of the numbers 1 to 10 (or 0 for random): 1 10 2 9 3 8 4 7 5 6

Input:    < 1 10 2 9 3 8 4 7 5 6 >  h(x): 4, g(x): 0, f(x): 4

0.121004765 sec
Iterations: 1096

Initial State:  < 1 10 2 9 3 8 4 7 5 6 >  h(x): 4, g(x): 0, f(x): 4
Move 1       :  < 1 10 2 9 3[4 8]7 5 6 >  h(x): 3, g(x): 1, f(x): 4
Move 2       :  < 1 10[9 2]3 4 8 7 5 6 >  h(x): 2, g(x): 2, f(x): 4
Move 3       :  < 1 10 9 2 3 4 8 7[6 5]>  h(x): 1, g(x): 3, f(x): 4
Move 4       :  < 1 10 9[5 6 7 8 4 3 2]>  h(x): 1, g(x): 4, f(x): 5
Move 5       :  < 1 10 9[8 7 6 5]4 3 2 >  h(x): 0, g(x): 5, f(x): 5
Move 6       :  < 1[2 3 4 5 6 7 8 9 10]>  h(x): 0, g(x): 6, f(x): 6

Nodes expanded: 30501
```

The way this implementation of A* works is by using a HashMap to keep track of the states that are in the OPEN set, that is, the ones that have been visited. It also uses a PriorityQueue, which in Java is based on a heap, to keep track of the states that have yet to be evaluated, that is, they are in the FRINGE set.

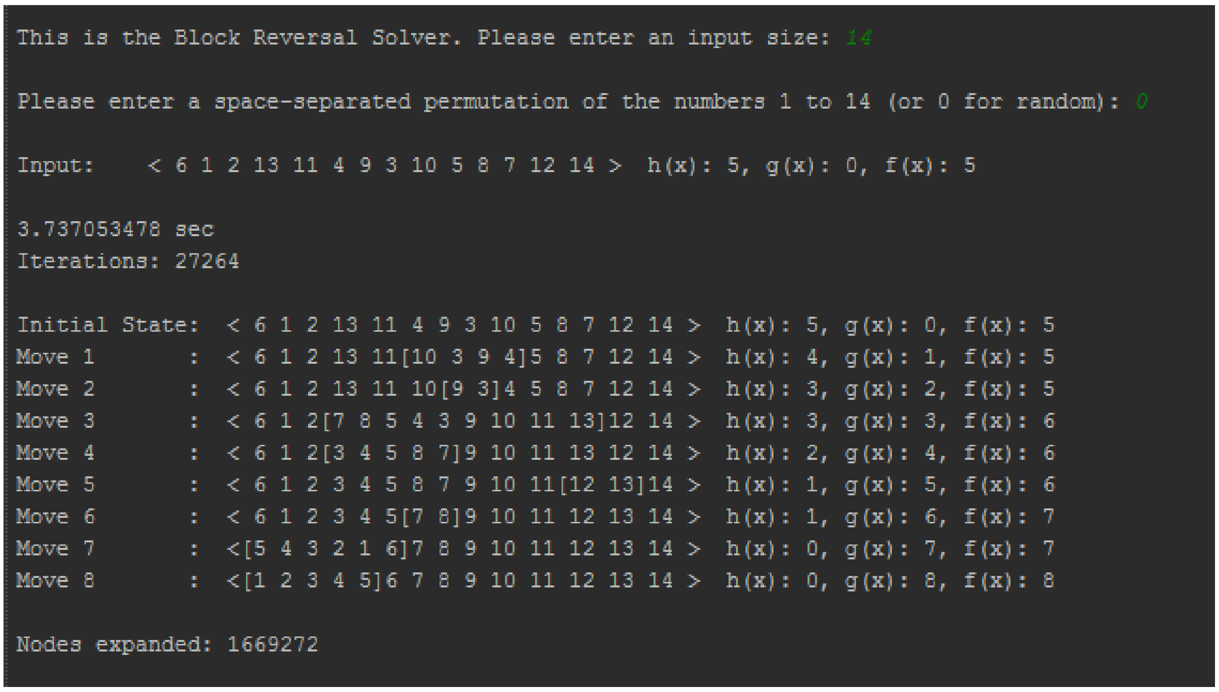
This program was tested with numerous inputs of various sizes and mostly random permutations and has proven to be quite fast, with most inputs with an n of 10 or less being completed in less than a second.

Test case 2:



```
This is the Block Reversal Solver. Please enter an input size: 13

Please enter a space-separated permutation of the numbers 1 to 13 (or 0 for random): 10 9 11 13 8 1 6 3 2 5 12 4 7

Input:    < 10 9 11 13 8 1 6 3 2 5 12 4 7 >  h(x): 5, g(x): 0, f(x): 5

0.20343579 sec
Iterations: 2029

Initial State:  < 10 9 11 13 8 1 6 3 2 5 12 4 7 >  h(x): 5, g(x): 0, f(x): 5
Move 1       :  < 10 9 11[12 5 2 3 6 1 8 13]4 7 >  h(x): 4, g(x): 1, f(x): 5
Move 2       :  < 10 9 11 12[13 8 1 6 3 2 5]4 7 >  h(x): 3, g(x): 2, f(x): 5
Move 3       :  < 10 9 11 12 13 8 1[2 3 6]5 4 7 >  h(x): 2, g(x): 3, f(x): 5
Move 4       :  < 10 9 11 12 13 8 1 2 3[4 5 6]7 >  h(x): 1, g(x): 4, f(x): 5
Move 5       :  < 10 9[8 13 12 11]1 2 3 4 5 6 7 >  h(x): 1, g(x): 5, f(x): 6
Move 6       :  < 10 9 8[7 6 5 4 3 2 1 11 12 13]>  h(x): 0, g(x): 6, f(x): 6
Move 7       :  <[1 2 3 4 5 6 7 8 9 10]11 12 13 >  h(x): 0, g(x): 7, f(x): 7

Nodes expanded: 117053
```

Random test case:



```
This is the Block Reversal Solver. Please enter an input size: 14

Please enter a space-separated permutation of the numbers 1 to 14 (or 0 for random): 0

Input:    < 6 1 2 13 11 4 9 3 10 5 8 7 12 14 >  h(x): 5, g(x): 0, f(x): 5

3.737053478 sec
Iterations: 27264

Initial State:  < 6 1 2 13 11 4 9 3 10 5 8 7 12 14 >  h(x): 5, g(x): 0, f(x): 5
Move 1       :  < 6 1 2 13 11[10 3 9 4]5 8 7 12 14 >  h(x): 4, g(x): 1, f(x): 5
Move 2       :  < 6 1 2 13 11 10[9 3]4 5 8 7 12 14 >  h(x): 3, g(x): 2, f(x): 5
Move 3       :  < 6 1 2[7 8 5 4 3 9 10 11 13]12 14 >  h(x): 3, g(x): 3, f(x): 6
Move 4       :  < 6 1 2[3 4 5 8 7]9 10 11 13 12 14 >  h(x): 2, g(x): 4, f(x): 6
Move 5       :  < 6 1 2 3 4 5 8 7 9 10 11[12 13]14 >  h(x): 1, g(x): 5, f(x): 6
Move 6       :  < 6 1 2 3 4 5[7 8]9 10 11 12 13 14 >  h(x): 1, g(x): 6, f(x): 7
Move 7       :  <[5 4 3 2 1 6]7 8 9 10 11 12 13 14 >  h(x): 0, g(x): 7, f(x): 7
Move 8       :  <[1 2 3 4 5]6 7 8 9 10 11 12 13 14 >  h(x): 0, g(x): 8, f(x): 8

Nodes expanded: 1669272
```