Artificial Intelligence Lab

Name : Sandip Kumar Maity
Roll : 001911001032
Department : Information Technology (UG-3)
Semester : 5
Assignment 5: Implement A star algorithm for eight puzzle problem

Assignment Description :

Consider two heuristic functions (h1(n) and h2(n)) along with g(n) and execute the algorithm both the times with h1 and g and then h2 and g. Compare the result of both the run and write the document.

Things to submit:
1. Algorithm implementation
2. Screenshots of the output
3. Document describing heuristic and output along with the understanding of the outputs for both the cases.

Manhattan Distance As Heuristic Function :

The *Manhattan Distance* between a board and the goal board is the sum of the Manhattan distances (*sum of the vertical and horizontal distance*) from the tiles to their goal positions.

Here, we are using this Manhattan Distance as Heuristic function in our A* algorithm.

Input Used :

```
1 2 3
8 6 4
7 5 0

0 2 8
1 4 3
7 6 5
```

Output :

```
Iteration #1     -->    h1(n) + g(n) = 8
Iteration #2     -->    h1(n) + g(n) = 8
Iteration #3     -->    h1(n) + g(n) = 8
Iteration #4     -->    h1(n) + g(n) = 8
Iteration #5     -->    h1(n) + g(n) = 8
Iteration #6     -->    h1(n) + g(n) = 8
Iteration #7     -->    h1(n) + g(n) = 8
Iteration #8     -->    h1(n) + g(n) = 10
Iteration #9     -->    h1(n) + g(n) = 10
Iteration #10    -->    h1(n) + g(n) = 10
Iteration #11    -->    h1(n) + g(n) = 10
Iteration #12    -->    h1(n) + g(n) = 10
Iteration #13    -->    h1(n) + g(n) = 10
Iteration #14    -->    h1(n) + g(n) = 10
Iteration #15    -->    h1(n) + g(n) = 10
Iteration #16    -->    h1(n) + g(n) = 10
Iteration #17    -->    h1(n) + g(n) = 10
Iteration #18    -->    h1(n) + g(n) = 10
Iteration #19    -->    h1(n) + g(n) = 10
Iteration #20    -->    h1(n) + g(n) = 10
Total no. of Iterations : 20

States from start to goal :
1 2 3
8 6 4
7 5 0

1 2 3
8 6 4
7 0 5

1 2 3
8 0 4
7 6 5
```

```
1 2 3
0 8 4
7 6 5

0 2 3
1 8 4
7 6 5

2 0 3
1 8 4
7 6 5

2 8 3
1 0 4
7 6 5

2 8 3
1 4 0
7 6 5

2 8 0
1 4 3
7 6 5

2 0 8
1 4 3
7 6 5

0 2 8
1 4 3
7 6 5

Total no. of moves to solve the puzzle : 10
```

Hamming Distance As Heuristic Function :

The *Hamming Distance* between a board and the goal board is *the number of tiles in the wrong position.*

Here, we are using this Hamming Distance as Heuristic function in our A* algorithm.

***Input Used* :**

```
1 2 3
8 6 4
7 5 0

0 2 8
1 4 3
7 6 5
```

***Output* :**

```
Iteration #1     -->    h2(n) + g(n) = 6
Iteration #2     -->    h2(n) + g(n) = 6
Iteration #3     -->    h2(n) + g(n) = 6
Iteration #4     -->    h2(n) + g(n) = 6
Iteration #5     -->    h2(n) + g(n) = 6
Iteration #6     -->    h2(n) + g(n) = 7
Iteration #7     -->    h2(n) + g(n) = 7
Iteration #8     -->    h2(n) + g(n) = 7
Iteration #9     -->    h2(n) + g(n) = 7
Iteration #10    -->    h2(n) + g(n) = 8
Iteration #11    -->    h2(n) + g(n) = 8
Iteration #12    -->    h2(n) + g(n) = 8
Iteration #13    -->    h2(n) + g(n) = 8
Iteration #14    -->    h2(n) + g(n) = 8
Iteration #15    -->    h2(n) + g(n) = 9
Iteration #16    -->    h2(n) + g(n) = 9
Iteration #17    -->    h2(n) + g(n) = 9
Iteration #18    -->    h2(n) + g(n) = 9
Iteration #19    -->    h2(n) + g(n) = 9
Iteration #20    -->    h2(n) + g(n) = 9
Iteration #21    -->    h2(n) + g(n) = 9
Iteration #22    -->     h2(n) + g(n) = 9
Iteration #23    -->     h2(n) + g(n) = 9
Iteration #24    -->     h2(n) + g(n) = 9
Iteration #25    -->     h2(n) + g(n) = 9
Iteration #26    -->     h2(n) + g(n) = 10
Iteration #27    -->     h2(n) + g(n) = 10
Iteration #28    -->     h2(n) + g(n) = 10
Iteration #29    -->     h2(n) + g(n) = 10
Iteration #30    -->     h2(n) + g(n) = 10
Iteration #31    -->     h2(n) + g(n) = 10
Iteration #32    -->     h2(n) + g(n) = 10
Iteration #33    -->     h2(n) + g(n) = 10
Iteration #34    -->     h2(n) + g(n) = 10
Iteration #35    -->     h2(n) + g(n) = 10
Iteration #36    -->     h2(n) + g(n) = 10
Iteration #37    -->     h2(n) + g(n) = 10
Iteration #38    -->     h2(n) + g(n) = 10
Iteration #39    -->     h2(n) + g(n) = 10
Iteration #40    -->     h2(n) + g(n) = 10
Iteration #41    -->     h2(n) + g(n) = 10
Iteration #42    -->     h2(n) + g(n) = 10
Iteration #43    -->     h2(n) + g(n) = 10
Total no. of Iterations : 43

States from start to goal :
1 2 3
8 6 4
7 5 0

1 2 3
8 6 4
7 0 5

1 2 3
8 0 4
7 6 5
```

```
1 2 3
0 8 4
7 6 5

0 2 3
1 8 4
7 6 5

2 0 3
1 8 4
7 6 5

2 8 3
1 0 4
7 6 5

2 8 3
1 4 0
7 6 5

2 8 0
1 4 3
7 6 5

2 0 8
1 4 3
7 6 5

0 2 8
1 4 3
7 6 5

Total no. of moves to solve the puzzle : 10
```

Conclusion :

1. As we can clearly see from the above outputs, we are reaching the goal state in *20 iterations using Manhattan Distance* as Heuristic function for our A* search implementation and we are reaching the goal state in *43 iterations using Hamming Distance* as Heuristic function for our A* search implementation.

2. Clearly, it is taking less time for the Manhattan Distance as heuristic function (Total Iterations : 20) than the Hamming Distance as the heuristic function (Total Iterations : 43) for our A* search implementation.

3. The range of values of Hamming distance is smaller than the range of the values of Manhattan distance. So, there may be two different states where hamming distance is equal for both of them but Manhattan distances are different and whose Manhattan distance is smaller than the other, that state is much closer to the goal state.

4. Therefore, ***Manhattan Distance is better Heuristic function than Hamming Distance in our A* search***.