CS561 -ARTIFICIALINTELLIGENCELAB

ASSIGNMENT-4 : HILL CLIMBING

Intaj Choudhury – 2211MC09

Ankit Anand – 2311MC04

Khushbu Bharti – 2311MC21

QUESTION :

A local search algorithm tries to find the optimal solution by exploring the states in the local region. Hill climbing is a local search technique that always looks for a better solution in its neighbourhood.

a. Implement the Hill Climbing Search Algorithm for solving the 8-puzzle problem.

b. Check the algorithm for the following heuristics:

i. h1(n) = number of tiles displaced from their destined position.

ii. h2(n) = sum of the Manhattan distance of each tile from the goal position

Algorithm:

STEP1 : Take the initial state of the puzzle from user. Target state is fixed.
STEP2 : Check whether the puzzle is solvable or not by counting number of inversion.
STEP 3 : If puzzle is Solvable. We initialize the constructer of class "state" with parameters value and hx , creating object for initial state.

STEP 4 : Taking input from user for the hx function. According to the input we are calculating the hx value.

STEP 5 : Calculate the hx value for all the children of the of the current node . Putting the heuristic value of all children in open list and then sorting the list.

STEP 6 : Check the minimum hx from the open list (open[0])

- If h value = 0 :
  - Then target state is reached.
  - Exit.
- If this h value is greater than the hx of parent state,
  - then local maxima is reached and searching will end.
  - Exit.
- If this h is less than hx of parent
  - Then we will put current state = open[0]
  - Repeat step 5 and step 6.
- If the h value is equal to hx of parent
  - We will check for flat. if it's a flat then exit, otherwise repeat step 5 and 6.

Case 1: Target reached

```
Enter Initial State :
1 2 3
-1 4 6
7 5 8

-------Initial State-------

[1, 2, 3]
[-1, 4, 6]
[7, 5, 8]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]



--------Choose huristic : -------

1. Missplaced Tiles

2. Manhattan Distance

1
[1, 2, 3]
[-1, 4, 6]
[7, 5, 8]


[1, 2, 3]
[4, -1, 6]
[7, 5, 8]


[1, 2, 3]
[4, 5, 6]
[7, -1, 8]


[1, 2, 3]
[4, 5, 6]
[7, 8, -1]


Target Reached

Length of Optimal path :  3
No. of steps explored = 3
Total execution time in minute :
  0.06502477
```

Case 2 : Local Maxima

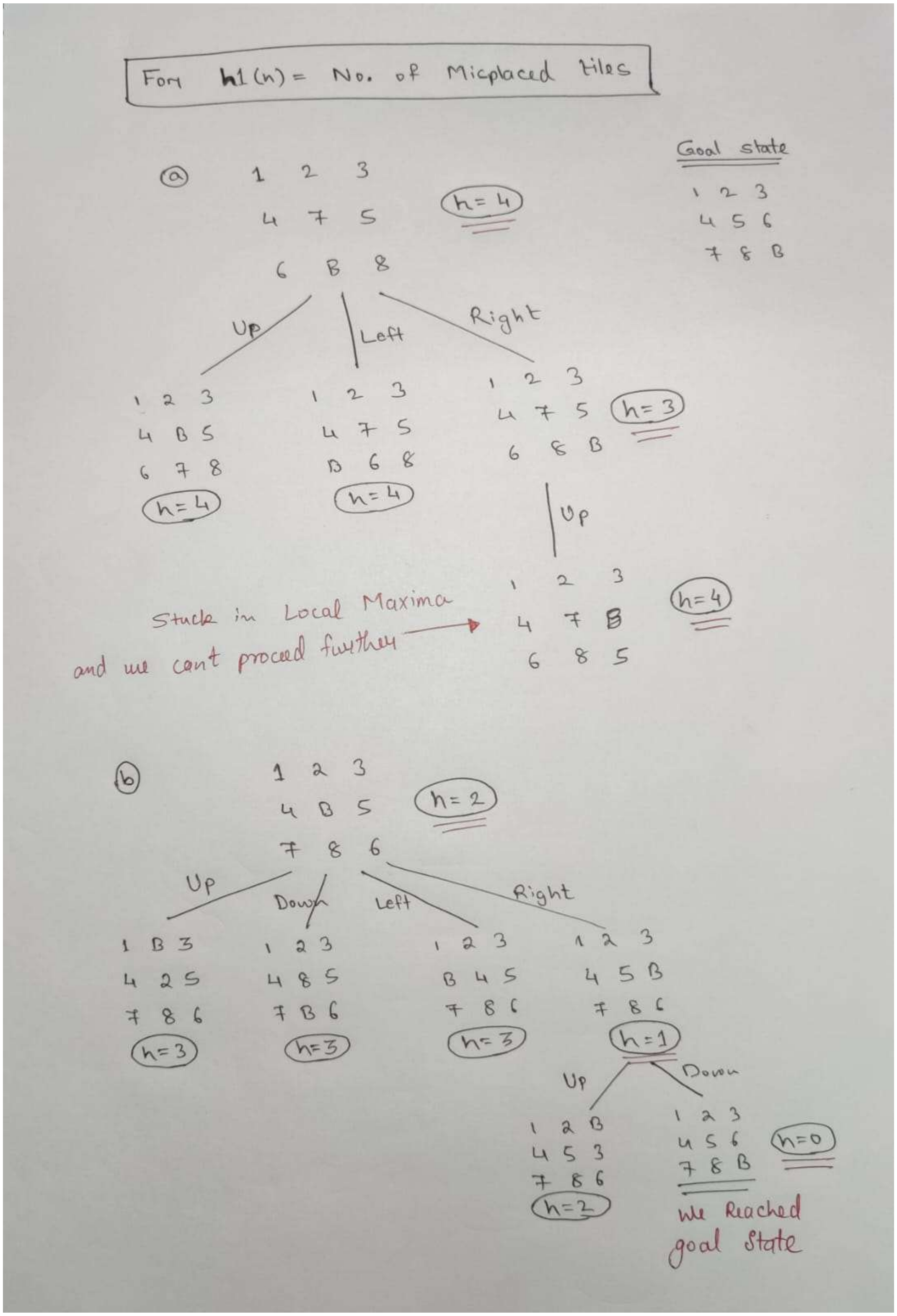
```
Enter Initial State :
1 2 3
4 7 5
6 -1 8

-------Initial State-------

[1, 2, 3]
[4, 7, 5]
[6, -1, 8]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]



--------Choose huristic : -------

1. Missplaced Tiles

2. Manhattan Distance

2
[1, 2, 3]
[4, 7, 5]
[6, -1, 8]


[1, 2, 3]
[4, 7, 5]
[6, 8, -1]


Local maxima reached

No. of steps explored = 2
Total execution time in minute :
  0.061489692222222225
```

Case 3 : Stuck in Flat or Shoulder

```
Enter Initial State :
1 2 3
6 4 5
7 8 -1

-------Initial State-------

[1, 2, 3]
[6, 4, 5]
[7, 8, -1]


-------Target State-------

[1, 2, 3]
[4, 5, 6]
[7, 8, -1]



--------Choose huristic : -------

1. Missplaced Tiles

2. Manhattan Distance

1
[1, 2, 3]
[6, 4, 5]
[7, 8, -1]


[1, 2, 3]
[6, 4, -1]
[7, 8, 5]


[1, 2, 3]
[6, -1, 4]
[7, 8, 5]
```

```
[1, 2, 3]
[6, -1, 4]
[7, 8, 5]


[1, 2, 3]
[6, 4, -1]
[7, 8, 5]


[1, 2, 3]
[6, -1, 4]
[7, 8, 5]


[1, 2, 3]
[6, 4, -1]
[7, 8, 5]


Stuck in Flat or Shoulder

[1, 2, 3]
[6, -1, 4]
[7, 8, 5]


No. of steps explored = 202
Total execution time in minute :
  0.028841053611111108
```

Demonstration for a sample input-



For $h2(n)$ = Manhattan Distance

(a)

1 2 3
4 7 5
6 B 8

h=7

Goal State

1 2 3
4 5 6
7 8 B

Up

1 2 3
4 B 5
6 7 8

h=6

Left

1 2 3
4 7 5
B 6 8

h=6

Right

1 2 3
4 7 5
6 8 B

h=6

Up

1 2 3
4 7 B
6 8 5

h=7

Stuck

(b)

1 2 3
4 B 5
7 8 6

h=2

Up

1 B 3
4 2 5
7 8 6

h=3

D

1 2 3
4 8 5
7 B 6

h=3

L

1 2 3
B 4 5
7 8 6

h=3

Right

1 2 3
4 5 B
7 8 6

h=1

U

1 2 B
4 5 3
7 8 6

h=2

D

1 2 3
4 5 6
7 8 B

h=0

Goal Reached