Anish Dubey
523002969
Artificial Intelligence - HW-2

First Implemented Heuristic Function :

h1 = 2.5*Curr_move + Fut_move + 0.5*num_blocks/num_ordered_elements_0th_stack -0.2*global_position

Explanation:

- **Curr_mov:** Number of moves required for next in order alphabet to come in 0th stack.
  eg:
  0 | A B C F G H
  1 | D L M
  ……..
  So next in order alphabet is D. It requires 2 moves to remove alphabet L, M and will take 3 moves to remove F, G and H. Thus 5 will be curr_mov here. This gives importance to current moves.

- **Fut_move**: Number of moves required for the rest of alphabet to come in 0th stack. Since this is a large number as compared to Curr_mov, a weightage is given to curr_mov factor. This is done in order to keep both factors equal weightage.

- ***Num_blocks/num_ordered_elements_0th_stack:*** In later part of searching, the above factor gives more importance to the blocks where 0th stack contains more ordered element. On trying different parameters, 0.5 showed a good result.

Combination of all above parameters resulted in good heuristic.

- ***Global Position:*** This factor gives importance to the search space where blocks are positioned in the right order.
  Like:
  Stack 4| A, B, C, D
  Here Since A needs to move out penalty is given to A as it needs to shift 3 blocks.
  Stack 4|D, C, B, A
  Here Since A is already in position, no penalty is given to this position.
  Ref:
  "http://www.cse.hcmut.edu.vn/~tru/AI/chapter3.pdf"

Second Standard Heuristic Function

h0 = Number of blocks out of place

Heuristic is implemented to compare with the implemented heuristic. Results shown below.

Evaluation Results are below

Evaluation metrics

BlocksWorld	Heuristic	Stacks	Blocks	Iterations	Queue Size	Solution Length
0 | B D A 1 | C 2 | E	h1	3	5	9	21	9
	h0	3	5	2519	595	9
0 | A E 1 | D B F 2 | G 3 | C	h1	4	7	12	70	11
	h0	4	7	Crossed 10000 Iterations. Code terminated		
1| IG A F E 2| J 3| D H 4| 5| C B	h1	5	10	32	410	21
	h0	5	10	Crossed 10000 Iterations. Code terminated		
1 | D **2 | E F I J** **3 | B G** **4 | C H**	**h1**	**5**	**10**	**41**	**541**	**18**

5 | A						
	h0	5	10	Crossed 10000 Iterations. Code terminated		
1|LJHKCEFI 2 | 3 | 4 | G D 5 | A 6 | B	h1	6	12	30	593	26
	h0	6	12	Crossed 10000 Iterations. Code terminated		
1|M H N 2|KO I G E 3|F J B A L 4| 5| 6| 7| C D	h1	7	15	32	882	27
	h0	7	15	Crossed 10000 Iterations. Code terminated		
1| L 2| B C G O 3| A N 4| H M K J 5| I P D 6| E 7| 8| F	h1	8	16	49	2023	30
	h0	8	16	Crossed 10000 Iterations.		

				Code terminated		
1|QPHAO KECJB 2|MNLRD 3| S 4| F I 5 | T 6| 7| 8| 9| G	h1	9	20	142	7837	43
	h0	9	20	Crossed 10000 Iterations. Code terminated		

As we can see, the heuristic performs very bad in case of standard heuristic function(h0).

Admissible:
Function value is f+g = 13.5
Depth at 9
0 |
1 | D A B
2 | C E

Since we know it can be reached by 9 steps but our heuristic estimates 13.5. So it is not admissible. But it is seen that in later iterations, heuristic becomes more closer to approximation and is closer to being admissible.

Scalability:

Stacks	Blocks	Iterations	Queue Size	Solution Path
3	5	**12**	30	11
3	6	**17**	39	14
3	7	**171**	363	16
3	8	**185**	420	20
3	9	**230**	574	24
3	10	**734**	1701	31

As we increase the number of blocks, problems becomes harder to solve.

Stacks	Blocks	Iterations	Queue Size	Solution Path
3	10	**734**	1701	31
4	10	**21**	135	18
5	10	**32**	414	20
6	10	**19**	312	17
7	10	**19**	475	17
8	10	**16**	427	16
9	10	**19**	732	18
10	10	**16**	585	16

As we increase the number of stack for a given block, it becomes easier to solve. Thus it shows that the above problem with the defined heuristic is **scalable**.

Transcript:

Below example shows transcript for following input:

```
0 |  D
1 |  E F I J
2 |  B G
3 |  C H
4 |  A
-----------------------------
Final Stage
0 | A B C D E F G H I J
1 |
2 |
3 |
4 |
-----
Search Algorithm A Star Search
Total Iterations 41
Max Frontier Size 541
Goal Reached
Depth 18
Depth at 18
0 |  D
1 |  E F I J
2 |  B G
3 |  C H
4 |  A
-----------
Depth at 17
0 |
1 |  E F I J
2 |  B G
3 |  C H D
4 |  A
-----------
Depth at 16
0 |  A
1 |  E F I J
2 |  B G
```

```
3 |  C H D
4 |
-----------
Depth at 15
0 |  A
1 |  E F I J
2 |  B
3 |  C H D
4 |  G
-----------
Depth at 14
0 |  A
1 |  E F I J
2 |  B
3 |  C H
4 |  G D
-----------
Depth at 13
0 |  A B
1 |  E F I J
2 |
3 |  C H
4 |  G D
-----------
Depth at 12
0 |  A B
1 |  E F I
2 |  J
3 |  C H
4 |  G D
-----------
Depth at 11
0 |  A B
1 |  E F I
2 |  J H
3 |  C
4 |  G D
-----------
Depth at 10
0 |  A B C
1 |  E F I
2 |  J H
3 |
4 |  G D
-----------
Depth at 9
0 |  A B C
1 |  E F
2 |  J H
3 |  I
4 |  G D
-----------
Depth at 8
0 |  A B C D
1 |  E F
2 |  J H
3 |  I
4 |  G
-----------
Depth at 7
0 |  A B C D
1 |  E
2 |  J H
3 |  I F
4 |  G
-----------
Depth at 6
0 |  A B C D E
1 |
2 |  J H
3 |  I F
4 |  G
-----------
Depth at 5
0 |  A B C D E F
1 |
2 |  J H
3 |  I
4 |  G
-----------
Depth at 4
0 |  A B C D E F G
1 |
2 |  J H
3 |  I
4 |
-----------
Depth at 3
0 |  A B C D E F G H
1 |
2 |  J
3 |  I
4 |
-----------
Depth at 2
0 |  A B C D E F G H I
1 |
2 |  J
3 |
4 |
-----------
Depth at 1
0 |  A B C D E F G H I J
1 |
2 |
3 |
4 |
-----------
```