Programing Assignment #2

CSCE 625 - Artificial Intelligence
Spring 2015

Name: Xiaoshu Zhang

Email: kallen5208@gmail.com

1. My heuristic

The BlocksWorld actions are described as:

"A block can move only from a top location to only the top of any other stack".

Since a block could be placed to its right place only after its previous has been placed to the right place, an operation towards the optimal path should always devote to move the first misplaced block to its right place. Then we can generate a relaxed problem:

"The first misplaced block can move only from a top location(but ignore blocks put on it later) to only the top of any other stack. But other blocks can move from any location to only the top of any other stack."

We can derive a heuristic from this relaxed problem. The operation procedure is:

(1) Find the first misplaced block's ordinal.

(2) Clear the first stack until the first misplaced block is removed, this step-cost is h_1.

(3) Find the first misplaced block's place.

(4) Remove the first misplaced block's upper blocks, this step-cost is h_2.

(5) Put the first misplaced block to its right place, this step-cost is 1.

(6) Put other blocks to their right places, each block's step-cost is 1.

Here, step-cost $h_3 = (5) + (6)$. Total $h = h_1 + h_2 + h_3$.

Example:

1 | D F

2 | C A G

3 | B E

For this stack-3 block-7 problem: $h_1 = 2$, $h_2 = 1$, $h_3 = 7$.

Because the derived heuristic is an exact cost for the relaxed problem, it obeys the triangle inequality and is therefore consistent. And is admissible of course.

2. Default heuristic

The default heuristic is number of blocks out of place. The default heuristic is just h_3, a part of my heuristic. As my heuristic is never less than the default heuristic, my heuristic is better.

3. Performances

On managing frontier, A* search uses a priority queue sorted on $f(n) = g(n) + h(n)$. $g(n)$ is the number of steps already used, $h(n)$ is the estimated number of remaining steps needed to reach to goal.

Problems of stacks = 3, 5, 7, and blocks = 5, 6, 7, 8, 9, 10 have been chosen to test my heuristic's performances. Besides, a default heuristic(number of blocks out of place) is also tested

to get a comparison.

Parts of transcripts are shown in Appendix A and Appendix B.

The detailed transcripts are in "BlocksWorld.txt".

The performance of my heuristic is shown in Table 1.

The performance of default heuristic is shown in Table 2.

Table 1 My heuristic's performance

stacks	blocks	total iterations	max queue size	path length
3	5	30	49	9
	6	37	70	10
	7	383	699	13
	8	6319	8608	17
	9	3966	5571	17
	10	9824	12969	20
5	5	63	269	7
	6	87	536	8
	7	157	807	9
	8	1247	7607	12
	9	426	3108	12
	10	4835	29483	15
7	5	83	841	7
	6	20	277	7
	7	8	143	7
	8	1786	22647	11
	9	80	1420	11
	10	442	7695	13

Table 2 Default heuristic's performance

stacks	blocks	total iterations	max queue size	path length
3	5	10	21	5
	6	803	946	11
	7	1910	2584	13
	8	2890	4082	14
	9	8768	12805	16
5	5	19	78	4
	6	57	332	7
	7	1387	5920	10
	8	16050	60151	12
7	5	40	501	6
	6	472	3707	8
	7	54	1041	8
	8	684	8900	9
	9	893	10682	11

	10	999	12872	12

4. Data comparison and analysis

Excluding some individuals, my heuristic always has less total iterations and smaller max queue size, indicating my heuristic is better.

The path length' situations are the same, because both two heuristics are admissible and consistent, resulting to optimal solution paths.

For my heuristic, when blocks numbers are small(less than 7), problems having fewer stacks run faster. This is because fewer stacks have smaller branching factors.

When when blocks numbers are big(greater than 7), problems having more stacks run faster. This is because more stacks mean more place to sparse the blocks and make the game easier.

For a given number of stacks, increasing the block number always increases the total iteration. This is because usually more blocks need more steps to place them to right places.

---------- There are appendixes in the next pages. ----------

Appendix A: My heuristic's performance on 3 stacks and 7 blocks

initial state:
1 | D A F B
2 |
3 | E C G
depth: 0, heuristic: 11

iter=0, queue=0, f=g+h=11, depth=0
iter=1, queue=3, f=g+h=11, depth=1
iter=2, queue=6, f=g+h=11, depth=1
iter=3, queue=7, f=g+h=11, depth=2
iter=4, queue=10, f=g+h=11, depth=2
iter=5, queue=13, f=g+h=11, depth=2
iter=6, queue=16, f=g+h=11, depth=2
iter=7, queue=17, f=g+h=11, depth=3
iter=8, queue=20, f=g+h=11, depth=3
. . .
. . .
iter=104, queue=199, f=g+h=12, depth=7
iter=105, queue=202, f=g+h=12, depth=8
iter=106, queue=205, f=g+h=12, depth=9
iter=107, queue=208, f=g+h=12, depth=10
iter=108, queue=209, f=g+h=12, depth=11
iter=109, queue=210, f=g+h=12, depth=12
success! depth=12, total_goal_tests=110, max_queue_size=210
1 | D A F B
2 |
3 | E C G
depth: 0, heuristic: 11
1 | D A F B
2 | G
3 | E C
depth: 1, heuristic: 11
1 | D A F
2 | G
3 | E C B
depth: 2, heuristic: 10
1 | D A
2 | G F
3 | E C B
depth: 3, heuristic: 9
1 | D
2 | G F

3 | E C B A
depth: 4, heuristic: 8
1 |
2 | G F D
3 | E C B A
depth: 5, heuristic: 7
1 | A
2 | G F D
3 | E C B
depth: 6, heuristic: 6
1 | A B
2 | G F D
3 | E C
depth: 7, heuristic: 5
1 | A B C
2 | G F D
3 | E
depth: 8, heuristic: 4
1 | A B C D
2 | G F
3 | E
depth: 9, heuristic: 3
1 | A B C D E
2 | G F
3 |
depth: 10, heuristic: 2
1 | A B C D E F
2 | G
3 |
depth: 11, heuristic: 1
1 | A B C D E F G
2 |
3 |
depth: 12, heuristic: 0

Appendix B: Default heuristic's performance on 3 stacks and 7 blocks

Use the default heuristic.
initial state:
1 | D E
2 | F C
3 | B A G
depth: 0, heuristic: 7

iter=0, queue=0, f=g+h=7, depth=0
iter=1, queue=5, f=g+h=8, depth=1
iter=2, queue=8, f=g+h=8, depth=1
iter=3, queue=11, f=g+h=8, depth=1
iter=4, queue=14, f=g+h=8, depth=1
iter=5, queue=17, f=g+h=8, depth=1
iter=6, queue=20, f=g+h=8, depth=1
. . .
. . .
iter=2063, queue=2480, f=g+h=13, depth=10
iter=2064, queue=2480, f=g+h=13, depth=12
iter=2065, queue=2481, f=g+h=13, depth=11
iter=2066, queue=2480, f=g+h=13, depth=13
success! depth=13, total_goal_tests=2067, max_queue_size=2481
1 | D E
2 | F C
3 | B A G
depth: 0, heuristic: 7
1 | D E
2 | F
3 | B A G C
depth: 1, heuristic: 7
1 | D
2 | F E
3 | B A G C
depth: 2, heuristic: 7
1 |
2 | F E D
3 | B A G C
depth: 3, heuristic: 7
1 |
2 | F E D C
3 | B A G
depth: 4, heuristic: 7
1 |

2 | F E D C G
3 | B A
depth: 5, heuristic: 7
1 | A
2 | F E D C G
3 | B
depth: 6, heuristic: 6
1 | A B
2 | F E D C G
3 |
depth: 7, heuristic: 5
1 | A B
2 | F E D C
3 | G
depth: 8, heuristic: 5
1 | A B C
2 | F E D
3 | G
depth: 9, heuristic: 4
1 | A B C D
2 | F E
3 | G
depth: 10, heuristic: 3
1 | A B C D E
2 | F
3 | G
depth: 11, heuristic: 2
1 | A B C D E F
2 |
3 | G
depth: 12, heuristic: 1
1 | A B C D E F G
2 |
3 |
depth: 13, heuristic: 0