**Heuristics for BlocksWorld Problem**

**DESCRIPTION OF HEURISTICS:**

**For h(0):**

F(n)=g(n)+h(n)

where,

g(n)=g(n-1)+1; {g(0)=0 for root node}

h(n)= No of blocks out of place

**For h(1)**

f(n)=g(n)+h(n)

where,

g(n)=g(n-1)+1; {g(0)=0 for root node}

h(n)= s1+s2+W\*s3;

s1=No of blocks to be popped out of Stack 1 which is equal to number of blocks since the first element out of place.

s2= No of blocks to be popped out of other stacks which is equal to number of blocks above the minimum value block.

s3=number of blocks needed to be inserted in sorted order on Stack 1 to reach goal state.

W=can be any integer, here considered as (number of Stacks -1)

**Argument in favor of Admissibility:**

DEPTH 0:

Stack1: 1,2,9,4

Stack2: 5,8,

Stack3: 10,6,7,3

Stack 1 has to be in sorted state, so all the elements above the first element out of place in Stack 1 needs to be popped out of the stack. This gives s1 which is No of elements to be popped out of Stack 1.

In the above example, although 1, 2 and 4 are in right places, we need to pop out elements after the first unsorted element which is 9. Hence s1=2.

Also, s2=1+0=1 and s3=10-2=8

Considering w=number of Stack-1=3-1=2

Hence, f(n)=g(n)+h(n)

=0+(2+1+2\*8)

=19.

DEPTH 1:

|  |  |
| --- | --- |
| State 1 | State2 |
| Stack1: 1,2,9  Stack2: 5,8,4  Stack3: 10,6,7,3 | Stack1: 1,2,9  Stack2: 5,8,  Stack3: 10,6,7,3,4 |
| f(n)=1+(1+0+2\*8)=18 | f(n)=1+(1+2+2\*8)=20 |

DEPTH 2 (derived from State 1):

Stack1: 1,2,

Stack2: 5,8,4,9

Stack3: 10,6,7,3

The next least number must be popped out from the remaining stacks, wherever it may be present and pushed into Stack 1.

In the example above, for depth 1, the next element to be pushed o to Stack 1 must be 3 to get a sorted stack. Number of elements that must be popped in order to retrieve 3=0 and to retrieve 4=0.

Hence s2=0+0=0

Also, s1=1, s3=8, W=2

Therefore, g(n)=1+(1+0+8\*2)=18

For depth 2:

s1=0

s2=1+0=0

s3=8

g(n)=2+(0+1+2\*8)=19

DEPTH 3:

Comparison between two states:

|  |  |
| --- | --- |
| State 1 | State2 |
| Stack1: 1,2,3  Stack2: 5,8,4,9  Stack3: 10,6,7 | State2  Stack1: 1,2  Stack2: 5,8,4,9,3  Stack3: 10,6,7 |
| Without considering weight W,  f(n)=g(n)+h(n)  =3+(0+2+7)  =12 | Without considering weight W,  f(n)=g(n)+h(n)  =3+(0+1+8)  =12 |
| With Weight = number of Stacks -1 = 2  f(n)=3+(0+2+7\*2)  =19 | With Weight = number of Stacks -1 = 2  f(n)=3+(0+2+8\*2)  =21 |

Since, state 2 will be equivalent to state 1 if we pop 3 from stack 2 and push it onto stack 1.

Hence, selecting State 1 in the first place would give a solution with lesser number of goal tests and depth. Therefore, it is necessary to quantify the extent to which Stack 1 is sorted and to select the State with the maximum number of elements in sorted state.

One way would have been to subtract the number of elements present in sorted order in Stack 1, but that would have resulted in a negative value. So instead, we can use a weighted number of elements yet to be added in order to reach the goal state.

Hence the first stack will prefer only insertion of the next ordered block, preventing large number of push and pop operations on it. Hence the blocks would be pushed and popped mostly from the remaining stacks (n-1, if =number of stacks).

Hence weight has been considered as n-1.

**Table of Statistics:**

**Table 1: Comparison with respect to different parameters**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Blocks Out of Place Heuristics: h(0)** | | | **New Heuristics: h(1)** | | |
| **Depth** | **Goal Tests** | **Maximum Queue Size** | **Depth** | **Goal Tests** | **Maximum Queue Size** |
| **Stacks=3 Blocks=5** | 11 | 658 | 519 | 12 | 35 | 60 |
| **Stacks=3 Blocks=5** | 7 | 55 | 84 | 7 | 10 | 22 |
| **Stacks=3 Blocks=6** | 11 | 404 | 543 | 11 | 18 | 34 |
| **Stacks=3 Blocks=6** | 11 | 669 | 773 | 11 | 16 | 34 |
| **Stacks=3 Blocks=7** | 14 | 4461 | 5369 | 14 | 22 | 44 |
| **Stacks=3 Blocks=7** | 13 | 2020 | 2615 | 15 | 23 | 53 |
| **Stacks=4 Blocks=5** | 8 | 240 | 541 | 8 | 11 | 60 |
| **Stacks=4 Blocks=5** | 7 | 117 | 296 | 7 | 11 | 49 |
| **Stacks=4 Blocks=5** | 8 | 251 | 594 | 8 | 10 | 45 |
| **Stacks=4 Blocks=6** | 10 | 1078 | 2414 | 10 | 12 | 55 |
| **Stacks=4 Blocks=6** | 11 | 1996 | 4183 | 11 | 21 | 111 |
| **Stacks=4 Blocks=7** | 9 | 166 | 656 | 9 | 10 | 52 |
| **Stacks=4 Blocks=7** | 11 | 1495 | 4235 | 11 | 31 | 183 |
| **Stacks=4 Blocks=7** | 12 | 3666 | 9991 | 12 | 13 | 57 |
| **Stacks=4 Blocks=7** | 12 | 3927 | 9907 | 12 | 16 | 63 |
| **Stacks= 5 Blocks=6** | 9 | 480 | 2289 | 9 | 11 | 81 |
| **Stacks=5 Blocks=6** | 10 | 2869 | 8883 | 9 | 10 | 92 |
| **Stacks=5 Blocks=7** | 11 | 4327 | 17037 | 11 | 17 | 166 |
| **Stacks=5 Blocks=7** | 11 | 3974 | 16713 | 11 | 19 | 148 |
| **Stacks=5 Blocks=7** | 11 | 4495 | 18370 | 12 | 20 | 197 |
| **Mean** | 10.35 | 1867.4 | 5300.6 | 10.5 | 16.8 | 80.3 |

For number of stacks=5 and number of blocks =7, number of Goal Tests for h(1)is reduced by 99% when compared to h(0)while maximum queue size is reduced by 98%.

**Table 1: Comparison with Stack=4, Block=6**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Blocks Out of Place Heuristics: h(0)** | | | **New Heuristics: h(1)** | | |
| **Depth** | **Goal Tests** | **Maximum Queue Size** | **Depth** | **Goal Tests** | **Maximum Queue Size** |
| **Stacks=4 Blocks=6** | 10 | 1159 | 2517 | 10 | 13 | 70 |
| **Stacks=4 Blocks=6** | 11 | 3994 | 6323 | 12 | 17 | 98 |
| **Stacks=4 Blocks=6** | 7 | 80 | 321 | 7 | 8 | 32 |
| **Stacks=4 Blocks=6** | 8 | 240 | 688 | 8 | 13 | 68 |
| **Stacks=4 Blocks=6** | 7 | 72 | 285 | 7 | 9 | 53 |
| **Stacks=4 Blocks=6** | 8 | 185 | 614 | 7 | 8 | 45 |
| **Stacks=4 Blocks=6** | 11 | 754 | 2144 | 11 | 20 | 73 |
| **Stacks=4 Blocks=6** | 10 | 378 | 1114 | 10 | 12 | 56 |
| **Stacks=4 Blocks=6** | 11 | 1944 | 4329 | 10 | 11 | 50 |
| **Stacks=4 Blocks=6** | 10 | 1078 | 2414 | 10 | 12 | 55 |
| **Stacks=4 Blocks=6** | 11 | 1996 | 4183 | 11 | 21 | 111 |
| **Stacks=4 Blocks=6** | 11 | 1945 | 4164 | 10 | 18 | 109 |
| **Stacks=4 Blocks=6** | 10 | 928 | 2221 | 10 | 26 | 131 |
| **Stacks=4 Blocks=6** | 10 | 928 | 2337 | 10 | 13 | 61 |
| **Stacks=4 Blocks=6** | 9 | 497 | 1312 | 9 | 12 | 65 |
| **Stacks=4 Blocks=6** | 11 | 1950 | 4218 | 10 | 18 | 99 |
| **Stacks=4 Blocks=6** | 10 | 377 | 1196 | 10 | 14 | 63 |
| **Stacks=4 Blocks=6** | 11 | 2291 | 4644 | 12 | 20 | 110 |
| **Stacks=4 Blocks=6** | 11 | 1834 | 4201 | 10 | 12 | 65 |
| **Stacks=4 Blocks=6** | 9 | 577 | 1392 | 9 | 11 | 60 |
| **Mean** | 9.8 | 1160.35 | 2530.85 | 9.65 | 14.4 | 73.7 |

For number of stacks=4 and number of blocks =6, number of Goal Tests for h(1)is reduced by 98% when compared to h(0)while maximum queue size is reduced by 97%.

**Table 1: Comparison with Stack=3, Block=5**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Blocks Out of Place Heuristics: h(0)** | | | **New Heuristics: h(1)** | | |
| **Depth** | **Goal Tests** | **Maximum Queue Size** | **Depth** | **Goal Tests** | **Maximum Queue Size** |
| **Stacks=3 Blocks=5** | 11 | 658 | 519 | 12 | 35 | 60 |
| **Stacks=3 Blocks=5** | 7 | 55 | 84 | 7 | 10 | 22 |
| **Stacks=3 Blocks=5** | 9 | 177 | 200 | 9 | 10 | 20 |
| **Stacks=3 Blocks=5** | 9 | 135 | 190 | 9 | 11 | 17 |
| **Stacks=3 Blocks=5** | 7 | 56 | 82 | 7 | 10 | 22 |
| **Stacks=3 Blocks=5** | 11 | 725 | 505 | 11 | 24 | 48 |
| **Stacks=3 Blocks=5** | 7 | 3 | 49 | 7 | 9 | 15 |
| **Stacks=3 Blocks=5** | 3 | 7 | 9 | 3 | 4 | 8 |
| **Stacks=3 Blocks=5** | 7 | 65 | 92 | 7 | 13 | 31 |
| **Stacks=3 Blocks=5** | 8 | 97 | 113 | 8 | 10 | 18 |
| **Stacks=3 Blocks=5** | 9 | 222 | 212 | 9 | 13 | 27 |
| **Stacks=3 Blocks=5** | 8 | 76 | 117 | 8 | 10 | 16 |
| **Stacks=3 Blocks=5** | 9 | 207 | 234 | 9 | 11 | 19 |
| **Stacks=3 Blocks=5** | 6 | 25 | 42 | 6 | 8 | 16 |
| **Stacks=3 Blocks=5** | 9 | 102 | 164 | 9 | 13 | 21 |
| **Stacks=3 Blocks=5** | 9 | 131 | 184 | 9 | 13 | 21 |
| **Stacks=3 Blocks=5** | 9 | 91 | 142 | 9 | 10 | 10 |
| **Stacks=3 Blocks=5** | 9 | 162 | 177 | 11 | 16 | 37 |
| **Stacks=3 Blocks=5** | 4 | 10 | 20 | 4 | 5 | 11 |
| **Stacks=3 Blocks=5** | 3 | 5 | 9 | 3 | 4 | 8 |
| **Mean** | 7.7 | 150.45 | 157.2 | 7.85 | 11.95 | 22.35 |

For different set of values for stack and blocks averaged over 20 runs, number of Goal Tests for h(1)is reduced by 92% when compared to h(0)while maximum queue size is reduced by 86%.

**Table 2: Reports of Failure**

|  |  |  |
| --- | --- | --- |
|  | **Blocks Out of Place Heuristics: h(0)** | **New Heuristics: h(1)** |
| **Stacks=15**  **Blocks=25** | Failure | Success with  goal tests: 53, depth =40 |
| **Stacks=10**  **Blocks=20** | Failure | Success with  goal tests: 85, depth = 35 |
| **Stacks=5**  **Blocks=10** | Failure | Success with  goal tests: 35, depth = 16 |
| **Stacks=4**  **Blocks=7** | Failure | Success with  goal tests=20, depth =13 |
| **Stacks=4**  **Blocks=7** | Failure | Success with  goal tests=12 , depth = 11 |

**EXAMPLE RUN 1:**

Input the number of stacks required

4

Input the total number of elements required

6

Initial state:

Stack 1 :

Stack 2 : 5,

Stack 3 : 2, 6,

Stack 4 : 1, 3, 4,

Goal state:

Stack 1 : 1, 2, 3, 4, 5, 6,

Stack 2 :

Stack 3 :

Stack 4 :

Select a number to select the heuristics to reach the goal state

1. Number of blocks out of place h(0)

2. New heuristics h(1)

3. Enter any other key to Exit

1

goaltest :1

goaltest :2

goaltest :3

goaltest :4

….

….

….

goaltest :170

goaltest :171

goaltest :172

Solution found

Stack 1 :

Stack 2 : 5,

Stack 3 : 2, 6,

Stack 4 : 1, 3, 4,

Stack 1 :

Stack 2 : 5, 4,

Stack 3 : 2, 6,

Stack 4 : 1, 3,

Stack 1 :

Stack 2 : 5, 4, 3,

Stack 3 : 2, 6,

Stack 4 : 1,

Stack 1 : 1,

Stack 2 : 5, 4, 3,

Stack 3 : 2, 6,

Stack 4 :

Stack 1 : 1,

Stack 2 : 5, 4, 3,

Stack 3 : 2,

Stack 4 : 6,

Stack 1 : 1, 2,

Stack 2 : 5, 4, 3,

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3,

Stack 2 : 5, 4,

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3, 4,

Stack 2 : 5,

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3, 4, 5,

Stack 2 :

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3, 4, 5, 6,

Stack 2 :

Stack 3 :

Stack 4 :

Solution at depth: 9

No of goal tests: 172

Maximum Queue Size 586

Select a number to select the heuristics to reach the goal state

1. Number of blocks out of place h(0)

2. New heuristics h(1)

3. Enter any other key to Exit

2

goaltest :1

goaltest :2

goaltest :3

goaltest :4

goaltest :5

goaltest :6

goaltest :7

goaltest :8

goaltest :9

goaltest :10

Solution found

Stack 1 :

Stack 2 : 5,

Stack 3 : 2, 6,

Stack 4 : 1, 3, 4,

Stack 1 :

Stack 2 : 5, 4,

Stack 3 : 2, 6,

Stack 4 : 1, 3,

Stack 1 :

Stack 2 : 5, 4, 3,

Stack 3 : 2, 6,

Stack 4 : 1,

Stack 1 : 1,

Stack 2 : 5, 4, 3,

Stack 3 : 2, 6,

Stack 4 :

Stack 1 : 1,

Stack 2 : 5, 4, 3,

Stack 3 : 2,

Stack 4 : 6,

Stack 1 : 1, 2,

Stack 2 : 5, 4, 3,

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3,

Stack 2 : 5, 4,

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3, 4,

Stack 2 : 5,

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3, 4, 5,

Stack 2 :

Stack 3 :

Stack 4 : 6,

Stack 1 : 1, 2, 3, 4, 5, 6,

Stack 2 :

Stack 3 :

Stack 4 :

Solution at depth: 9

No of goal tests: 10

Maximum Queue Size 49

Select a number to select the heuristics to reach the goal state

1. Number of blocks out of place h(0)

2. New heuristics h(1)

3. Enter any other key to Exit

3

**EXAMPLE RUN 2:** FOR HARDER PROBLEMS: h(0) fails, h(1) is successful

Input the number of stacks required

10

Input the total number of elements required

15

Initial state:

Stack 1 : 2,

Stack 2 : 4, 13,

Stack 3 : 6, 10,

Stack 4 : 9,

Stack 5 : 14, 15,

Stack 6 : 7, 12,

Stack 7 : 1, 3, 8,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Goal state:

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,

Stack 2 :

Stack 3 :

Stack 4 :

Stack 5 :

Stack 6 :

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 :

Select a number to select the heuristics to reach the goal state

1. Number of blocks out of place h(0)

2. New heuristics h(1)

3. Enter any other key to Exit

2

goaltest :1

goaltest :2

…….

…….

goaltest :26

goaltest :27

goaltest :28

Solution found

Stack 1 : 2,

Stack 2 : 4, 13,

Stack 3 : 6, 10,

Stack 4 : 9,

Stack 5 : 14, 15,

Stack 6 : 7, 12,

Stack 7 : 1, 3, 8,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 :

Stack 2 : 4, 13, 2,

Stack 3 : 6, 10,

Stack 4 : 9,

Stack 5 : 14, 15,

Stack 6 : 7, 12,

Stack 7 : 1, 3, 8,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 :

Stack 2 : 4, 13, 2,

Stack 3 : 6,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12,

Stack 7 : 1, 3, 8,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 :

Stack 2 : 4, 13, 2,

Stack 3 :

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 1, 3, 8,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 :

Stack 2 : 4, 13, 2,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 1, 3,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 :

Stack 2 : 4, 13, 2,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 1,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11, 3,

Stack 1 : 1,

Stack 2 : 4, 13, 2,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 : 5, 11, 3,

Stack 1 : 1, 2,

Stack 2 : 4, 13,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 : 5, 11, 3,

Stack 1 : 1, 2, 3,

Stack 2 : 4, 13,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 : 1, 2, 3,

Stack 2 : 4,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 13,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 : 1, 2, 3, 4,

Stack 2 :

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 13,

Stack 8 :

Stack 9 :

Stack 10 : 5, 11,

Stack 1 : 1, 2, 3, 4,

Stack 2 : 11,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 13,

Stack 8 :

Stack 9 :

Stack 10 : 5,

Stack 1 : 1, 2, 3, 4, 5,

Stack 2 : 11,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12, 6,

Stack 7 : 13,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6,

Stack 2 : 11,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7, 12,

Stack 7 : 13,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6,

Stack 2 : 11,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 : 7,

Stack 7 : 13, 12,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7,

Stack 2 : 11,

Stack 3 : 8,

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 :

Stack 7 : 13, 12,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8,

Stack 2 : 11,

Stack 3 :

Stack 4 : 9,

Stack 5 : 14, 15, 10,

Stack 6 :

Stack 7 : 13, 12,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9,

Stack 2 : 11,

Stack 3 :

Stack 4 :

Stack 5 : 14, 15, 10,

Stack 6 :

Stack 7 : 13, 12,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

Stack 2 : 11,

Stack 3 :

Stack 4 :

Stack 5 : 14, 15,

Stack 6 :

Stack 7 : 13, 12,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,

Stack 2 :

Stack 3 :

Stack 4 :

Stack 5 : 14, 15,

Stack 6 :

Stack 7 : 13, 12,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,

Stack 2 :

Stack 3 :

Stack 4 :

Stack 5 : 14, 15,

Stack 6 :

Stack 7 : 13,

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,

Stack 2 :

Stack 3 :

Stack 4 :

Stack 5 : 14, 15,

Stack 6 :

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,

Stack 2 : 15,

Stack 3 :

Stack 4 :

Stack 5 : 14,

Stack 6 :

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,

Stack 2 : 15,

Stack 3 :

Stack 4 :

Stack 5 :

Stack 6 :

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 :

Stack 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,

Stack 2 :

Stack 3 :

Stack 4 :

Stack 5 :

Stack 6 :

Stack 7 :

Stack 8 :

Stack 9 :

Stack 10 :

Solution at depth: 24

No of goal tests: 28

Maximum Queue Size 1192

Select a number to select the heuristics to reach the goal state

1. Number of blocks out of place h(0)

2. New heuristics h(1)

3. Enter any other key to Exit

3