Assignment 2

Name: Vishwas Siravara
UIN: 623009600

Performance Metrics

	Number of Stacks	Number of Blocks	Stack Config	Simple Heuristic	My Heuristic
	3	5	BC AD E		
Running Time				0.032	0.0178561211
Iterations				101	39
Queue Size				121	50
Solution Depth				8	8
	3	7	FC GE ABD		
Running Time				0.1389	0.0372548103
Iterations				507	137
Queue Size				761	180
Solution Depth				12	12
	4	7	Empty E BFD ACG		
Running Time				14.2770009041	3.7684290409
Iterations				2992	1301
Queue Size				7906	3277
solution Depth				12	12
	5	7	F Empty GC AD BE		
Running Time				9.2030698586	0.7649800777
Iterations				1341	373
Queue Size				6350	1892
Solution Depth				10	10
	6	8	D GC A BF EH Empty		
Running Time			0.0149929523	324.757354975	6.6580150127
Iterations			268	4422	702
Queue Size			22	30934	5974
Solution Depth			30	11	11

Logic behind my heuristic:
I have used a min heap for my priority queue , hence lower score gets higher priority. In the simple heuristic(number of blocks out of place) we do not consider whether the blocks are continuous or whether the elements out of place on top of stack has any significance. I have designed a heuristic which considers whether elements are continuously in sequence and the effect of number of blocks out of sequence on stack one.

For example for 2 configurations
Configuration 1

Stack1: AB
Stack2: ED
Stack3:C

Configuration 2:
Stack1: ABD
Stack2: E
Stack3: C

my heuristic assigns a lower score for configuration 1 , hence it gets higher priority . The way I have implemented it is Priority = depth of node (from start state) + score

score = Num of blocks – number of continuous blocks on stack 1 from from index 0 + number of blocks out of place on stack1

For example config 1 will get a score of (5 – 2 + 0) = 3 and config 2 will get a score of (5 -2 + 1)= 4
Assuming both the configurations occur at the same depth , the algorithm choses config 1 over config 2.

On average this heuristic gives me an improvement in the factor of 10 when compared to the naive heuristic .

This heuristic is admissible because it confers to the triangle rule of inequality.

For example if parent state is :

Stack1: ABD
Stack2: E
Stack3: C

For the best case the child will be
Stack1 : AB
Stack2: ED
Stack3: C

score(parent) <= score(child) + depth

For parent we get a score of 4 and for the child state we get a score of 3 and a heuristic value of 4 since we add the depth .

Hence it obeys the triangle rule of inequality and is therefore admissible.

My program terminates after 10000 iterations if goal state is not found and it was not able to solve the 10 block 5 stack problem ie

Stack1:D
Stack2:EFIJ
Stack3:BG
Stack4:CH
Stack5:A

Program Traces :