heuristic_analysis

Table of Contents

1 Heuristic Analysis

1.1 Metrics for non-heuristic planning solution searches

1.1.1 Problem 1

Search	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed	Optimal
$breadth_{firstsearch}$	43	56	180	6	0.04683144300361164	Yes
$depth_{firstgraphsearch}$	21	22	84	20	0.01650939101818949	No
$uniform_{costsearch}$	55	57	224	6	0.05334271999890916	Yes

1.1.2 Problem 2

Search	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed	Optimal
$breadth_{firstsearch}$	3343	4609	30509	9	17.596341395023046	Yes
$depth_{firstgraphsearch}$	624	625	5602	619	4.402504776982823	No
$uniform_{costsearch}$	4853	4855	44041	9	14.847201219003182	Yes

1.1.3 Problem 3

Search	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed	Optimal
$breadth_{firstsearch}$	8602	11196	64308	12	63.57823093398474	Yes
$depth_{firstgraphsearch}$	1292	1293	5744	875	5.141792179987533	No
$uniform_{costsearch}$	11482	11484	85785	12	47.939871934009716	Yes

BFS and UCS gives us the optimal and complete solution. DFS while being fastest for this small problem and more efficient in term of resources gives us a non optimal solution.

1.2 Metrics for non-heuristic planning solution searches

1.2.1 Problem 1

Heuristic	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
h_1	55	57	224	6	0.04679326497716829
$h_{ignorepreconditions}$	41	43	170	6	0.05496499201399274
$h_{pglevelsum}$	39	41	158	6	1.2185538750200067

1.2.2 Problem 2

Heuristic	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
h_1	4853	4855	44041	9	17.488993248000043
$h_{ignorepreconditions}$	1450	1452	13303	9	6.052451260999078
$h_{pglevelsum}$	1129	1131	10232	9	532.737853553990

1.2.3 Problem 3

Heuristic	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
h_1	11482	11484	85785	12	52.21306998498039
$h_{ignorepreconditions}$	4119	4121	31485	12	22.876345383003354
$h_{pglevelsum}$	1958	1960	15626	12	1196.0613465080096

$h_{ignorepreconditions}$ heuristic provides results in least time. It expands less nodes than h_1. $h_{pglevelsum}$ provides results in slowest time. A* search gives optimal results, so these results were optimal.

1.3 Optimal Plans for Problems

1.3.1 Problem 1

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)
3. Fly(P2, JFK, SFO)
4. Unload(C2, P2, SFO)
5. Fly(P1, SFO, JFK)
6. Unload(C1, P1, JFK)

1.3.2 Problem 2

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)
3. Load(C3, P3, ATL)
4. Fly(P2, JFK, SFO)
5. Unload(C2, P2, SFO)
6. Fly(P1, SFO, JFK)
7. Unload(C1, P1, JFK)
8. Fly(P3, ATL, SFO)
9. Unload(C3, P3, SFO)

1.3.3 Problem 3

1. Load(C1, P1, SFO)
2. Fly(P1, SFO, ATL)
3. Load(C3, P1, ATL)
4. Fly(P1, ATL, JFK)
5. Load(C2, P1, JFK)
6. Unload(C1, P1, JFK)
7. Unload(C3, P1, JFK)
8. Fly(P1, JFK, ORD)
9. Load(C4, P1, ORD)
10. Fly(P1, ORD, SFO)
11. Unload(C2, P1, SFO)
12. Unload(C4, P1, SFO)

Author: Khurram Baig
Created: 2017-04-25 Tue 01:30
Emacs 25.1.1 (Org mode 8.2.10)