

HEURISTIC ANALYSIS

Problems Definition & Result Matrix:

- Air Cargo Action Schema:

...

Action(Load(c, p, a),

PRECOND: $At(c, a) \wedge At(p, a) \wedge Cargo(c) \wedge Plane(p) \wedge Airport(a)$

EFFECT: $\neg At(c, a) \wedge In(c, p)$)

Action(Unload(c, p, a),

PRECOND: $In(c, p) \wedge At(p, a) \wedge Cargo(c) \wedge Plane(p) \wedge Airport(a)$

EFFECT: $At(c, a) \wedge \neg In(c, p)$)

Action(Fly($p, from, to$),

PRECOND: $At(p, from) \wedge Plane(p) \wedge Airport(from) \wedge Airport(to)$

EFFECT: $\neg At(p, from) \wedge At(p, to)$)

...

- Problem 1 initial state and goal:

...

Init($At(C1, SFO) \wedge At(C2, JFK)$

$\wedge At(P1, SFO) \wedge At(P2, JFK)$

$\wedge Cargo(C1) \wedge Cargo(C2)$

$\wedge Plane(P1) \wedge Plane(P2)$

$\wedge Airport(JFK) \wedge Airport(SFO))$

Goal($At(C1, JFK) \wedge At(C2, SFO)$)

...

Optimal Plan:

Load($C1, P1, SFO$)

Load($C2, P2, JFK$)

Fly($P2, JFK, SFO$)

Unload($C2, P2, SFO$)

Fly($P1, SFO, JFK$)

Unload($C1, P1, JFK$)

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
breadth_first_search	Yes	6	0.052	180	43	56
depth_first_graph_search	No	20	0.029	84	21	22
greedy_best_first_graph_search h_1	Yes	6	0.01	28	7	9

In Problem 1, greedy_best_first_graph_search h_1 performs best , highly efficiency and consumed least amount of memory(node expand). BFS optimum result but takes more time and

consume more memory. *Depth_first_graph_search* didn't optimize result but it consume less time and memory than *BFS*.

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
astar_search h_1	Yes	6	0.061	224	55	57
astar_search h_ignore_pre conditions	Yes	6	0.059	43	41	170

- Problem 2 initial state and goal:

...

$Init(At(C1, SFO) \wedge At(C2, JFK) \wedge At(C3, ATL)$
 $\wedge At(P1, SFO) \wedge At(P2, JFK) \wedge At(P3, ATL)$
 $\wedge Cargo(C1) \wedge Cargo(C2) \wedge Cargo(C3)$
 $\wedge Plane(P1) \wedge Plane(P2) \wedge Plane(P3)$
 $\wedge Airport(JFK) \wedge Airport(SFO) \wedge Airport(ATL))$
 $Goal(At(C1, JFK) \wedge At(C2, SFO) \wedge At(C3, SFO))$

Optimal Plan

$Load(C1, P1, SFO)$
 $Load(C2, P2, JFK)$
 $Load(C3, P3, ATL)$
 $Fly(P2, JFK, SFO)$
 $Unload(C2, P2, SFO)$
 $Fly(P1, SFO, JFK)$
 $Unload(C1, P1, JFK)$
 $Fly(P3, ATL, SFO)$
 $Unload(C3, P3, SFO)$

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
breadth_first_search	Yes	9	11.86	30509	3343	4609
depth_first_graph_search	No	619	4.99	5602	624	625
greedy_best_first_graph_search h_1	No	17	3.40	8910	990	992

The table shows *depth_first_graph_search* and *greedy_best_first_graph_search h_1* have no optimal result, execute quickly and consume less memory. *depth_first_graph_search* output large

plane Length. Breadth_first_search reach optimal solution and the Node expand more than two other algorithms.

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
astar_search h_1	Yes	9	16.82	44030	4852	16.82
astar_search h_ignore_preconditions	Yes	9	6.084	13303	1450	1452

'''

- Problem 3 initial state and goal:

'''

$Init(At(C1, SFO) \wedge At(C2, JFK) \wedge At(C3, ATL) \wedge At(C4, ORD)$
 $\wedge At(P1, SFO) \wedge At(P2, JFK)$
 $\wedge Cargo(C1) \wedge Cargo(C2) \wedge Cargo(C3) \wedge Cargo(C4)$
 $\wedge Plane(P1) \wedge Plane(P2)$
 $\wedge Airport(JFK) \wedge Airport(SFO) \wedge Airport(ATL) \wedge Airport(ORD))$
 $Goal(At(C1, JFK) \wedge At(C3, JFK) \wedge At(C2, SFO) \wedge At(C4, SFO))$
 '''

Optimal Plan:

Load(C1, P1, SFO)
 Load(C2, P2, JFK)
 Fly(P2, JFK, ORD)
 Load(C4, P2, ORD)
 Fly(P1, SFO, ATL)
 Load(C3, P1, ATL)
 Fly(P1, ATL, JFK)
 Unload(C1, P1, JFK)
 Unload(C3, P1, JFK)
 Fly(P2, ORD, SFO)
 Unload(C2, P2, SFO)
 Unload(C4, P2, SFO)

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
breadth_first_search	Yes	12	59.71	129631	14663	18098
depth_first_graph_search	No	392	2.47	3364	408	409

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
greedy_best_first_graph_search h_1	No	22	25.08	49429	5614	5616

The table shows again depth_first_graph_search and greedy_best_first_graph_search h_1 have no optimal result but execute quickly and consume less memory. depth_first_graph_search output large plane Length. Breadth_first_search reach optimal solution and the Node expand more than two other algorithms.

Search Method	Optimality	Plane Length	Time Elapsed	New nodes	# Node Expand	Goal Tests
astar_search h_1	Yes	12	77.17	159716	18235	18237
astar_search h_ignore_pre conditions	Yes	12	25.27	4494	5040	5042

References:

<http://aima.cs.berkeley.edu/2nd-ed/newchap11.pdf>