Planning Search Written Analysis

In the context of the AIND Planning Search Heuristic project, multiple search algorithms were tested using 3 problems on the Air Cargo scenario. These tests were done over a Ubuntu 14 virtual machine with 2 GB of RAM and 2 processor cores.

Planning Problem Representation

The following 3 problems where given to test the search algorithms:

Problem 1

$Init(At(C1, SFO)$

$\wedge\ At(C2, JFK) \wedge\ At(P1, SFO) \wedge\ At(P2, JFK)$

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

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

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

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

Problem 2

$Init(At(C1, SFO)$

$\wedge\ At(C2, JFK) \wedge\ At(C3, ATL) \wedge\ At(P1, SFO) \wedge\ At(P2, JFK) \wedge\ At(P3, ATL)$

$\wedge\ Cargo(C1) \wedge\ Cargo(C2) \wedge\ Cargo(C3)$

$\wedge\ Plane(P1) \wedge\ Plane(P2) \wedge\ Plane(P3)$

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

$Goal(At(C1, JFK)\ \wedge\ At(C2, SFO)\ \wedge\ At(C3, SFO))$

Problem 3

$Init(At(C1, SFO)$

$\wedge\ At(C2, JFK) \wedge\ At(C3, ATL) \wedge\ At(C4, ORD) \wedge\ At(P1, SFO) \wedge\ At(P2, JFK)$

$\wedge\ Cargo(C1) \wedge\ Cargo(C2) \wedge\ Cargo(C3) \wedge\ Cargo(C4)$

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

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

$Goal(At(C1, JFK)\ \wedge\ At(C3, JFK)\ \wedge\ At(C2, SFO)\ \wedge\ At(C4, SFO))$

Optimal Sequence of Actions

To select the optimal sequence of actions for each problem, the performance of the algorithms was evaluated using the time, space, optimality and completeness criteria described in the lectures. As a consequence, the following measures were taken after each algorithm run:

- Number of node expansions: The least the better, as it will take less space.
- Execution time: time taken to reach the goal. The lest, the better.
- Number of actions taken: the least actions taken the better.
- Completeness: defines if reaches the goal (Boolean value). Problems that took more than 10 minutes are not complete. If it didn't reach the goal, a "- "sign can be found in the table.

The different paths given by the algorithms can be found in the *Analysis.xlsx* file in the root of this project.

Algorithm/ Problem	P1											
Measure	Expansions	Goal Tests	New Nodes	Time(s)	Expansions	Goal Tests	New Nodes	Time(s)	Expansions	Goal Tests	New Nodes	Time(s)
BFS	43	56	180	0.059343	3343	4609	30509	17.31	14663	18098	129631	127.23
Breadth_first_tree_search	1458	1459	5960	1.106	-	-	-	-	-	-	-	-
depth_first_graph_search	12	13	48	0.017	582	583	5211	4.52	627	628	5176	4.45
depth_limited_search	101	271	414	0.1849	222719	2053741	2054119	1382	-	-	-	-
uniform_cost_search	55	57	224	0.1289	4853	4855	44041	50.723	18223	18225	159618	510.634
recursive_best_first_search h_1	4229	4230	17029	3.202	69347731	69347732	625574973	190736.109	-	-	-	-
greedy_best_first_graph_search h_1	7	9	28	0.00688	998	1000	8982	9.0374	5578	5580	49150	159.348
astar_search h_1	55	57	224	0.1093	4853	4855	44041	53.019	18223	18225	159618	683.3317
astar_search h_ignore_preconditions	55	57	224	0.13328	4853	4855	44041	61.241	18223	18225	159618	686.491
astar_search h_pg_levelsum	41	43	167	14.97	2927	2929	26804	35171	-	-	-	-

Figure 1 Algorithms run results

Given the results, the following optimal sequence of actions (OSA) were selected based on an analysis where the objective is to minimize the number of actions, nodes expanded and time taken for each algorithm:

Problem 1

$Load(C1, P1, SFO)$

$Load(C2, P2, JFK)$

$Fly(P1, SFO, JFK)$

$Fly(P2, JFK, SFO)$

$Unload(C1, P1, JFK)$

$Unload(C2, P2, SFO)$

For problem 1, choosing the OSA is easy. Even if 8 algorithms give results with 9 actions, the SA given by *greedy best first graph search* h_1 takes only 0.00688 seconds with 7 nodes. It's the minimum amount of time, space, and actions needed to achieve the goal, between all the available options.

Problem 2

$Load(C2, P2, JFK)$

$Load(C1, P1, SFO)$

$Load(C3, P3, ATL)$

$Fly(P2, JFK, SFO)$

$Unload(C2, P2, SFO)$

$Fly(P1, SFO, JFK)$

$Unload(C1, P1, JFK)$

$Fly(P3, ATL, SFO)$

$Unload(C3, P3, SFO)$

Problem 2 is more complicated, as there are 6 algorithms that take 9 actions. The SA given by *A* search* levelsum shows the least amount of expanded nodes, which is space efficient, but takes 35171 seconds (or roughly 9.7 hours) to run so it's very time inefficient. The result given by a simple *BFS* gives the least amount of time between all algorithms with a 17.31 execution time by expanding 3343 nodes, which would be the minimum of all available results ignoring *A* search levelsum* node expansion.

Problem 3

$Load(C2, P2, JFK)$

$Fly(P2, JFK, ORD)$

$Load(C4, P2, ORD)$

$Fly(P2, ORD, SFO)$

$Unload(C4, P2, SFO)$

$Load(C1, P1, SFO)$

$Fly(P1, SFO, ATL)$

$Load(C3, P1, ATL)$

$Fly(P1, ATL, JFK)$

$Unload(C3, P1, JFK)$

$Unload(C1, P1, JFK)$

$Unload(C2, P2, SFO)$

For problem 3, there are 4 algorithms that take 12 actions. However, choosing the optimal SA is quite easy considering that *A* Ignore Preconditions* takes 125.953 seconds vs 127.23, 510.63, and 683.33 given by the other algorithms with 12 actions and expands 5118 nodes, less than 1/3 of the second best option. Nevertheless, it's important to mention that *depth first graph search* found the SA with the least amount of nodes expanded and in the shortest time but with more than 30 actions to take, which isn't optimal according to the definition of the problem.

Performance Comparison

A comparison of planning algorithms and heuristics using as metrics the optimality of the solutions (Boolean), time elapsed (seconds), and the number of node expansions required, is presented below.

Uninformed planning algorithms

Algorithm/ Problem	P1			P2			P3		
Measure	**Optimal?**	**Expansions**	**Time(s)**	**Optimal?**	**Expansions**	**Time(s)**	**Optimal?**	**Expansions**	**Time(s)**
BFS	yes	43	0.059343	yes	3343	17.31	yes	14663	127.23
Breadth_first_tree_search	yes	1458	1.106	**no**	-	-	**no**	-	-
depth_first_graph_search	yes	12	0.017	yes	582	4.52	yes	627	4.45
depth_limited_search	yes	101	0.1849	**no ***	222719	1382	**no**	-	-
uniform_cost_search	yes	55	0.1289	yes	4853	50.723	yes	18223	510.634
recursive_best_first_search h_1	yes	4229	3.202	**no ***	69347731	190736.109	**no**	-	-
greedy_best_first_graph_search h_1	yes	7	0.00688	yes	998	9.0374	yes	5578	159.348
***Note: didn't finished under 10 min, which makes them "Non Optimal" according to the asssumptions of the project.**									

Figure 2 Metrics over Uninformed algorithms per problem

From the given results, it's clear that all the algorithms are optimal for P1. However, for P2 and P3, it can be observed that *breadth first tree search*, *depth limited search* and *recursive best first search* aren't optimal, as there are problems where they never reach a solution under 10 min. It can be observed that *depth first graph search* is the fastest algorithm that expands the least amount of nodes to reach a solution, which is important to consider for problems where space optimality and speed are crucial. However, it delivered an OAS with so many nodes that it was difficult to count them, which is not optimal. Leaving *depth first graph search* aside, it can be observed that time wise, *BFS*, *greedy best first graph search,* and *uniform cost search* are the algorithms that deliver a OAS with the given restrictions (being more time effective in the mentioned order) while in number of expansion nodes required, *greedy best first graph search* takes *BFS* first place with 1/3 of nodes expanded. The 3 algorithms deliver an OAS of 9 actions.

Automatic heuristics

Algorithm/ Problem	P1			P2			P3		
Measure	**Optimal?**	**Expansions**	**Time(s)**	**Optimal?**	**Expansions**	**Time(s)**	**Optimal?**	**Expansions**	**Time(s)**
astar_search h_1	yes	55	0.1093	yes	4853	53.019	yes	18223	683.3317
astar_search h_ignore_preconditions	yes	41	0.07803	yes	1506	19.479	yes	5118	125.953
astar_search h_pg_levelsum	yes	41	14.97	**no ***	2927	35171	**no**	-	-
***Note: didn't finished under 10 min, which makes them "Non Optimal" according to the asssumptions of the project.**									

From the given result, it's clear that all Heuristics are optimal for P1. However, for P2 and P3, it can be observed that *A* search with level sum* is not optimal. This behaviour was expected as this heuristic sums all the available actions to reach the goal, which exceeds the 10 min limit by far for P3. For all the problems, it can be observed that *A* Search Ignore Preconditions* excels vs the other heuristics as it not only more efficient in terms of space and time, but it also gives the best OAS, with only 12 moves. This was expected as it was explained in the lectures that relaxing the constraints reduces the search space greatly.