Planning Agent Final Report

In this project we focused on the problem of solving a deterministic logistics planning problem for Air Cargo Transport System. Our search and planning agent efficiently plans a strategy for this process.

I have experimented several search algorithms and heuristics on four different types of Air Cargo problem 1, 2, 3, 4 respectively.
There are total 11 search algorithms that are used in the analysis presented below tested on 4 different problems related to Air Cargo Transportation.

Problem Definition
We are given three different types of planning problems that need to be solved using the same action schema.

Action(Fly(p, from,to),
PRECOND: At(p, from) ∧ Plane(p) ∧ Airport(from) ∧ Airport(to)
EFFECT: ¬At(p, from) ∧ At(p,to))
Action(Load(c, p, a),
PRECOND: At(c, a) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
EFFECT: ¬ At(c, a) ∧ In(c, p))
Action(Unload(c, p, a),
PRECOND: In(c, p) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
EFFECT: At(c, a) ∧ ¬ In(c, p))

Analysis on problem 1, 2, 3, 4

Problem 1
The optimal plan is of length 6 for this problem.

Search Algorithm	Optimal	Expansions	Path Length	Execution Time (s)
Breadth First Search	Yes	43	6	0.00594411
Depth First Graph Search	No	21	20	0.00303032
Uniform cost Search	Yes	60	6	0.00894590
Greedy Best First Graph Search 1	Yes	7	6	0.00164158
Greedy Best First Graph Search 2	Yes	6	6	0.41193465
Greedy Best First Graph Search 3	Yes	6	6	0.29929283

Greedy Best First Graph Search 4	Yes	6	6	0.50776247
A Star Search 1	Yes	50	6	0.00890677
A Star Search 2	Yes	28	6	1.02987356
A Star Search 3	Yes	43	6	1.05396838
A Star Search 4	Yes	33	6	1.21358165

Problem 2
Optimal plan length = 9

Search Algorithm	**Optimal**	**Expansions**	**Path Length**	**Execution Time (s)**
Breadth First Search	Yes	3343	9	1.81303950
Depth First Search	No	624	619	2.78556195
Uniform Cost Search	Yes	5154	9	3.08115778
Greedy Best First Search 1	Yes	17	9	0.01767745
Greedy Best First Search 2	Yes	9	9	9.00224264
Greedy Best First Search 3	Yes	27	9	18.50740043
Greedy Best First Search 4	Yes	9	9	12.47610014
A Star Search 1	Yes	2467	9	2.095102238
A Star Search 2	Yes	357	9	227.82771191
A Star Search 3	Yes	2887	9	1228.34163389
A Star Search 4	Yes	1037	9	1042.03801595

Problem 3
Optimal plan length = 12

Search Algorithm	Optimal	Expansions	Path Length	Execution Time (s)
Breadth First Search	Yes	14663	12	8.53046281
Depth First Search	No	408	392	0.98303380
Uniform Cost Search	Yes	18512	12	11.76837428
Greedy Best First Search 1	No	25	15	0.03065162
Greedy Best First Search 2	No	14	14	18.51682551
A Star Search 1	Yes	7388	12	6.90584898
A Star Search 2	Yes	369	12	339.75939135

Problem 4
Optimal plan length = 14

Search Algorithm	Optimal	Expansions	Path Length	Execution Time (s)
Breadth First Search	Yes	99736	14	76.85300303
Depth First Search	-	-	-	-
Uniform Cost Search	Yes	113339	14	93..64420320
Greedy Best First Search 1	No	29	18	0.04785316
Greedy Best First Search 2	No	17	17	33.73382394
A Star Search 1	Yes	34330	14	44.22115365
A Star Search 2	No	1208	15	1877.04277142

Number of nodes expanded vs Number of Actions

Algorithm	Action- problem 1	Nodes-problem 1	Action-problem 2	Action-problem 2
BFS	20	43	72	3343
DFS	20	21	72	624
UCS	20	60	72	5154
GBFS 1	20	7	72	17
GBFS 2	20	6	72	9
GBFS 3	20	6	72	27
GBFS 4	20	6	72	9
A* Search 1	20	50	72	2467
A* Search 2	20	28	72	357
A* Search 3	20	43	72	2887
A* Search 4	20	33	72	1037

Algorithm	Action-problem 3	Nodes-problem 3	Action-problem 4	Nodes-problem 4
BFS	88	14663	104	99736
DFS	88	408	104	-
UCS	88	18512	104	113339
GBFS 1	88	25	104	29
GBFS 2	88	14	104	17
A* Search 1	88	7388	104	34330
A* Search 2	88	369	104	1208

Analysis - The amount of nodes expanded increases as the number of actions for the problem increases. The number of nodes expand for each next problem. However the increase is observed to be different depending on search algorithms and heuristics. In case of uninformed search like BFS, UCS, DFS and A* search, the amount of nodes significantly increases as the number of actions increases. However the amount of increase in case of Greedy Best First Search algorithms is slightly slow but it also increases with actions.

Search Time vs Number of Actions

Algorithm	Search Time 1	Actions Problem 1	Search Time 2	Actions Problem 2
BFS	0.00594411	20	1.81303950	72
DFS	0.00303032	20	2.78556195	72
UCS	0.00894590	20	3.08115778	72
GBFS 1	0.00164158	20	0.01767745	72
GBFS 2	0.41193465	20	9.00224264	72
GBFS 3	0.29929283	20	18.50740043	72
GBFS 4	0.50776247	20	12.47610014	72
A* Search 1	0.00890677	20	2.095102238	72
A* Search 2	1.02987356	20	227.82771191	72
A* Search 3	1.05396838	20	1228.34163389	72
A* Search 4	1.21358165	20	1042.03801595	72

Algorithm	Search Time 3	Actions Problem 3	Search Time 4	Actions Problem 4
BFS	8.53046281	88	76.85300303	104
DFS	0.98303380	88	-	104
UCS	11.76837428	88	93..64420320	104
GBFS 1	0.03065162	88	0.04785316	104
GBFS 2	18.51682551	88	33.73382394	104
A* Search 1	6.90584898	88	44.22115365	104
A* Search 2	339.75939135	88	1877.04277142	104

Analysis - Search time also increase as the number of actions increases corresponding to the different problems. The number of actions for each problem increases as the complexity of the problem increases. Run time increases greatly for A* search which is way more than other algorithms like BFS, DFS, UCS, GBFS etc.

Complexity Analysis

Greedy Best First Search 1, 2 uses less memory than Depth first Search. BFS uses less memory than Uniform Cost Search. In case of larger domains A* search uses less memory than BFS and DFS..

Search Time Analysis

Of all the strategies, Breadth first search always finds the optimal path with quite less search time. However as the size of the problem domain increases, Greedy Best first search 1 algorithm outperforms BFS with lesser search time. Depth First Search also needs less time to search however the path planning is not optimal just like Greedy best first search 1.

Questions

Ques 1. Which algorithm or algorithms would be most appropriate for planning in a very restricted domain (i.e., one that has only a few actions) and needs to operate in real time?

Ans) In a very restricted domain, algorithm 4 (greedy_best_first_graph_search h_unmet_goals) was shown to find near-optimal plans within a millisecond.

Ques 2. Which algorithm or algorithms would be most appropriate for planning in very large domains (e.g., planning delivery routes for all UPS drivers in the U.S. on a given day)

Ans) An example of larger domain would be Air Cargo problem 4. In this case Greedy Best First Search performs much better than other algorithms..

Ques 3. Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?

Ans) Algorithms 8-11 (astar_search with multiple heuristics) would be the most appropriate in this case, since A* is guaranteed to find only optimal place if we have an admissible heuristic