Report for Forward-Planning agent

By @Divya Amin

Introduction

In this report we focused on solving a deterministic logistics planning problem for an air cargo transport system by implementing a search and planning agent.

In this project, we were given 4 problems, each having different numbers of aeroplanes, cargo items, and airports and as a result, varying complexities. There are 11 search algorithms given to us to experiment on these 4 problems. They are as follows:

1. Breadth First Search
2. Depth First Graph Search
3. Uniform Cost Search
4. Greedy Best First Graph Search with heuristic h_unmet_goals
5. Greedy Best First Graph Search with heuristic h_pg_levelsum
6. Greedy Best First Graph Search with heuristic h_pg_maxlevel
7. Greedy Best First Graph Search with heuristic h_pg_setlevel
8. A* Search with heuristic h_unmet_goals
9. A* Search with heuristic h_pg_levelsum
10. A* Search with heuristic h_pg_maxlevel
11. A* Search with heuristic h_pg_setlevel

The goal of this project is to perform an analysis of the search algorithms on each of the 4 problems and record the following information about the algorithms:

- number of actions in the domain
- number of new node expansions
- time to complete the plan search

In addition to recording this information, this report seeks to answer the following questions:

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

Problem Description

The 4 problems use the below 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 the problems

Problem 1

Search Algorithm	Actions	Expansions	Path Length	Execution Time
Breadth First Search	20	43	6	0.006757973000048878
Depth First Graph Search	20	21	20	0.003534721000050922
Uniform Cost Search	20	60	6	0.010460765999937394
Greedy Best First Graph Search with heuristic h_unmet_goals	20	7	6	0.0024077629999510464
Greedy Best First Graph Search with heuristic h_pg_levelsum	20	6	6	0.4692074989999355

Search Algorithm	Actions	Expansions	Path Length	Execution Time
Greedy Best First Graph Search with heuristic h_pg_maxlevel	20	6	6	0.35779809600001045
Greedy Best First Graph Search with heuristic h_pg_setlevel	20	6	6	0.6142740919999596
A* Search with heuristic h_unmet_goals	20	50	6	0.009698603000060757
A* Search with heuristic h_pg_levelsum	20	28	6	1.191269531999751
A* Search with heuristic h_pg_maxlevel	20	43	6	1.2406019310001284
A* Search with heuristic h_pg_setlevel	20	33	6	1.4630804359999274

The optimal path length in Problem 1 is 6

Problem 2

Search Algorithm	Actions	Expansions	Path Length	Execution Time
Breadth First Search	72	3343	9	1.9982181989998935
Depth First Graph Search	72	624	619	3.0761934849997488
Uniform Cost Search	72	5154	9	3.3623013939999282
Greedy Best First Graph Search with heuristic h_unmet_goals	72	17	9	0.019361825999567372
Greedy Best First Graph Search with heuristic h_pg_levelsum	72	9	9	10.810248023999975
Greedy Best First Graph Search with heuristic h_pg_maxlevel	72	27	9	21.229660099000284
Greedy Best First Graph Search with heuristic h_pg_setlevel	72	9	9	14.85959442999956
A* Search with heuristic h_unmet_goals	72	2467	9	2.2257588270003907
A* Search with heuristic h_pg_levelsum	72	357	9	267.2171387919998
A* Search with heuristic h_pg_maxlevel	72	2887	9	1543.3977732800004

Search Algorithm	Actions	Expansions	Path Length	Execution Time
A* Search with heuristic h_pg_setlevel	72	1037	9	1473.381522091

The optimal path length in Problem 2 is 9

Problem 3

Search Algorithm	Actions	Expansions	Path Length	Execution Time
Breadth First Search	88	14663	12	10.589345288999993
Depth First Graph Search	88	408	392	1.1678024960001494
Uniform Cost Search	88	18510	12	14.493687367999883
Greedy Best First Graph Search with heuristic h_unmet_goals	88	25	15	0.03679395900007876
Greedy Best First Graph Search with heuristic h_pg_levelsum	88	14	14	23.35424014499995
Greedy Best First Graph Search with heuristic h_pg_maxlevel	88	21	13	28.397349992999807
Greedy Best First Graph Search with heuristic h_pg_setlevel	88	35	17	81.99345919200005
A* Search with heuristic h_unmet_goals	88	7388	12	8.495581813000172
A* Search with heuristic h_pg_levelsum	88	369	12	425.12412937399995

The optimal path length is 12

Problem 4

Search Algorithm	Actions	Expansions	Path Length	Execution Time
Breadth First Search	104	99736	14	95.2429285280001
Depth First Graph Search	-	-	-	-
Uniform Cost Search	104	113339	14	191.60442371800002
Greedy Best First Graph Search with heuristic h_unmet_goals	104	29	18	0.06220758300014495
Greedy Best First Graph Search with heuristic h_pg_levelsum	104	17	17	42.276428506999764

Search Algorithm	Actions	Expansions	Path Length	Execution Time
Greedy Best First Graph Search with heuristic h_pg_maxlevel	104	56	17	100.87494778499968
Greedy Best First Graph Search with heuristic h_pg_setlevel	104	107	23	481.362857686
A* Search with heuristic h_unmet_goals	104	34330	14	57.13325663600017
A* Search with heuristic h_pg_levelsum	104	1208	15	2368.3028229680003

The optimal path length is 14.

Number of nodes expanded vs number of actions

Search Algorithm	Nodes expanded-1	Number of Actions-1	Nodes Expanded-2	Number of Actions-2
Breadth First Search	43	20	3343	72
Depth First Graph Search	21	20	624	72
Uniform Cost Search	60	20	5154	72
Greedy Best First Graph Search with heuristic h_unmet_goals	7	20	17	72
Greedy Best First Graph Search with heuristic h_pg_levelsum	6	20	9	72
Greedy Best First Graph Search with heuristic h_pg_maxlevel	6	20	27	72
Greedy Best First Graph Search with heuristic h_pg_setlevel	6	20	9	72
A* Search with heuristic h_unmet_goals	50	20	2467	72
A* Search with heuristic h_pg_levelsum	28	20	357	72
A* Search with heuristic h_pg_maxlevel	43	20	2887	72
A* Search with heuristic h_pg_setlevel	33	20	1037	72

Search Algorithms	Nodes Expanded-3	Number of Actions-3	Nodes Expanded-4	Number of Actions-4
Breadth First Search	14663	88	99736	104
Depth First Graph Search	408	88	-	-
Uniform Cost Search	18510	88	113339	104
Greedy Best First Graph Search with heuristic h_unmet_goals	25	88	29	104
Greedy Best First Graph Search with heuristic h_pg_levelsum	14	88	17	104
Greedy Best First Graph Search with heuristic h_pg_maxlevel	21	88	56	104
Greedy Best First Graph Search with heuristic h_pg_setlevel	35	88	107	104
A* Search with heuristic h_unmet_goals	7388	88	34330	104
A* Search with heuristic h_pg_levelsum	369	88	1208	104

From the above 2 tables, we see that the number of nodes expanded increases as the number of actions increases. As we solve and move from one problem on to another, the number of actions required to find a path to the goal increases and so does the number of nodes. The amount of increase in the expanded nodes when going from one problem to another varies from algorithm to algorithm. In the case of the uninformed search algorithms and A* with heuristic h_pg_levelsum, the number of nodes expanded increases significantly as the number of actions increases. However, in the case of A* search with heuristic h_pg_levelsum, the number of nodes expanded increases when going from problem 1 to problem 2 but fewer nodes are expanded in problem 3(compared to problem 2). Finally, from problem 3 to problem 4, we see an increase in the number of nodes expanded.

Search time vs number of actions

Search Algorithms	Search time-1	Number of actions-1	Search time-2	Number of actions-2
Breadth First Search	0.006757973000048878	20	1.9982181989998935	72

Search Algorithms	Search time-1	Number of actions-1	Search time-2	Number of actions-2
Depth First Graph Search	0.003534721000050922	20	3.0761934849997488	72
Uniform Cost Search	0.010460765999937394	20	3.3623013939999282	72
Greedy Best First Graph Search with heuristic h_unmet_goals	0.0024077629999510464	20	0.019361825999567372	72
Greedy Best First Graph Search with heuristic h_pg_levelsum	0.4692074989999355	20	10.810248023999975	72
Greedy Best First Graph Search with heuristic h_pg_maxlevel	0.35779809600001045	20	21.229660099000284	72
Greedy Best First Graph Search with heuristic h_pg_setlevel	0.6142740919999596	20	14.85959442999956	72
A* Search with heuristic h_unmet_goals	0.009698603000060757	20	2.2257588270003907	72
A* Search with heuristic h_pg_levelsum	1.191269531999751	20	267.2171387919998	72
A* Search with heuristic h_pg_maxlevel	1.2406019310001284	20	1543.3977732800004	72
A* Search with heuristic h_pg_setlevel	1.4630804359999274	20	1473.381522091	72

Search Algorithms	Search time-3	Number of Actions-3	Search time-4	Number of Actions-4
Breadth First Search	10.589345288999993	88	95.2429285280001	104
Depth First Graph Search	1.1678024960001494	88	-	-
Uniform Cost Search	14.493687367999883	88	191.60442371800002	104
Greedy Best First Graph Search with heuristic h_unmet_goals	0.03679395900007876	88	0.06220758300014495	104
Greedy Best First Graph Search with heuristic h_pg_levelsum	23.35424014499995	88	42.276428506999764	104
Greedy Best First Graph Search with heuristic h_pg_maxlevel	28.397349992999807	88	100.87494778499968	104
Greedy Best First Graph Search with heuristic h_pg_setlevel	81.99345919200005	88	481.362857686	104
A* Search with heuristic h_unmet_goals	8.495581813000172	88	57.13325663600017	104
A* Search with heuristic h_pg_levelsum	425.12412937399995	88	2368.3028229680003	104

As we can see above, the search time increases with the number of actions needed to find a path to the goal. The number of actions, in turn, increases with a problem's complexity. How much the search time increases is dependent on the search algorithm. In the cases of all search algorithms, we can see a gradual increase in the search time from one problem to another except for Depth First Graph Search where it increases from Problem 1 to 2 and then drops at Problem

3(Problem 4's time couldn't be found due to search time being too long). Out of all the algorithms, A* search with heuristic h_pg_levelsum takes the longest time to execute.

Name	Plan Length-1	Plan Length-2	Plan Length-3	Plan Length-4
Breadth First Search	6	9	12	14
Depth First Graph Search	20	619	392	-
Uniform Cost Search	6	9	12	14
Greedy Best First Graph Search with heuristic h_unmet_goals	6	9	15	18
Greedy Best First Graph Search with heuristic h_pg_levelsum	6	9	14	17
Greedy Best First Graph Search with heuristic h_pg_maxlevel	6	9	13	17
Greedy Best First Graph Search with heuristic h_pg_setlevel	6	9	17	23
A* Search with heuristic h_unmet_goals	6	9	12	14
A* Search with heuristic h_pg_levelsum	6	9	12	15
A* Search with heuristic h_pg_maxlevel	6	9	-	-
A* Search with heuristic h_pg_setlevel	6	9	-	-

In the first and second problems, all the search algorithms except Depth First Graph Search, have the same plan length(which is the optimal plan length for the problem). In the third and fourth problems, nearly all the search problems give different plan lengths. The plan length increases with the complexity of the problem.

We can also see that the Greedy Best First Graph Search with heuristic h_unmet_goals is the fastest search algorithm overall and Greedy Best First Search with heuristic h_pg_levelsum is the algorithm that expands the least number of nodes.

Final Questions

- 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?

A problem with a very restricted domain would be like Air Cargo Transport system problem 1. So the best algorithms for such problems would be algorithms with lower execution times. The algorithms with the lowest execution times in Problem 1 would be Greedy Best First Graph Search with heuristic h_unmet_goals and Depth First Graph Search. These algorithms will also be most appropriate for planning in a very restricted domain and can operate in real time.

- 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)

A problem with larger domain would be like Problem 4. In that case, Greedy Best First Graph Search with heuristic h_unmet_goals would be most appropriate because it has the fastest execution time and while it doesn't give the plan length that's optimal for the problem, its plan length isn't too far off from the optimal plan length(Problem 4's optimal plan length is 14, but the Greedy Best First Graph Search algorithm mentioned gave a plan length of 18)

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

If the aim is to find only optimal plans in a problem, then Breadth First search is the best algorithm for the job. A* Search with heuristic h_pg_levelsum is also an option for this, but it has a longer execution time than Breadth First Search.