AIND Experimental Results and report

Laying the groundwork for the problem

There are four chargo problems with different numbers of airplanes, cargo items and airports, thar increase the complexity, from the simplest to the hardest, and It has to compare several implementation of planning agent to see what is the better.

1. Launch all test on first problem

Data:

- 2 Airports: ['JFK', 'SFO']
- 2 Cargos: ['C1', 'C2']
- 2 Planes: ['P1', 'P2']
- Goal: [At(C1, JFK), At(C2, SFO)]
- Initial State: [At(C1, SFO) At(C2, JFK) At(P1, SFO) At(P2, JFK)]

Results:

Search	Heuristic	Actions	Expansions	Goal Test	New Nodes	Plan Length	Time (s)
breadth_first_search	-	20	43	56	178	6	0.00933
depth_first_graph_search	-	20	21	22	84	20	0.00461
uniform_cost_search	-	20	60	62	240	6	0.01296
greedy_best_first_graph_search	h_unmet_goals	20	7	9	29	6	0.00227
greedy_best_first_graph_search	h_pg_levelsum	20	6	8	28	6	0.28713
greedy_best_first_graph_search	h_pg_maxlevel	20	6	8	24	6	0.49627
greedy_best_first_graph_search	h_pg_setlevel	20	6	8	28	6	0.82801
astar_search	h_unmet_goals	20	50	52	206	6	0.01294
astar_search	h_pg_levelsum	20	28	30	122	6	0.69871
astar_search	h_pg_maxlevel	20	43	45	180	6	1.73573
astar_search	h_pg_setlevel	20	33	35	138	6	1.96344

2. Launch all test on second problem

Data:

- 3 Airports: ['JFK', 'SFO', 'ATL']
- 3 Cargos: ['C1', 'C2', 'C3']
- 3 Planes: ['P1', 'P2', 'P3']
- Goal: [At(C1, JFK), At(C2, SFO), At(C3, SFO)]
- Initial State: [At(C1, SFO) At(C2, JFK) At(C3, ATL) At(P1, SFO) At(P2, JFK) At(P3, ATL)]

Results:

Search	Heuristic	Actions	Expansions	Goal Test	New Nodes	Plan Length	Time (s)
breadth_first_search	-	72	3343	4609	30503	9	3.37649

depth_first_graph_search	-	72	624	625	5602	619	4.12341
uniform_cost_search	-	72	5154	5156	46618	9	4.75669
greedy_best_first_graph_search	h_unmet_goals	72	17	19	170	9	0.02665
greedy_best_first_graph_search	h_pg_levelsum	72	9	11	86	9	5.97442
greedy_best_first_graph_search	h_pg_maxlevel	72	27	29	249	9	29.9526
greedy_best_first_graph_search	h_pg_setlevel	72	9	11	84	9	20.0634
astar_search	h_unmet_goals	72	2467	2469	22522	9	2.10916
astar_search	h_pg_levelsum	72	357	359	3426	9	162.185
astar_search	h_pg_maxlevel	72	2887	2889	26594	9	2182.91
astar_search	h_pg_setlevel	72	1037	1039	9605	9	1782.21

3. Launch some test on third problem

Data:

- 4 Airports: ['JFK', 'SFO', 'ATL', 'ORD']
- 4 Cargos: ['C1', 'C2', 'C3', 'C4']
- 2 Planes: ['P1', 'P2']
- Goal: [At(C1, JFK), At(C2, SFO), At(C3, JFK), At(C4, SFO)]
- Initial State: [At(C1, SFO) At(C2, JFK) At(C3, ATL) At(C4, ORD) At(P1, SFO) At(P2, JFK)]

Results:

Search	Heuristic	Actions	Expansions	Goal Test	New Nodes	Plan Length	Time (s)
breadth_first_search	-	88	14663	18098	129625	12	14.8118
depth_first_graph_search	-	88	408	409	3364	392	1.40387
uniform_cost_search	-	88	18510	18512	161936	12	19.6123
greedy_best_first_graph_search	h_unmet_goals	88	25	27	230	15	0.04924
greedy_best_first_graph_search	h_pg_levelsum	88	14	16	126	14	13.8246
greedy_best_first_graph_search	h_pg_maxlevel	88	21	23	195	13	39.8045
greedy_best_first_graph_search	h_pg_setlevel	88	35	37	345	17	108.748
astar_search	h_unmet_goals	88	7388	7390	65711	12	11.2731
astar_search	h_pg_levelsum	88	369	371	3403	12	291.654
astar_search	h_pg_maxlevel	88	9580	8582	86313	12	10493.8
astar_search	h_pg_setlevel	88	4323	3425	31596	12	9538.92

4. Launch some test on forth problem

Data:

- 4 Airports: ['JFK', 'SFO', 'ATL', 'ORD']
- 5 Cargos: ['C1', 'C2', 'C3', 'C4', 'C5']
- 2 Planes: ['P1', 'P2']
- Goal: [At(C1, JFK), At(C2, SFO), At(C3, JFK), At(C4, SFO), At(C5, JFK)]
- Initial State: [At(C1, SFO) At(C2, JFK) At(C3, ATL) At(C4, ORD) At(C5, ORD) At(P1, SFO) At(P2, JFK)]

Results:

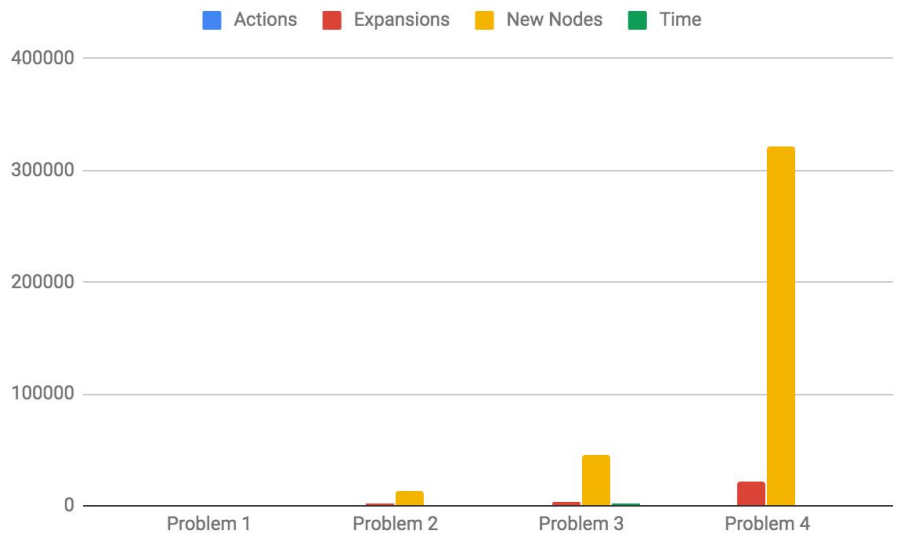
Search	Heuristic	Actions	Expansions	Goal Test	New Nodes	Plan Length	Time (s)
breadth_first_search	-	104	99736	114953	944130	14	133.109
depth_first_graph_search	-	104	25174	25175	228849	24132	5141.07
uniform_cost_search	-	104	113339	113341	1066413	14	191.115
greedy_best_first_graph_search	h_unmet_goals	104	29	31	280	18	0.09650
greedy_best_first_graph_search	h_pg_levelsum	104	17	19	165	17	29.1641
greedy_best_first_graph_search	h_pg_maxlevel	104	56	58	580	17	182.452
greedy_best_first_graph_search	h_pg_setlevel	104	107	109	1164	23	626.030
astar_search	h_unmet_goals	104	34330	34332	328609	14	93.8534
astar_search	h_pg_levelsum	104	1208	1210	12210	15	1842.94
astar_search	h_pg_maxlevel	-	-	-	-	-	-
astar_search	h_pg_setlevel	-	-	-	-	-	-

Conclusions.

Comparing the means of all columns grouped by search in fourth problems…

	Problem 1	Problem 2	Problem 3	Problem 4
Actions	20	72	88	104
Expansions	26	1448	4067	21783
New Nodes	112	13224	45125	321274
Time	1	382	1867	**Too much**

It seems to be a clearly trend to increase of node expansions exponentially and the same trend occurs with the time.



- 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?

In this case, I would suggest the use of ***greedy_best_first_graph_search*** *with* ***h_unmet_goals*** *because It has proved to be the fastest and independently of the number of actions.*

- 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)

In this case, I'll propose the use of **astar_search** with **h_unmet_goals** because I think that is better to wait a few seconds, minutes or even hours, to get the best route than to do an unoptimized route with unnecessary segments with all that means, waste of resources, gasoline, cost, etc.

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

This is the same case of previous point. **star_search** with **h_unmet_goals** because the algorithm always gives you a better solution until the best is reached and if you can afford to wait, the more you wait the better results it will gives you.