Experiment Details

I've run `run_search.py` Python script, the result is recorded as below.

- Problem1

Category	Number of actions in the domain	Number of new node expansions	Time to complete the plan search	Plan length
1. breadth_first_search	20	43	0.00756960001308471	6
2. depth_first_graph_search	20	21	0.0035024620010517538	20
3. uniform_cost_search	20	60	0.009290272952057421	6
4. greedy_best_first_graph_search h_unmet_goals	20	7	0.0016047820099629462	6
5. greedy_best_first_graph_search h_pg_levelsum	20	6	0.36281756800599396	6
6. greedy_best_first_graph_search h_pg_maxlevel	20	6	0.28398865001508966	6
7. greedy_best_first_graph_search h_pg_setlevel	20	6	0.4174564839922823	6
8. astar_search h_unmet_goals	20	50	0.008155328046996146	6
9. astar_search h_pg_levelsum	20	28	0.9013208069954999	6
10. astar_search h_pg_maxlevel	20	43	0.9440778809948824	6
11. astar_search h_pg_setlevel	20	33	0.9781918580411002	6

- Problem2

Category	Number of actions in the domain	Number of new node expansions	Time to complete the plan search	Plan length
1. breadth_first_search	72	3343	1.8369988469639793	9
2. depth_first_graph_search	72	624	2.4815439780359156	619
3. uniform_cost_search	72	5154	2.986213866970502	9
4. greedy_best_first_graph_search h_unmet_goals	72	17	0.017138831957709044	9
5. greedy_best_first_graph_search h_pg_levelsum	72	9	8.893125845992472	9
6. greedy_best_first_graph_search h_pg_maxlevel	72	27	18.81525965797482	9
7. greedy_best_first_graph_search	72	9	11.101002601033542	9

h_pg_setlevel				
8. astar_search h_unmet_goals	72	2467	2.1007573819952086	9
9. astar_search h_pg_levelsum	72	357	229.283577263006	9
10. astar_search h_pg_maxlevel	72	2887	1311.5637586569646	9
11. astar_search h_pg_setlevel	72	1037	986.0691579009872	9

As we see the result of Problem1 and Problem2, uninformed search algorithms should be fine with a
reasonable large problem set.

- Problem3

Category	Number of actions in the domain	Number of new node expansions	Time to complete the plan search	Plan length
1. breadth_first_search	88	14663	10.49843726301333	12
2. depth_first_graph_search	88	408	1.0789199330029078	392
3. uniform_cost_search	88	25	0.036107662017457 19	15
4. greedy_best_first_graph_search h_unmet_goals	88	14	20.0398263669922	14
5. greedy_best_first_graph_search h_pg_levelsum	72	9	8.893125845992472	9
8. astar_search h_unmet_goals	88	7388	8.214598022052087	12
9. astar_search h_pg_levelsum	88	369	358.1219947949867	14

- Problem4

Category	Number of actions in the domain	Number of new node expansions	Time to complete the plan search	Plan length
1. breadth_first_search	104	99736	103.18970193102723	14
2. depth_first_graph_search	104	25174	3286.7369884649524	24132
3. uniform_cost_search	104	113339	102.79693892097566	14
4. greedy_best_first_graph_search h_unmet_goals	104	29	0.057383099978324026	18
5. greedy_best_first_graph_search h_pg_levelsum	104	17	38.346990627003834	17
8. astar_search h_unmet_goals	104	34330	53.57759687898215	14
9. astar_search h_pg_levelsum	104	1208	2013.923558992974	15

Questions and Answers

- Analysis of the number of nodes expanded against number of actions in the domain.

As the problem size increases, the number of actions in the domain also increases. For the uninformed
search, the nodes expansion will explode very fast.

- Analysis of the search time against the number of actions in the domain.

  As the number of actions increasing, the runtime increase rapidly. For some search algorithms, the
  increase speed is far more than the exponential due to the increase of the search space.

- 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?

  breadth_first_search may gives the best real-time performance, but I think the three uninformed search all should be fine. This is because in a very restricted domain problem, we don't need to explore too many space, that makes the information less important compared to the
  larger problem space.

- 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)

  greedy_best_first_graph_search. Even if the solution may not be the optimal, but greedy_best_first_graph_search
  can at least give a reasonable solution in a reasonable time.

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

  breadth_first_search.