Analysis of number of nodes expanded vs. number of actions

The results for each of the problems points to the reasoning that as the number of nodes and the complexity of the graph increases, the number of expansions increases as well. This is due to the fact as the domain gets larger, there are more possible paths to the end state, and thus the heuristic searches more of them to reach the best solution. However, it appears that regardless of the number of expansions, all of the heuristics perform the same number of actions, which would make sense as all of the heuristics are designed to find the absolute best solution: the only real difference is how much time it takes for them to reach it.

Analysis of search time vs number of actions

From these results, it appears that there is no correlation between the number of actions and the time. All of the heuristics attempt to find the absolute best solution to the problem and given an infinite amount of time, they will eventually be able to do so. The number of actions each heuristic performs would be the same as the others since each is finding the best plan for the problem. However, as the complexity of the problem increases, time exponentially increases. Some heuristics are able to manage the complexity better than others but all are affected by slower search times as the problem becomes more complex.

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?

Depth-first search, h-unmet-goals, and breadth first search would be the fastest for very small domains as they are the fastest.

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)

Breadth first search, best first h-unmet-goals, best first h-levelsum, a* h-unmet goals would be the fastest and cover a large number of expansions to ensure all possibilities are taken into consideration

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

To find the plans where only the best plan is important and time is not a factor, A* maxlevel, a* setlevel, A* h-unmet-goals, and uniform cost search would be best.

Methods	Actions				Expansions				Time (sec)				Plan Length			
	P1	P2	P3	P4	P1	P2	P3	P4	P1	P2	P3	P4	P1	P2	P3	P4
breadth_first_search	20	72	88	104	43	3343	14663	99736	0.007815	1.897299	46.81812	688.524	6	9	12	14
depth_first_graph_search	20	72	88	104	21	624	408	25174	0.003049	2.701411	5.136771	5304.578	20	619	392	24132
uniform_cost_search	20	72	88	104	60	5124	18510	113339	0.010424	3.294185	65.6027	883.6901	6	9	12	14
greedy_best_first_graph_search h_unmet_goals	20	72	88	104	7	17	25	29	0.001894	0.018584	0.208901	0.433151	6	9	15	18
greedy_best_first_graph_search h_pg_levelsum	20	72	88	104	6	9	14	17	0.423524	7.06226	95.72129	345.3387	6	9	14	17
greedy_best_first_graph_search h_pg_maxlevel	20	72	88	104	6	27	21	56	0.369601	14.76552	94.25771	730.6588	6	9	13	17
greedy_best_first_graph_search h_pg_setlevel	20	72	88	104	6	9	35	107	1.392723	27.52584	141.6766	639.827	6	9	17	23
astar_search h_unmet_goals	20	72	88	104	50	2467	7388	34330	0.013631	2.245636	60.66291	415.6959	6	9	12	14
astar_search h_pg_levelsum	20	72	88	104	28	357	369	1208	1.04917	208.1731	358.3011	2252.549	6	9	12	15
astar_search h_pg_maxlevel	20	72	88	104	43	2887	9580	62077	1.354352	1346.092	7032.591	23078.24	6	9	12	14
astar_search h_pg_setlevel	20	72	88	104	51	2102	5963	37912	3.23951	2176.761	15048.22	63703.93	6	9	12	14