Artificial Intelligence Nanodegree

Project 2: Building a Forward Planning Agent

Tables and Charts

Table 1 - Results for all search algorithms applied to Problem 1

problem	algorithm	#Actions	Expansions	Goal Tests	*New Nodes*	*Plan Length*	Elapsed Time	% Expansions	% Goal Tests	*% New Nodes*	*% Plan Length*	% Elapsed Time
1. Air Cargo Problem 1	1. breadth_first_search	20	43	56	178	6	0.003464496	617%	600%	642%	0%	234%
1. Air Cargo Problem 1	2. depth_first_graph_search	20	21	22	84	20	0.001888452	250%	175%	250%	233%	82%
1. Air Cargo Problem 1	3. uniform_cost_search	20	60	62	240	6	0.006675638	900%	675%	900%	0%	543%
1. Air Cargo Problem 1	4. greedy_best_first_graph_search h_unmet_goals	20	7	9	29	6	0.001038294999	17%	13%	21%	0%	0%
1. Air Cargo Problem 1	5. greedy_best_first_graph_search h_pg_levelsum	20	6	8	28	6	0.074006606	0%	0%	17%	0%	7028%
1. Air Cargo Problem 1	6. greedy_best_first_graph_search h_pg_maxlevel	20	6	8	24	6	0.048494997	0%	0%	0%	0%	4571%
1. Air Cargo Problem 1	7. greedy_best_first_graph_search h_pg_setlevel	20	6	8	28	6	0.184879421	0%	0%	17%	0%	17706%
1. Air Cargo Problem 1	8. astar_search h_unmet_goals	20	50	52	206	6	0.005319054	733%	550%	758%	0%	412%
1. Air Cargo Problem 1	9. astar_search h_pg_levelsum	20	28	30	122	6	0.153868783	367%	275%	408%	0%	14719%
1. Air Cargo Problem 1	10. astar_search h_pg_maxlevel	20	43	45	180	6	0.161934941	617%	463%	650%	0%	15496%
1. Air Cargo Problem 1	11. astar_search h_pg_setlevel	20	33	35	138	6	0.432831479	450%	338%	475%	0%	41587%

Table 2 - Results for all search algorithms applied to Problem 2

problem	algorithm	#Actions	Expansions	Goal Tests	*New Nodes*	*Plan Length*	Elapsed Time	% Expansions	% Goal Tests	*% New Nodes*	*% Plan Length*	% Elapsed Time
2. Air Cargo Problem 2	1. breadth_first_search	72	3343	4609	30503	9	1.079045819	37044%	41800%	36213%	0%	10189%
2. Air Cargo Problem 2	2. depth_first_graph_search	72	624	625	5602	619	1.538673731	6833%	5582%	6569%	6778%	14572%
2. Air Cargo Problem 2	3. uniform_cost_search	72	5154	5156	46618	9	1.817410071	57167%	46773%	55398%	0%	17230%
2. Air Cargo Problem 2	4. greedy_best_first_graph_search h_unmet_goals	72	17	19	170	9	0.010487054	89%	73%	102%	0%	0%
2. Air Cargo Problem 2	5. greedy_best_first_graph_search h_pg_levelsum	72	9	11	86	9	1.127041378	0%	0%	2%	0%	10647%
2. Air Cargo Problem 2	6. greedy_best_first_graph_search h_pg_maxlevel	72	27	29	249	9	1.796024597	200%	164%	196%	0%	17026%
2. Air Cargo Problem 2	7. greedy_best_first_graph_search h_pg_setlevel	72	9	11	84	9	4.295722285	0%	0%	0%	0%	40862%
2. Air Cargo Problem 2	8. astar_search h_unmet_goals	72	2467	2469	22522	9	1.201313808	27311%	22345%	26712%	0%	11355%
2. Air Cargo Problem 2	9. astar_search h_pg_levelsum	72	357	359	3426	9	37.19211048	3867%	3164%	3979%	0%	354548%
2. Air Cargo Problem 2	10. astar_search h_pg_maxlevel	72	2887	2889	26594	9	192.6799824	31978%	26164%	31560%	0%	1837213%
2. Air Cargo Problem 2	11. astar_search h_pg_setlevel	72	1037	1039	9605	9	439.4727099	11422%	9345%	11335%	0%	4190521%

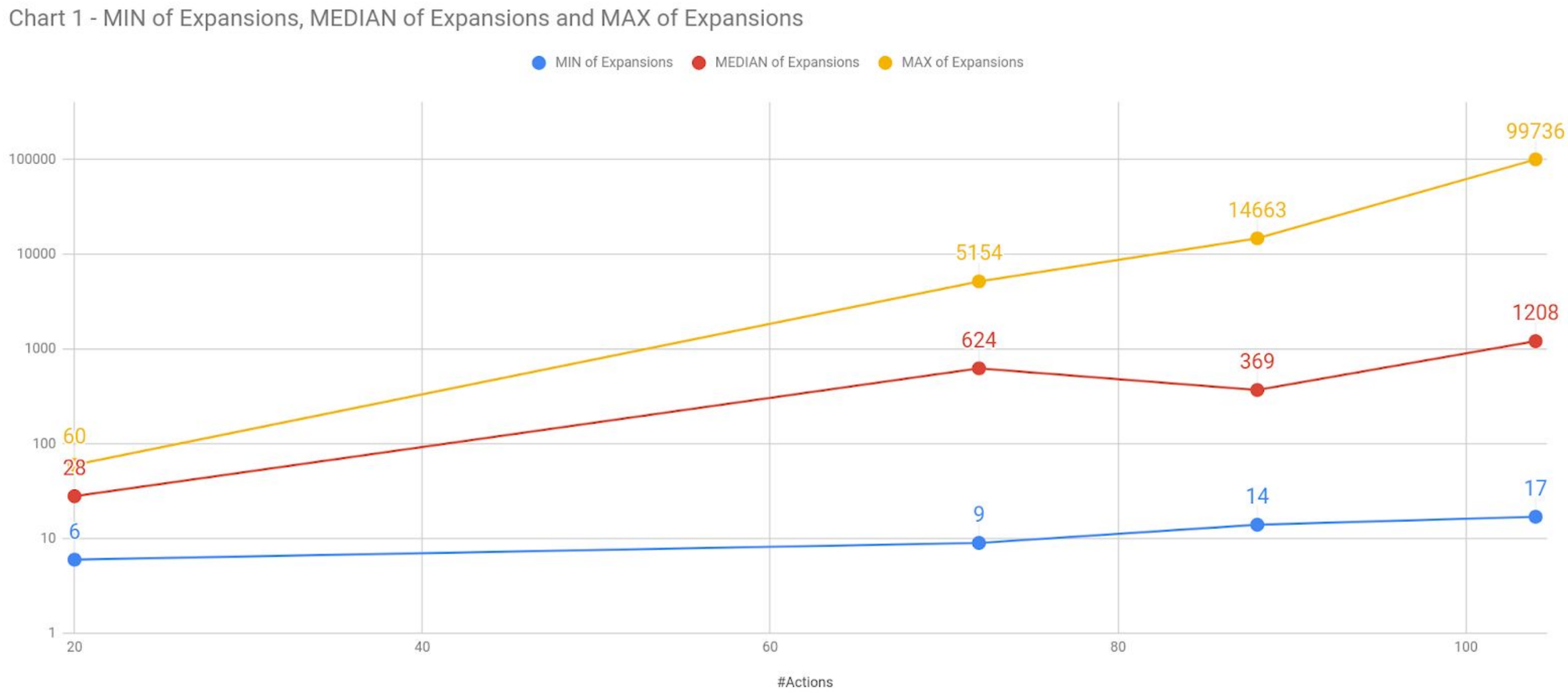
Table 3 - Results for selected search algorithms applied to Problem 3

problem	algorithm	#Actions	Expansions	Goal Tests	*New Nodes*	*Plan Length*	Elapsed Time	% Expansions	% Goal Tests	*% New Nodes*	*% Plan Length*	% Elapsed Time
3. Air Cargo Problem 3	1. breadth_first_search	88	14663	18098	129625	12	6.525260194	104636%	113013%	102777%	0%	24847%
3. Air Cargo Problem 3	4. greedy_best_first_graph_search h_unmet_goals	88	25	27	230	15	0.026156659	79%	69%	83%	25%	0%
3. Air Cargo Problem 3	5. greedy_best_first_graph_search h_pg_levelsum	88	14	16	126	14	2.749719616	0%	0%	0%	17%	10413%
3. Air Cargo Problem 3	8. astar_search h_unmet_goals	88	7388	7390	65711	12	4.682276619	52671%	46088%	52052%	0%	17801%
3. Air Cargo Problem 3	9. astar_search h_pg_levelsum	88	369	371	3403	12	57.89986809	2536%	2219%	2601%	0%	221258%

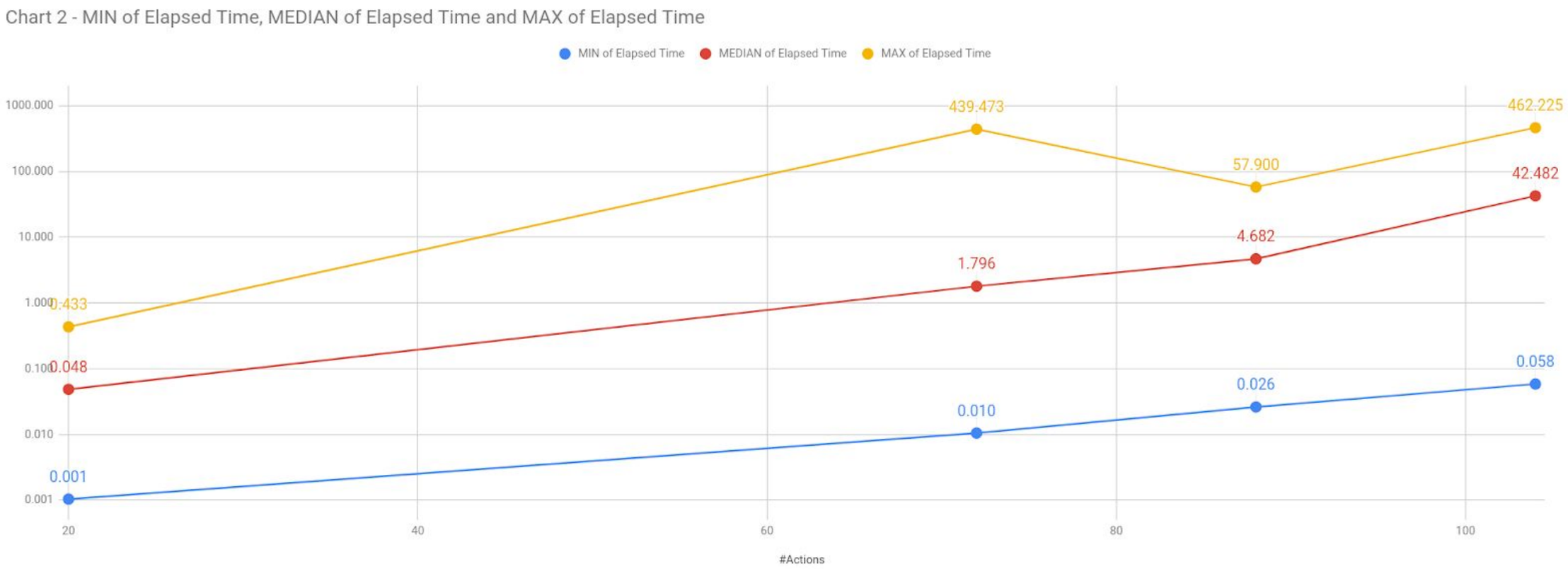
Table 4 - Results for selected search algorithms applied to Problem 4

problem	algorithm	#Actions	Expansions	Goal Tests	*New Nodes*	*Plan Length*	Elapsed Time	% Expansions	% Goal Tests	*% New Nodes*	*% Plan Length*	% Elapsed Time
4. Air Cargo Problem 4	1. breadth_first_search	104	99736	114953	944130	14	77.08585955	586582%	604916%	572100%	0%	131799%
4. Air Cargo Problem 4	4. greedy_best_first_graph_search h_unmet_goals	104	29	31	280	18	0.058443315	71%	63%	70%	29%	0%
4. Air Cargo Problem 4	5. greedy_best_first_graph_search h_pg_levelsum	104	17	19	165	17	6.742832589	0%	0%	0%	21%	11437%
4. Air Cargo Problem 4	8. astar_search h_unmet_goals	104	34330	34332	328509	14	42.48205252	201841%	180595%	198996%	0%	72589%
4. Air Cargo Problem 4	9. astar_search h_pg_levelsum	104	1208	1210	12210	15	462.2245844	7006%	6268%	7300%	7%	790794%

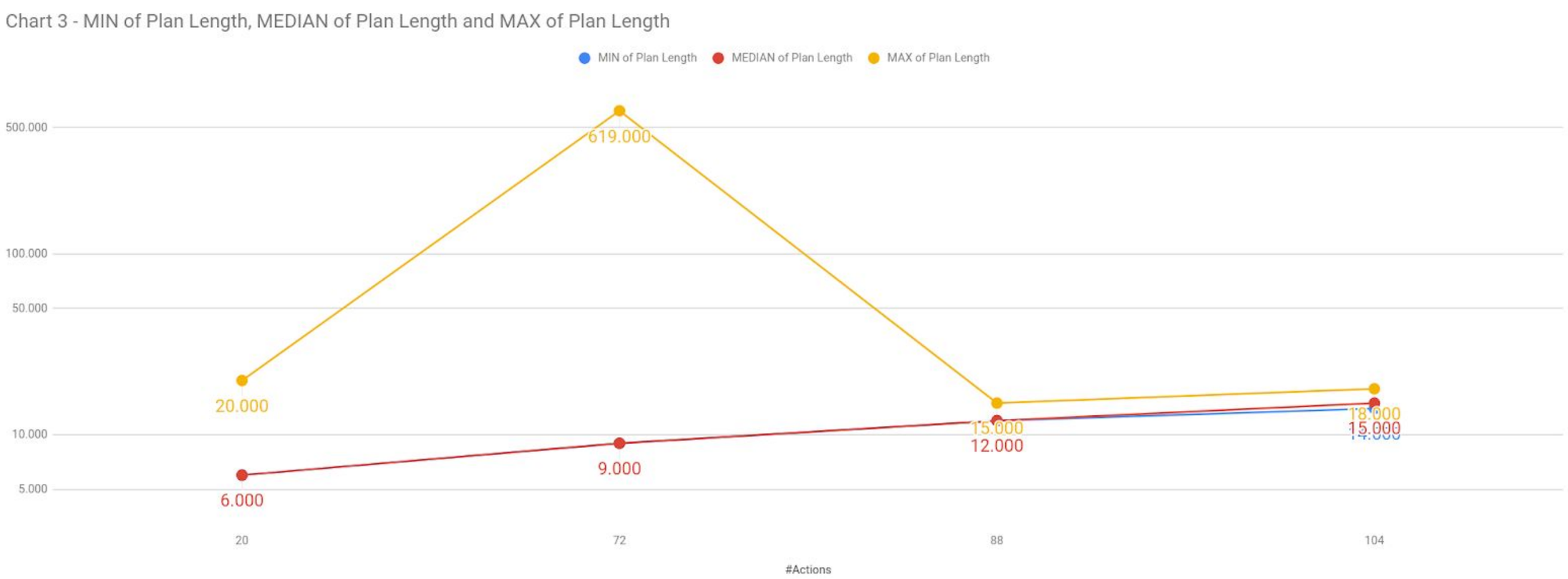
Use a table or chart to analyze the number of nodes expanded against number of actions in the domain



Use a table or chart to analyze the search time against the number of actions in the domain



Use a table or chart to analyze the length of the plans returned by each algorithm on all search problems



Tables 1-4 are provided to expose full information about the experiments that were carried on for this report:

- Columns 1-2 indicate which problem and algorithm were used to produce the results for a given row.
- Columns 3-8 were obtained directly from the results of running the `run_search.py` script for the different combinations of problems and algorithms.
- Columns 9-13 provide insight regarding the percent increase for a given metric relative to the minimum value found in the experiment.

The charts produced show a growth trend in nodes expanded, search time and plan length as the number of actions in a domain grow. In fact, since all the charts are log-linear and show an approximately linear trend for these variables, we can say that these quantities grow exponentially with the number of actions. An important thing to note in these graphs are the outliers in Chart 3, due to the explosion in plan length given as a result of using depth-first search.

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?

In a very restricted domain, such as the one seen in problem 1 **(Air Cargo Problem 1)**, algorithm 4 **(greedy_best_first_graph_search h_unmet_goals)** was shown to find near-optimal plans within a millisecond. As the number of actions grows, plans grow up to 25% larger than optimal, but elapsed time remained within a few milliseconds.

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)

The answer to this question depends on three factors: (1) how long one would be willing to wait for a plan to be generated, (2) how important it is for plans to be optimized for length and (3) how hard it is to achieve a solution within those constraints, given the size of the domain. The largest domain we have experimented with was problem 4 **(Air Cargo Problem 4)**, where three possible solutions to this question emerged:

- Algorithm 4 **(greedy_best_first_graph_search h_unmet_goals)** reached a solution quickest, but generated a plan that was 25% larger than optimal
- Algorithm 5 **(greedy_best_first_graph_search h_pg_levelsum)** was second quickest, even though it was 100 times slower, and generated a plan that was 17% larger than optimal
- Algorithm 8 **(astar_search h_unmet_goals)** was third quickest, even though it was 725 time slower, but generated the optimal plan

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

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.