Forward Planning Agent Project Report

This report details the results of a selection of different forward planning algorithms applied to 4 different cargo transportation problems of increasing domain complexity. A forward planning agent has been implemented in Python, using which the problems were approached using either one of three uninformed search algorithms, or one of two heauristic search algorithms, each with a choice of four different heuristics. Each algorithm was applied to the two most basic problems, with the results from these used to choose a selection for tackling the two more complex problems. These results are outlined and discussed below, with the full tables of collected results included at the end of the report as appendices.

The Setlevel and Maxlevel heuristics were found to take too long when applied to the two more basic problems to be easily tested on the more complex ones for both the Greedy Best First and A* Search algorithms. Depth First Search was dropped from testing on Problem 4 for the same reason.

1. Search Complexity as a function of Domain Size, Algorithm and Heuristic

Shown below in fig. 1 is a plot of the number of actions taken in each problem against the number of new nodes expanded. Each column of points is taken from a different problem, with the points representing the number of nodes expanded using a particular algorithm, using all of the four tables appearing in the appendices. Judging by the shape of the graph, the search complexity (number of nodes expanded) seems to generally increase exponentially with the domain size (number of actions). This is a trend followed by the Breadth First Search, Uniform Cost Search, and the A* Search algorithms, however the A* Search with Levelsum heuristic shows a lower increase in search complexity as the domain size rises. The Greedy Best First Search algorithms buck this trend by maintaining a relatively constant value of expanded nodes around an order of 10.

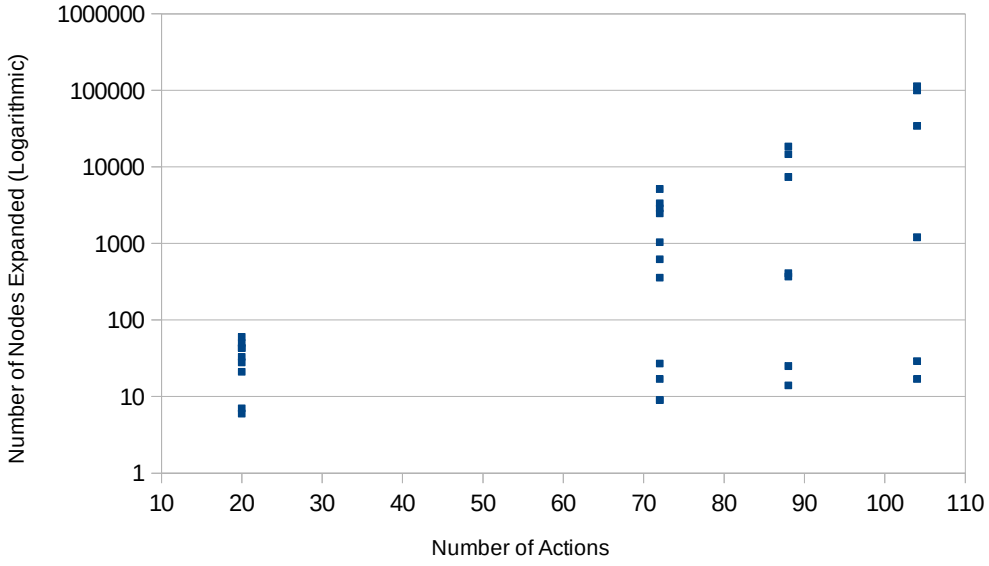

Figure 1. Scatter plot of (number of actions v number of expanded nodes) for each chosen problem/algorithm/heuristic combination

2. Search Time as a function of Domain Size, Algorithm and Heuristic

Shown below in fig. 2 is a plot of the search time in seconds for each number of actions and each algorithm. As the domain size increases from 20 actions to 72 actions, each algorithm/heuristic combination seems to make a significant jump in search time by at least an order of magnitude. However, increasing the complexity further doesn't seem to have such an effect on the search time and in fact there seems to be very littlediscernible trend in this portion of the graph.

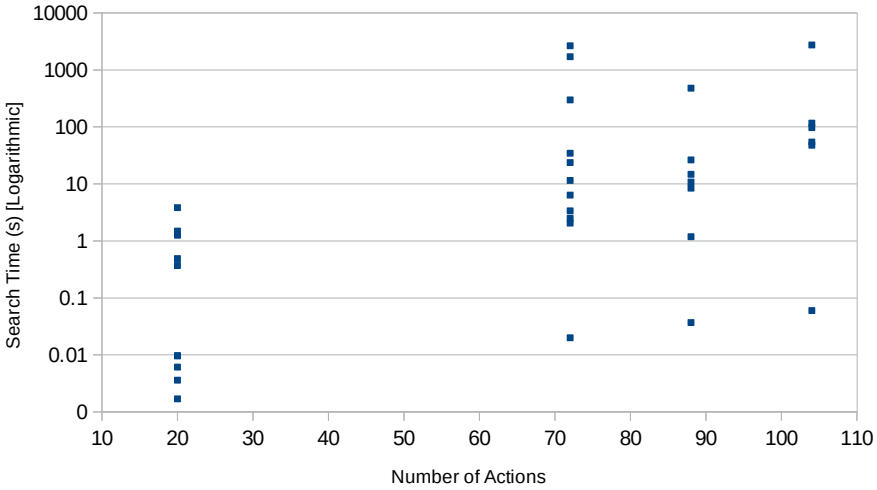

Figure 2. Scatter plot for number of actions against search time

3. Solution Optimality as a function of Domain Size, Algorithm and Heuristic

Shown below in fig. 3 is a logarithmic bar plot showing the length of the plans returned by each algorithm/heuristic combination for each problem that it was used for.

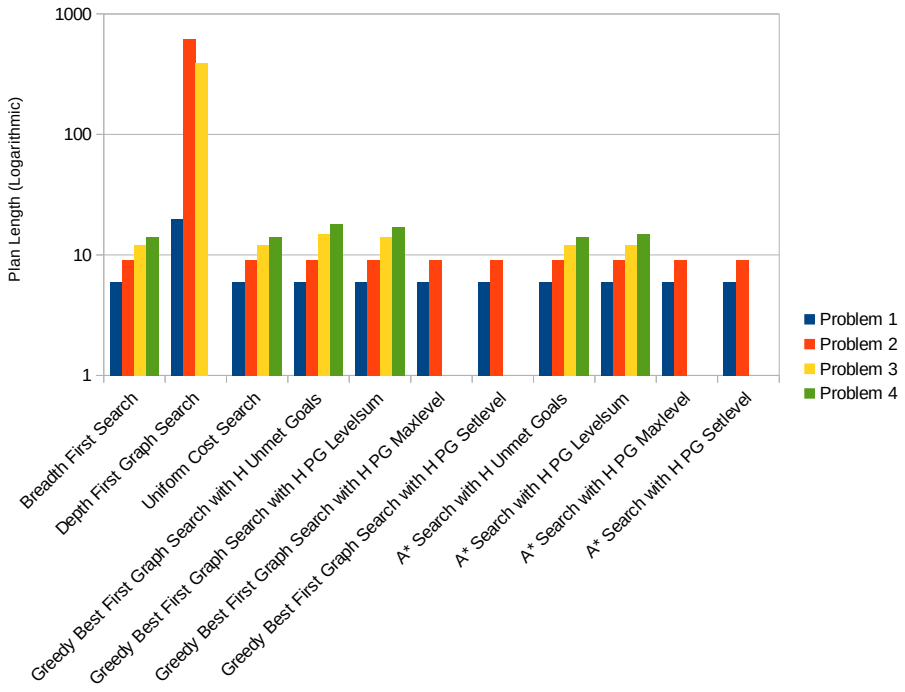

Figure 3. Bar plot of the length pf the plans returned by each algorithm/heuristic combination broken down by problem

4. Discussion

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?

The algorithm which returns a solution the fastest in the smallest domain of the first problem is the Greedy Best First Search algorithm combined with the Unmet Goals heuristic. This is a good choice as it also returns a plan of minimum length. All of the uninformed algorithms, as well as the A* Search with Unmet Goals heuristic would also perform well in this scenario

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)

Again, the Greedy Best First Search algorith with the Unmet Goals heuristic returns a solution the quickest in this scenario and would be an appropriate choice. If, in this case, a shorter plan length was preferred to a shorter runtime, then the A* algorithm used along with the Unmet Goals heuristic would be a better choice.

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

In this case, the Breadth First Search, Uniform Cost and A* Search with Unmet Goals algorithm could be reled upon to give the plan closest to optimal. These three algorithms also take a relatively similar amount of time and search complexity.

5. Appendices

Problem 1				
Algorithm	**Domain Actions**	**New node expansions**	**Time elapsed (s)**	**Plan Length**
Breadth First Search	20	43	0.0061	6
Depth First Search	20	21	0.0036	20
Uniform Cost Search	20	60	0.0096	6
Greedy Best First Search with Unmet Goals	20	7	0.0017	6
Greedy Best First Search with PG Levelsum	20	6	0.49	6
Greedy Best First Search with H PG Maxlevel	20	6	0.37	6
Greedy Best First Search with H PG Setlevel	20	6	1.48	6
A* Search with Unmet Goals	20	50	0.0097	6
A* Search with PG Levelsum	20	28	1.27	6
A* Search with PG Maxlevel	20	43	1.28	6
A* Search with PG Setlevel	20	33	3.85	6

Problem 2				
Algorithm	**Domain Actions**	**New node expansions**	**Time elapsed (s)**	**Plan Length**
Breadth First Search	72	3343	2.06	9
Depth First Search	72	624	6.38	619
Uniform Cost Search	72	5154	3.37	9
Greedy Best First Search with Unmet Goals	72	17	0.020	9
Greedy Best First Search with PG Levelsum	72	9	11.5	9
Greedy Best First Search with PG Maxlevel	72	27	23.7	9
Greedy Best First Search with PG Setlevel	72	9	34.5	9
A* Search with Unmet Goals	72	2467	2.49	9
A* Search with PG Levelsum	72	357	297	9
A* Search with PG Maxlevel	72	2887	1711	9
A* Search with PG Setlevel	72	1037	2641	9

Problem 3				
Algorithm	**Domain Actions**	**New node expansions**	**Time elapsed (s)**	**Plan Length**
Breadth First Search	88	14663	10.8	12
Depth First Search	88	408	1.19	392
Uniform Cost Search	88	18510	14.8	12
Greedy Best First Search with Unmet Goals	88	25	0.037	15
Greedy Best First Search with PG Levelsum	88	14	26.3	14
A* Search with Unmet Goals	88	7388	8.47	12
A* Search with PG Levelsum	88	369	477	12

Problem 4				
Algorithm	**Domain Actions**	**New node expansions**	**Time elapsed (s)**	**Plan Length**
Breadth First Search	104	99736	97.0	14
Uniform Cost Search	104	113339	117	14
Greedy Best First Search with Unmet Goals	104	29	0.06	18
Greedy Best First Search with PG Levelsum	104	17	47.7	17
A* Search with Unmet Goals	104	34330	54.6	14
A* Search with PG Levelsum	104	1208	2736	15