Report: Build a Forward-Planning Agent

1. You may plot multiple results for the same topic on the same chart or use multiple charts. (Hint: you may see more detail by using log space for one or more dimensions of these charts.)

A) Use a table or chart to analyze the number of nodes expanded against number of actions in the domain
B) Use a table or chart to analyze the search time against the number of actions in the domain
C) Use a table or chart to analyze the length of the plans returned by each algorithm on all search problems

I wrote the answers for A, B, C - all the questions at once.

Algorithm	#Problem	#Actions	#New Nodes	#Search Times	#Plan Length
BFS	1	20	178	6	6
	2	72	30503	9	9
	3	88	129625	12	12
	4	104	944130	14	14
DFS	1	20	240	0.01	6
	2	72	625	3.12	20
Uniform Cost Search	1	20	240	0.01	6
	2	72	46618	3.514	9
Greedy Best First Graph Search - unmet goals	1	20	29	0.0016	6
	2	72	170	0.019	9
	3	88	230	0.0364	15
	4	104	280	0.59	18
Greedy Best First Graph Search - levelsum	1	20	28	0.35	6
	2	72	86	7.95	9
	3	88	126	18	14
	4	104	165	32	17
Greedy Best First Graph Search - maxlevel	1	20	24	0.2681	6
	2	72	249	16.092	9
Greedy Best First Graph Search - setlevel	1	20	28	2.02	6
	2	72	84	61	9
A Star - unmet goals	1	20	206	0.0095	6
	2	72	22522	2.24	9
	3	88	65711	8.41	12
	4	104	328509	56.92	14
A Star - level sum	1	20	122	0.91	6
	2	72	3426	203	9
	3	88	65711	311	12
	4	104	328509	3219	15
A Star - max level	1	20	180	0.92	6
	2	72	26594	1159	9
A Star - set levels	1	20	138	5.22	6
	2	72	9605	1202	9

Against the number of actions. A* finds optimal solutions for the problems. But shows very long search times. This is interesting. As far as I know, A* shows the short search time and find not optimal route. Maybe heuristics are inappropriate for this problems. A* also needs many nodes that makes large memory usage.

Greedy Best First Graph Search Algorithms creates little nodes and less time overall against the number of actions. Also shows the fastest search time. But searched plans are not optimal as I learned.

For the overall number of actions. BFS creates most numerous nodes as I expected. But search time is fast enough. But shows the optimal solution and fast enough speed. I can't understand this result and maybe need more analysis…

DFS is expected to increase search time exponentially with more actions. Searched plan is not suitable for practical use. Number of new nodes are not many against number of actions.

Uniform Cost Search shows exponential number of new nodes over number of actions. Search time will be increased exponentially. Searched plans are optimal for problem 1, 2 but may not optimal for problem 3, 4 which has large number of actions.

2. Use your results to answer the following 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?

: Depth-First Search

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

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

: BFS, Uniform Cost Search