Artificial Intelligence Nanodegree Program

Project#2: Build a Forward-Planning Agent

March 12th, 2022

(1) Planning Graph Implementation

➔ **my_planning_graph.py**

(2) Heuristic Implementation

➔ **my_planning_graph.py**

(3) Experimental Results & Report

(a) Analyze the results as a function of domain size, search algorithm, and heuristic.

<Definition of 11 Search Algorithms>

The number of Search Algorithm is defined as below:

1 [["breadth_first_search", breadth_first_search, ""],

2 ['depth_first_graph_search', depth_first_graph_search, ""],

3 ['uniform_cost_search', uniform_cost_search, ""],

4 ['greedy_best_first_graph_search',greedy_best_first_graph_search, 'h_unmet_goals'],

5 ['greedy_best_first_graph_search',greedy_best_first_graph_search, 'h_pg_levelsum'],

6 ['greedy_best_first_graph_search',greedy_best_first_graph_search, 'h_pg_maxlevel'],

7 ['greedy_best_first_graph_search',greedy_best_first_graph_search, 'h_pg_setlevel'],

8 ['astar_search', astar_search, 'h_unmet_goals'],

9 ['astar_search', astar_search, 'h_pg_levelsum'],

10 ['astar_search', astar_search, 'h_pg_maxlevel'],

11 ['astar_search', astar_search, 'h_pg_setlevel']

<Definition of 4 Problems>

Problem is defined in air_cargo_problems.py as below:

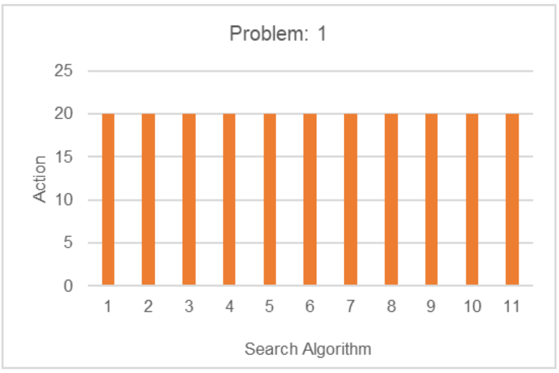
Problem	Cargos	Planes	Airports	Degree of Freedom
Problem1	'C1', 'C2'	'P1', 'P2'	'JFK', 'SFO'	8
Problem2	'C1', 'C2', 'C3'	'P1', 'P2', 'P3'	'JFK', 'SFO', 'ATL'	27
Problem3	'C1', 'C2', 'C3', 'C4'	'P1', 'P2'	'JFK', 'SFO', 'ATL', 'ORD'	32
Problem4	'C1', 'C2', 'C3', 'C4', 'C5'	'P1', 'P2'	'JFK', 'SFO', 'ATL', 'ORD'	40

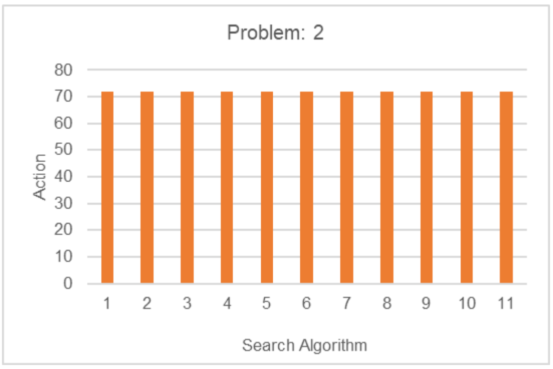
(I skip writing down every initial positions and goals.)

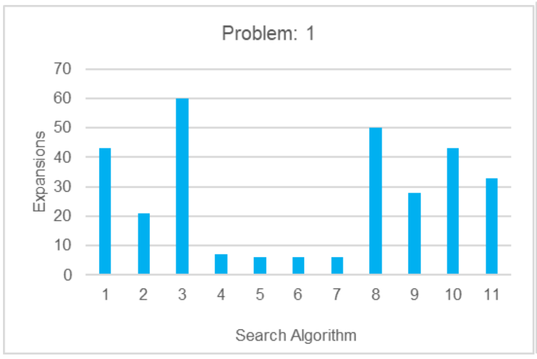
The search complexity, search time and the optimality as a function of domain size, search algorithm, and heuristic are shown below.

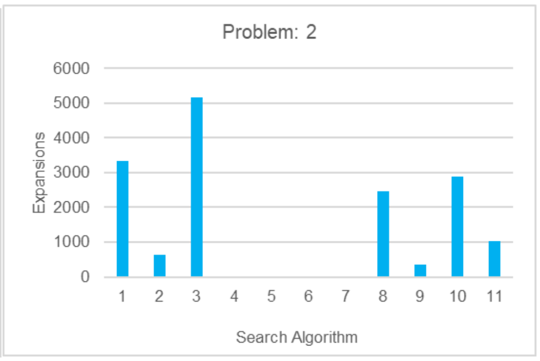
<Charts and Tables>
The chart and table includes data for all search & heuristic combinations for air cargo problems 1 and 2 are shown in the following table and graphs.

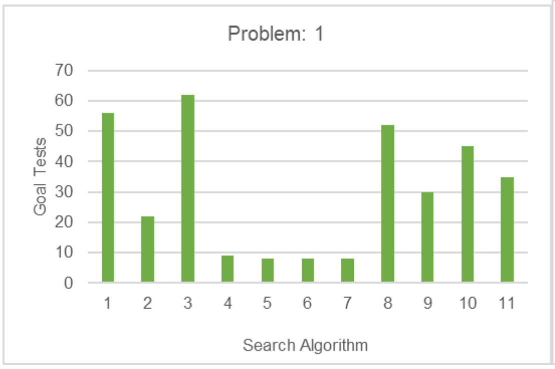
problem:	search:	Actions	Expansions	Goal Tests	New Nodes	Time[sec]	Length
1	1	20	43	56	178	0.0021486	6
1	2	20	21	22	84	0.0011303	20
1	3	20	60	62	240	0.005367	6
1	4	20	7	9	29	0.0007895	6
1	5	20	6	8	28	0.0760047	6
1	6	20	6	8	24	0.0494218	6
1	7	20	6	8	28	0.1923547	6
1	8	20	50	52	206	0.0042586	6
1	9	20	28	30	122	0.1624685	6
1	10	20	43	45	180	0.2002593	6
1	11	20	33	35	138	0.6102644	6
2	1	72	3343	4609	30503	0.9925236	9
2	2	72	624	625	5602	1.2478241	619
2	3	72	5154	5156	46618	1.3925428	9
2	4	72	17	19	170	0.0069921	9
2	5	72	9	11	86	2.0741232	9
2	6	72	27	29	249	2.7404866	9
2	7	72	9	11	84	4.7835643	9
2	8	72	2467	2469	22522	0.7209032	9
2	9	72	357	359	3426	37.8724868	9
2	10	72	2887	2889	26594	347.719866	9
2	11	72	1037	1039	9605	614.468431	9

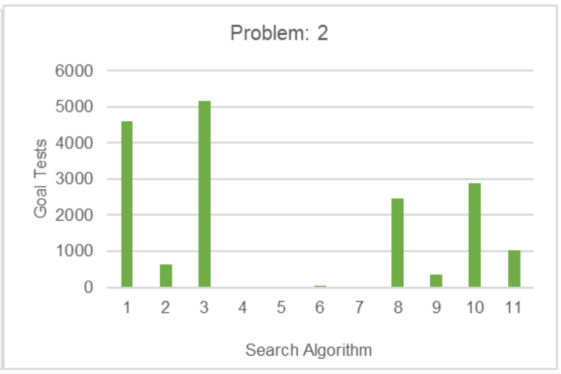


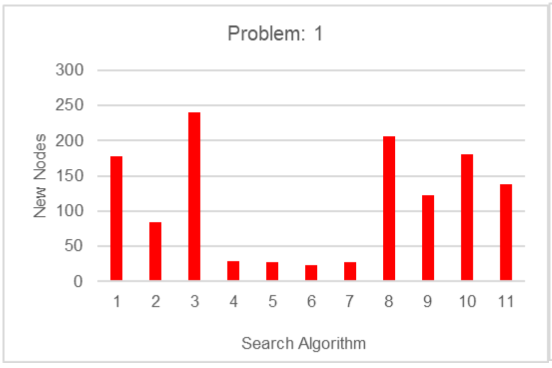


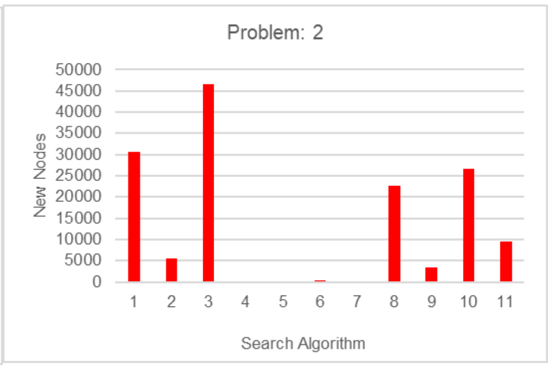

Problem: 1
Expansions
Search Algorithm

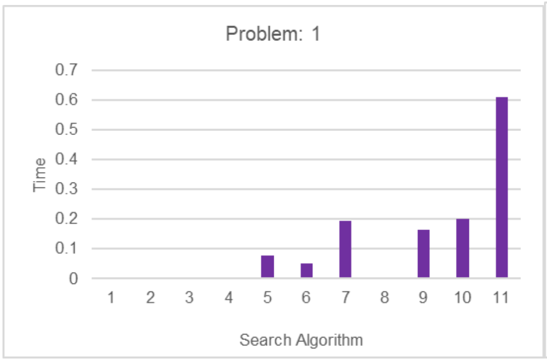

Problem: 2
Expansions
Search Algorithm

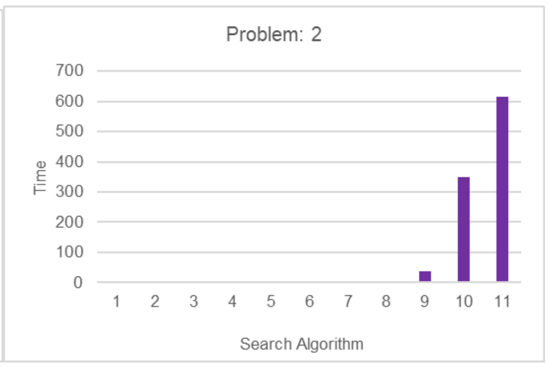

Problem: 1
Goal Tests
Search Algorithm

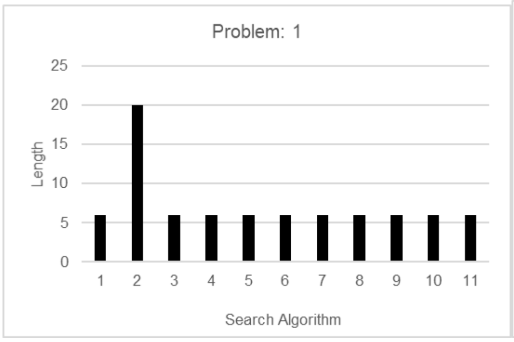

Problem: 2
Goal Tests
Search Algorithm

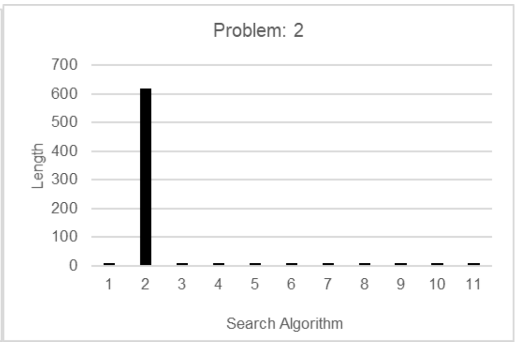

Problem: 1
New Nodes
Search Algorithm


Problem: 2
New Nodes
Search Algorithm


Problem: 1
Time
Search Algorithm

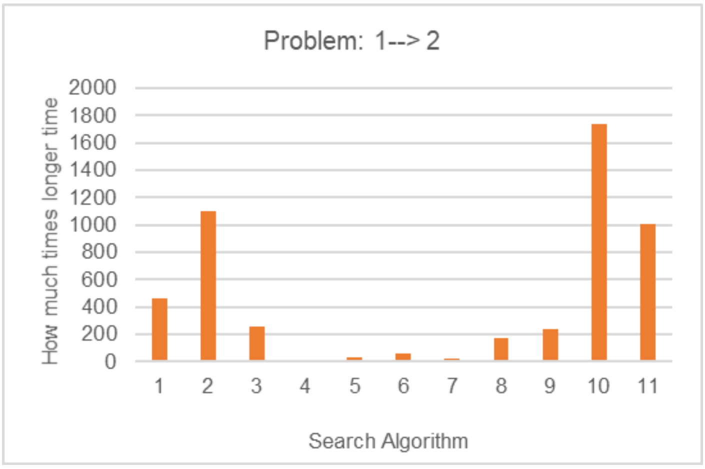

Problem: 2
Time
Search Algorithm





<1st Analysis>

- Only No.2 (depth first graph search) has big length (20) compared with other length (6) in problem1. In No.2, it became more than 30 times (619) even though other length became only 1.5 times (9) length. It might be memory shortage issue if the problem becomes more complicated.
- All of number of Actions became 3.6 times bigger from problem1: 20 to problem2: 72, but the Time became much longer in all of the search algorithm as below. At worst, in No.10 (A* search, 'h_pg_maxlevel') Time became from problem1: 0.2sec to problem2: 348sec, which means 1736 times longer! At best, in No.4 (greedy best first graph search, 'h_unmet_goals') Time became from problem1: 0.0008sec to problem2: 0.007sec, which means 9 times longer.



<Algorithm Selection for problem3, 4>

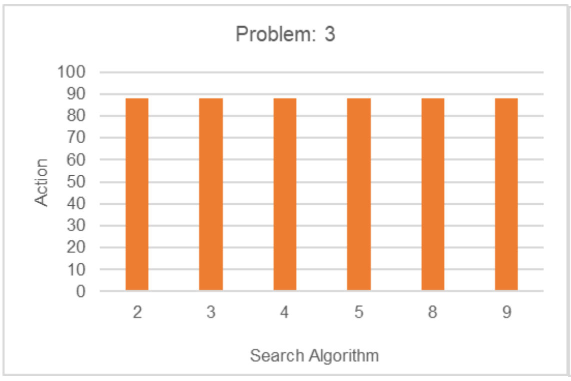
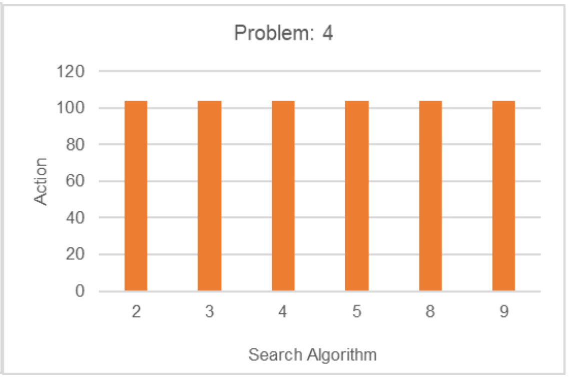
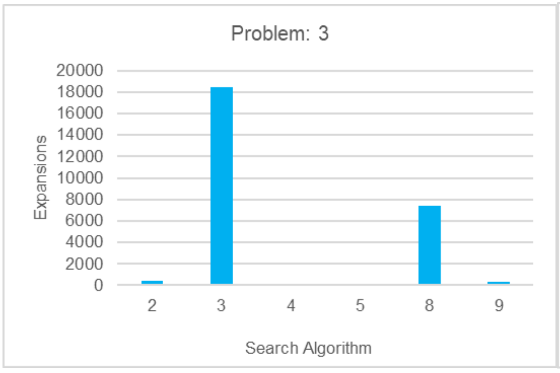
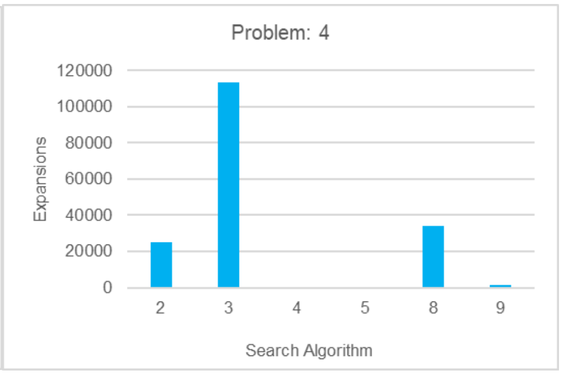
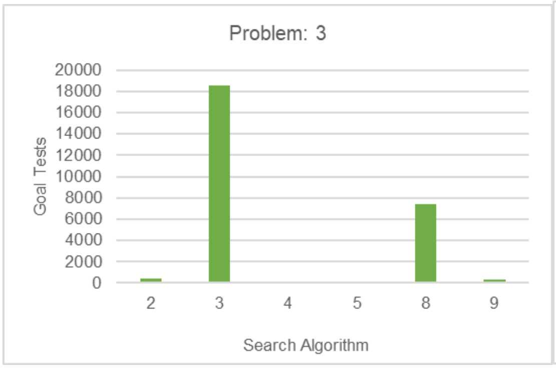
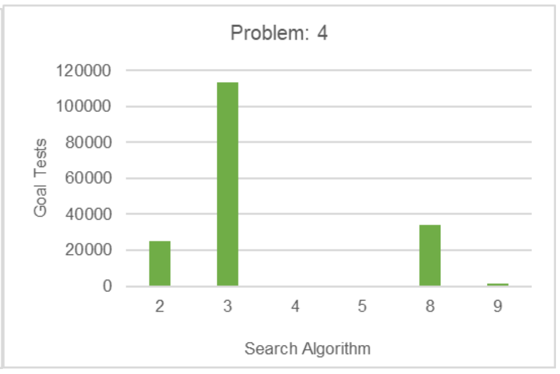
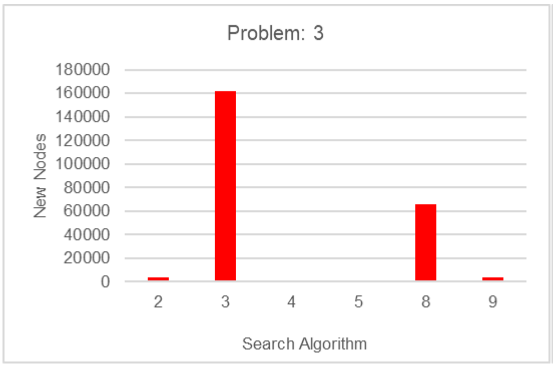
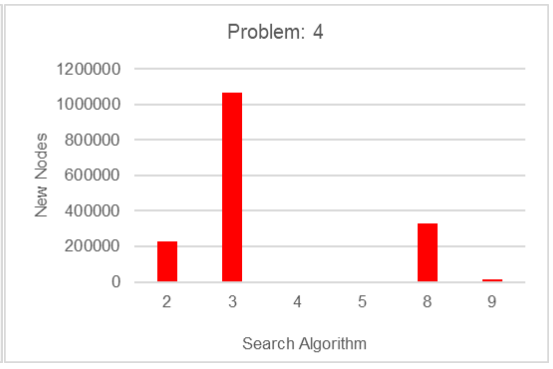
I selected the following search algorithms:

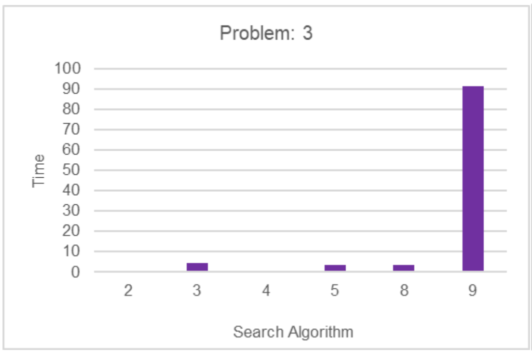
- Equal or more than one uninformed search:
  I should select one from No.1, 2, 3.
  Action number and Time are not so different, but No.2 has the least Expansions, Goal Tests, New Nodes and Time, so **I chose No.2.**
  However, the length of No.2 is much bigger than other 2, which means that it would be memory shortage issue in the more complicated problem.
  So for now, **I also chose No.3**.

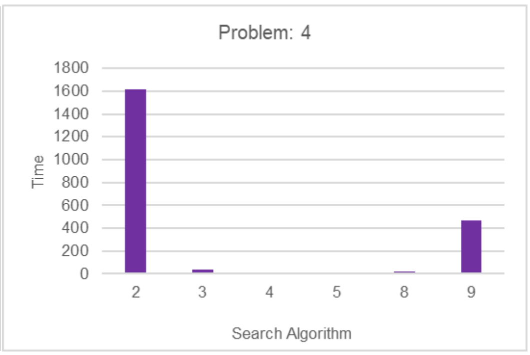
- Two heuristics with greedy best first search:
  I should select two from No.4, 5, 6, 7.
  No.4 is much faster than others in problem 1 and 2, so **I chose No.4**.
  There are not so difference between No. 5, 6, 7 in problem 1, but No.5 is faster than No.6, 7 in problem 2, so **I chose No.5**.
- Two heuristics with A*:
  I should select two from No.8, 9, 10, 11.
  No.10, 11 are much slower than No.8, 9, so **I chose No.8, 9**.

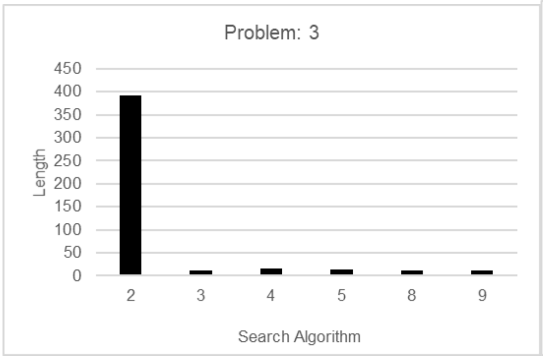
The chart and table includes data for selected search & heuristic combinations for air cargo problems 3 and 4 are shown in the following table and graphs.

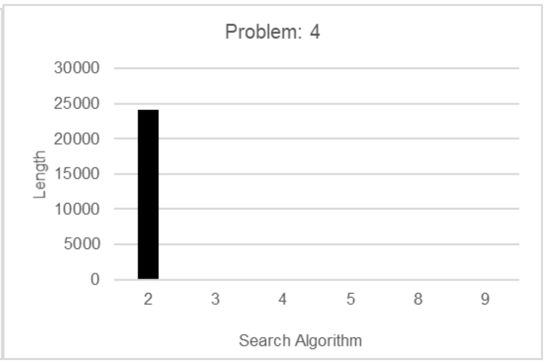
problem:	search:	Actions	Expansions	Goal Tests	New Nodes	Time[sec]	Length
3	2	88	408	409	3364	0.3132749	392
3	3	88	18510	18512	161936	4.5208902	12
3	4	88	25	27	230	0.0109552	15
3	5	88	14	16	126	3.2859022	14
3	8	88	7388	7390	65711	3.2535974	12
3	9	88	369	371	3403	91.4527926	12
4	2	104	25174	25175	228849	1613.09021	24132
4	3	104	113339	113341	1066413	37.411781	14
4	4	104	29	31	280	0.0243932	18
4	5	104	17	19	165	5.9306433	17
4	8	104	34330	34332	328509	17.7507104	14
4	9	104	1208	1210	12210	466.308389	15


Problem: 3
Action
Search Algorithm
2 3 4 5 8 9

Problem: 4
Action
Search Algorithm
2 3 4 5 8 9

Problem: 3
Expansions
Search Algorithm
2 3 4 5 8 9

Problem: 4
Expansions
Search Algorithm
2 3 4 5 8 9

Problem: 3
Goal Tests
Search Algorithm
2 3 4 5 8 9

Problem: 4
Goal Tests
Search Algorithm
2 3 4 5 8 9

Problem: 3
New Nodes
Search Algorithm
2 3 4 5 8 9

Problem: 4
New Nodes
Search Algorithm
2 3 4 5 8 9



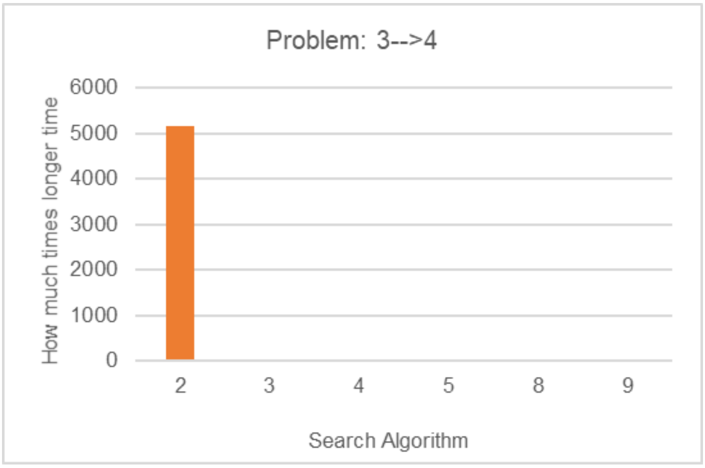






<2nd Analysis>

- No.2 (depth first graph search) is still fast (about 0.3sec) in proglem3, but it's the slowest (about 1613sec) algorithm in these 6 algorithm.
  The number of Actions became from 88 to 104 (about 1.2 times bigger), but the Length became from 392 to 24132 exponentially (about 62 times bigger). That's why it became so slow more than exponentially as below.
  On the contrary, other algorithm became slower only 2 ~ 8 times from Problem 3 to 4.



<u>(b) Report answers all required questions</u>

I answer the following 3 questions which are written in the rubric.

- **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?**

→ I think it's No.2 (depth first graph search) No.4 (greedy_best_first_graph_search, h_unmet_goals) because they are fast in a very restricted domain such as problem 1.

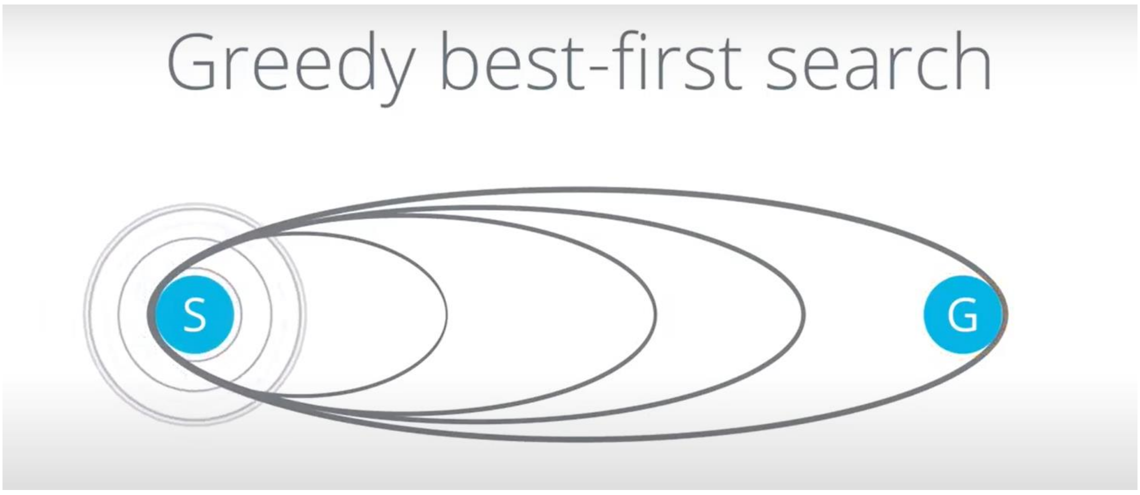
- **Which algorithm or algorithms would be most appropriate for planning in very large domains (e.g., planning delivery routes for all UPS drivers in the U.S. on a given day)**

→ I think it's No.4 (greedy_best_first_graph_search, h_unmet_goals) because it's the fastest algorithm in a very large domain such as problem 4.

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

→ I think it's Greedy Best First Graph Search (No.4, 5) because it not only is fastest but also has smallest Expansions, Goal Tests, New Nodes compared with other algorithms.

It's intuitively understandable as I learned in the Lesson as below.


Greedy best-first search
S
G