**Artificial Intelligence Nanodegree Program**

**Project#2: Build a Forward-Planning Agent**

**March 11th, 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.**

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'],

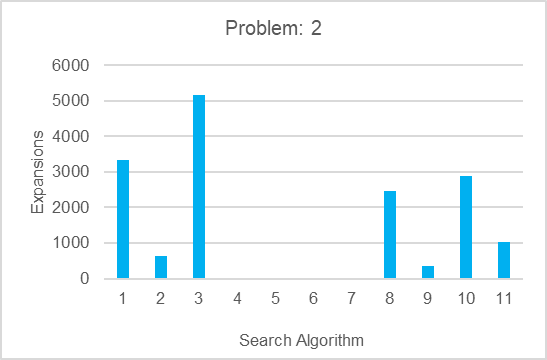
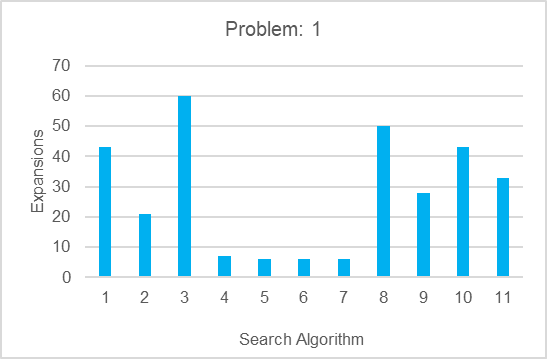
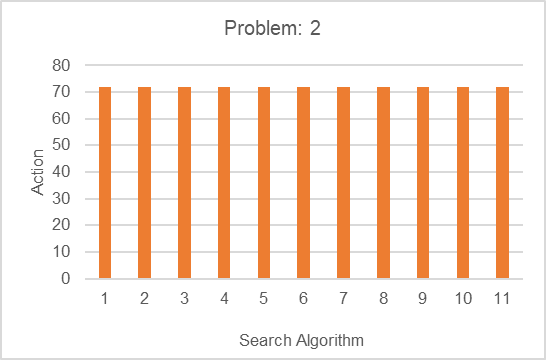
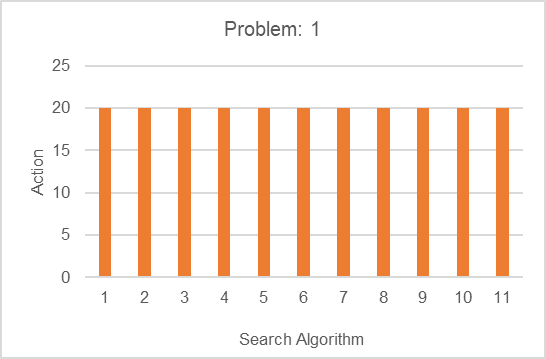
10 ['astar\_search', astar\_search, 'h\_pg\_maxlevel'],

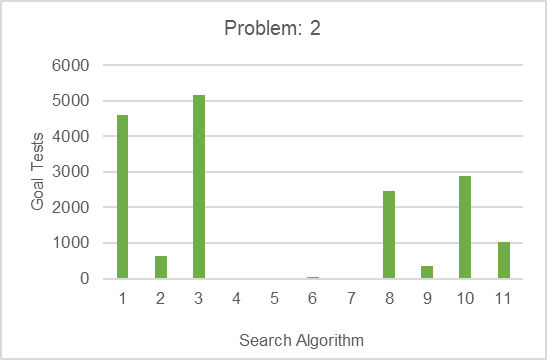
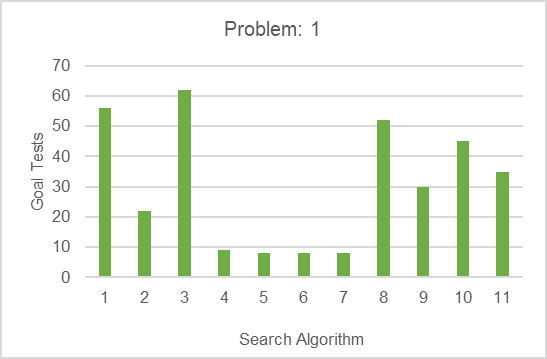
11 ['astar\_search', astar\_search, 'h\_pg\_setlevel']

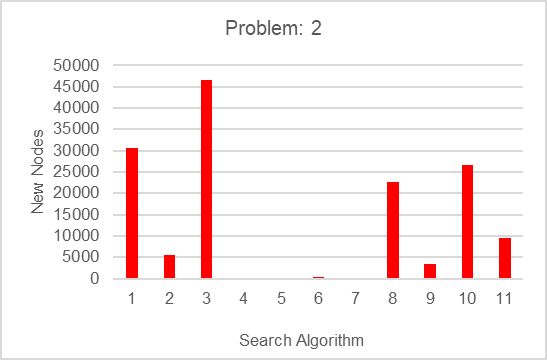
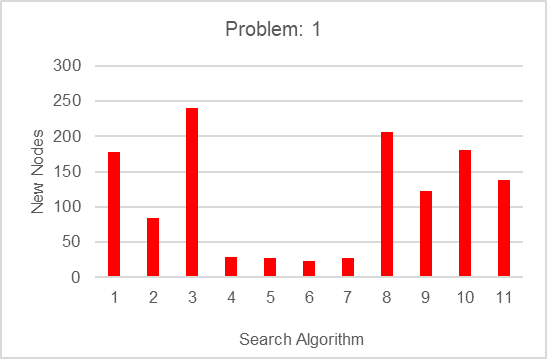
The search complexity, search time and the optimality as a function of domain size, search algorithm, and heuristic are shown below.

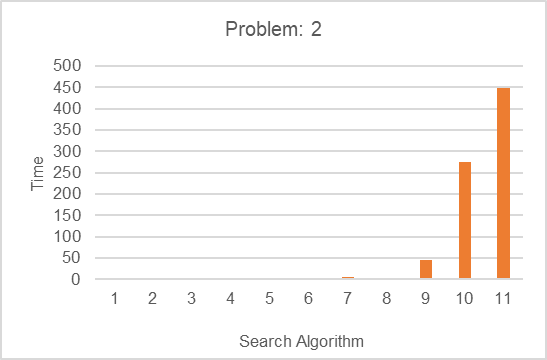
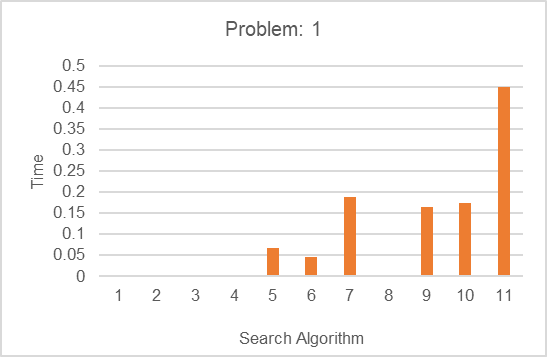
At first, the chart and table includes data for all search & heuristic combinations for air cargo problems 1 and 2 are shown in the following table and graphs.











Before I go to the problem 3 and 4, I selected the following search algorithms:

* 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 and New Nodes. It will be better to run with more complicated problems, so **I chose No.2 from uninformed search algorithm**.

* 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**.

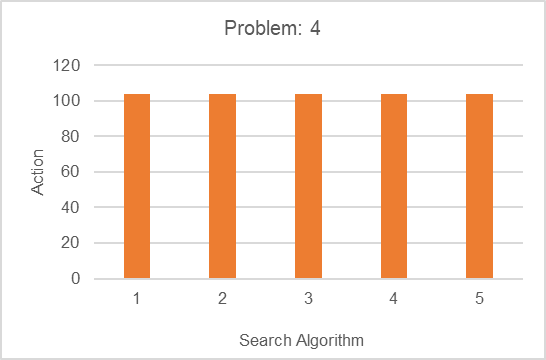
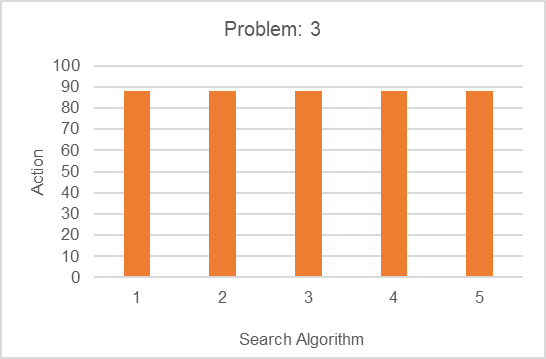
* Two heuristics with A\*:

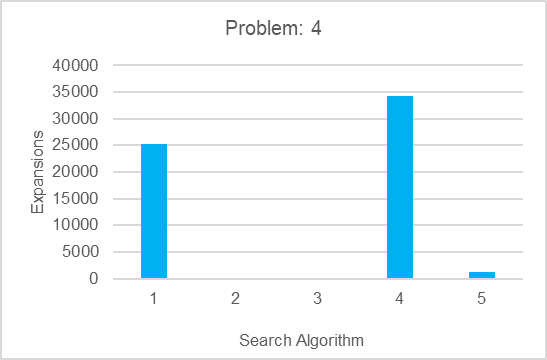
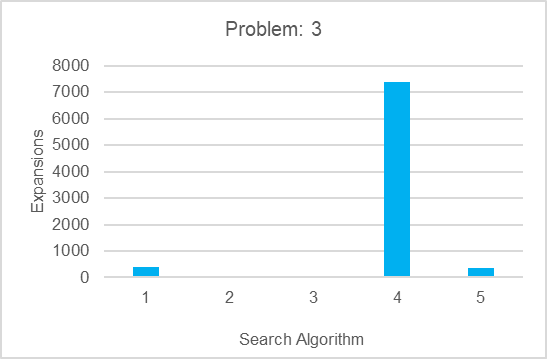
I should select two from No.8, 9, 10, 11.

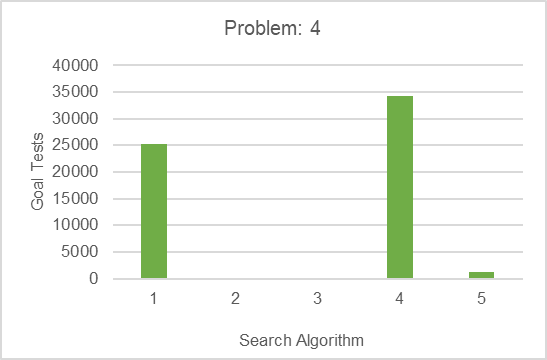
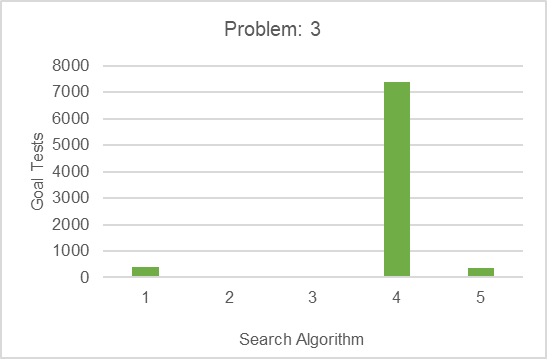
No.10, 11 are much slower than No.8, 9, so **I chose No.8, 9**.

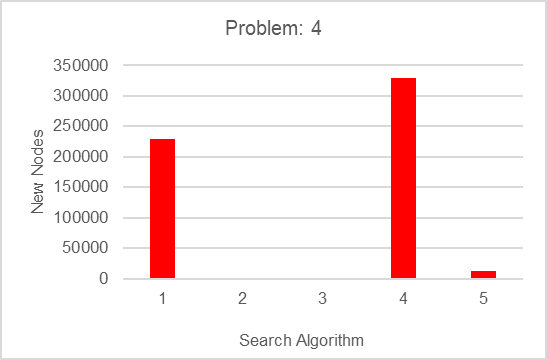
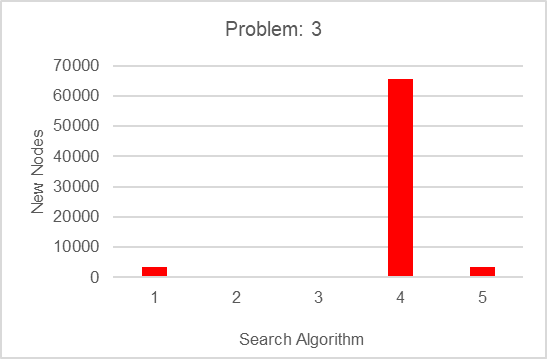
The chart and table includes data for selected search & heuristic combinations for air cargo problems 3 and 4 are shown in the following table and graphs.

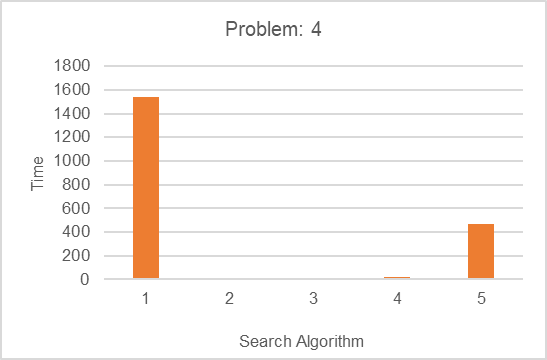
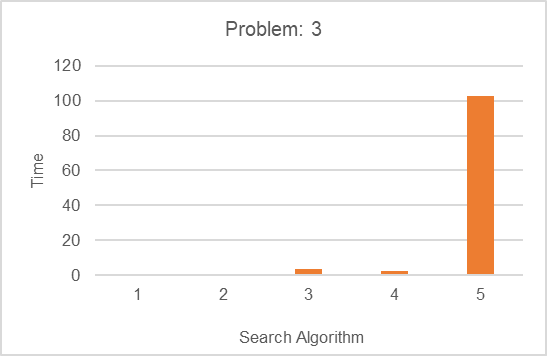












**(b) Report answers all required questions**

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.4 (greedy\_best\_first\_graph\_search, h\_unmet\_goals) because it’s the fastest algorithm in a very restricted domain (project 1, 2).

**- 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 (project 3, 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.

