Problem1

breadth_first_search:

```
Solving Air Cargo Problem 1 using breadth_first_search...

Expansions   Goal Tests   New Nodes
    43          56         180

Plan length: 6  Time elapsed in seconds: 0.05047972351741667
```

depth_first_graph_search:

```
Solving Air Cargo Problem 1 using depth_first_graph_search...

Expansions   Goal Tests   New Nodes
    21          22          84

Plan length: 20  Time elapsed in seconds: 0.02573355243532336
```

uniform_cost_search

```
Solving Air Cargo Problem 1 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
    55          57         224

Plan length: 6  Time elapsed in seconds: 0.06007537919750669
```

greedy_best_first_graph_search h_1

```
Solving Air Cargo Problem 1 using greedy_best_first_graph_search with h_1...

Expansions   Goal Tests   New Nodes
    7           9           28

Plan length: 6  Time elapsed in seconds: 0.00743163500316286
```

astar_search h_1

```
Solving Air Cargo Problem 1 using astar_search with h_1...

Expansions   Goal Tests   New Nodes
    55          57         224

Plan length: 6  Time elapsed in seconds: 0.05949298619444103
```

astar_search h_ignore_preconditions

```
Solving Air Cargo Problem 1 using astar_search with h_ignore_preconditions...

Expansions   Goal Tests   New Nodes
    41          43         170

Plan length: 6  Time elapsed in seconds: 0.058020013931599244
```

astar_search h_pg_levelsum

```
Solving Air Cargo Problem 1 using astar_search with h_pg_levelsum...
Expansions   Goal Tests   New Nodes
    11          13          50

Plan length: 6  Time elapsed in seconds: 0.9087934626582961
```

Problem2

breadth_first_search:

```
Solving Air Cargo Problem 2 using breadth_first_search...

Expansions   Goal Tests   New Nodes
   3343        4609       30509

Plan length: 9  Time elapsed in seconds: 19.67697101903366
```

depth_first_graph_search:

```
Solving Air Cargo Problem 2 using depth_first_graph_search...

Expansions   Goal Tests   New Nodes
   624         625         5602

Plan length: 619  Time elapsed in seconds: 4.489640398002795
```

uniform_cost_search

```
Solving Air Cargo Problem 2 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
   4852        4854       44030

Plan length: 9  Time elapsed in seconds: 17.99821607570635
```

greedy _best_first_graph_search with h_1

```
Solving Air Cargo Problem 2 using greedy_best_first_graph_search with h_1...

Expansions   Goal Tests   New Nodes
   990         992         8910

Plan length: 17  Time elapsed in seconds: 3.500348289981141
```

astar_search h_1

```
Solving Air Cargo Problem 2 using astar_search with h_1...

Expansions   Goal Tests   New Nodes
   4852        4854       44030

Plan length: 9  Time elapsed in seconds: 18.565844981522652
```

astar_search h_ignore_preconditions

```
Solving Air Cargo Problem 2 using astar_search with h_ignore_preconditions...

Expansions   Goal Tests   New Nodes
   1450        1452       13303

Plan length: 9  Time elapsed in seconds: 6.344480719986313
```

astar_search h_pg_levelsum

```
Solving Air Cargo Problem 2 using astar_search with h_pg_levelsum...

Expansions   Goal Tests   New Nodes
    86          88         841

Plan length: 9  Time elapsed in seconds: 64.66915968275781
```

Problem3

breadth_first_search:

```
Solving Air Cargo Problem 3 using breadth_first_search...

Expansions   Goal Tests   New Nodes
   8602       11196       64308

Plan length: 12  Time elapsed in seconds: 64.63650774396993
```

depth_first_graph_search

```
Solving Air Cargo Problem 3 using depth_first_graph_search...

Expansions   Goal Tests   New Nodes
   1292        1293        5744

Plan length: 875  Time elapsed in seconds: 5.168086364852226
```

uniform_cost_search

```
Solving Air Cargo Problem 3 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
  11484       11486       85804

Plan length: 12  Time elapsed in seconds: 48.215552579933835
```

greedy_best_first_graph_search with h_1

```
Solving Air Cargo Problem 3 using greedy_best_first_graph_search with h_1...

Expansions   Goal Tests   New Nodes
   907         909         5581

Plan length: 19  Time elapsed in seconds: 3.2517711819772677
```

astar_search h_1

```
Solving Air Cargo Problem 3 using astar_search with h_1...

Expansions   Goal Tests   New Nodes
  11484       11486       85804

Plan length: 12  Time elapsed in seconds: 46.75077767640996
```

astar_search h_ignore_preconditions

```
Solving Air Cargo Problem 3 using astar_search with h_ignore_preconditions...

Expansions   Goal Tests   New Nodes
   4118        4120       31475

Plan length: 13  Time elapsed in seconds: 19.878763872826546
```

astar_search h_pg_levelsum

```
Solving Air Cargo Problem 3 using astar_search with h_pg_levelsum...

Expansions   Goal Tests   New Nodes
   278         280         1946

Plan length: 12  Time elapsed in seconds: 149.54619257610915
```

Visualization

The table below shows Problem k Plan length and Problem k running time and the averages among each of the problems. Listed for each searching algorithms which are Breadth first search (BFS), Depth first graph search (DFGS), Uniform cost search (UCS), Greedy best first graph search h1 (GBFGSh1), Astar search h1 (A*h1), Astar search h ignore preconditions (A*hIP), and Astar search h pg levelsum (A*hPgLs)

	BFS	**DFGS**	**UCS**	**GBFGSh1**	**A*h1**	**A*hIP**	**A*hPgLs**
Problem 1							
Plan length	6	20	6	6	6	6	6
Time	0.05047972	0.02573355	0.06007538	0.00743164	0.05949299	0.05802001	0.90879346
Expansion	43	21	7	55	55	41	11
Problem 2							
Plan length	9	619	9	17	9	9	9
Time	19.676971	4.4896404	17.9982161	3.50034829	18.565845	6.34448072	64.6691597
Expansion	3,343	624	4,852	990	4,852	1450	86
Problem 3							
Plan length	12	875	12	19	12	13	12
Time	64.6365077	5.16808636	48.2155526	3.25177118	46.7507777	19.8787639	149.546193
Expansion	8,602	1292	11,484	907	11,484	4,118	278
Average							
Plan length	**9**	504.666667	**9**	14	**9**	9.33333333	**9**
Time	28.1213195	3.22782011	22.0912813	**2.2531837**	21.7920385	8.76042154	71.7080486
Expansion	3,996	646	5,448	651	5,464	1,870	**125**

Conclusion

The best result is A-Star search h1 because I consider the less plan length is the better result.

If considering computing time alone, Greedy Best First Graph Search h1 is the best because it's 9.67 times faster than A-Star Search h1! Although it provided 14 average plan length, whilst the minimum average is 9 plan length.

If considering only expansion which means consuming less memory, A-Star search h Planning graph Level sum would be the best. It has just 125 average expansions.

If expecting both minimum plan length and minimum computing time, A-Star h Ignore Preconditions would be the best.

Analysis

DFS is always suboptimal. It may be very fast comparing to BFS, but optimal in planning means shortest plan length. If the problem represents as a tree, DFS searches the deepest nodes in the search tree first from the leftmost. If it reaches the leaf child node, it back-tracks to the upper depth. The more depth and back-tracking means the more plan length. *(Becker, 2015)* The worst-case search space of DFS can be $O(b^d)$ *(Depth-first search, 2017)*. I guess many possible goals are in the planning graph, or the goal is not too far (rightmost), so DFS can find one of them (or it). That's why DFS is faster for our problems.

BFS is much slower as it checks to all branches for each depth (search space = $O(b^d)$ *(Breadth-first search, 2017)*, because it searches the shallowest nodes in the search tree first. If the tree is very wide (with lots of available actions per state), it could take a long time to search. *(Becker, 2015)*

A-star search (or BFS with heuristic) is more optimal. As a good heuristic helps BFS skips unnecessary branches using a calculated cost. An admissible heuristic can be derived by defining a relaxed problem that is easier to solve. The exact cost of a solution to this easier problem then becomes the heuristic for the original problem. Ignore preconditions heuristic drop all preconditions from actions. This almost implies that the number of step required to solve the relaxed problem is the number of unsatisfied goals. *(Stuart Russell, 2014)* It runs very fast with almost optimum result.

Whilst planning graph with level sum heuristic estimates the sum of the levels where any literal of the goal first appears (in the planning graph). *(Grastien)* Although it is optimal and it has the minimum average expansions (least memory usage), it has the maximum average time. Planning graph has advantage properties such as: search must terminate, any plan found is a sound plan, and it finds shortest length plan assuming that multiple actions may occur at the same time.

The reason why planning graph is the slowest is because it's disadvantage properties such as: it has polynomial time to construct the planning graph, and planning is PSPACE-complete. Thus, extraction may be intractable. *(Beek)*

Works Cited

Becker, K. (2015, November 06). *Artificial Intelligence Planning with STRIPS, A Gentle Introduction*. Retrieved May 17, 2017, from Primary Objects: http://www.primaryobjects.com/2015/11/06/artificial-intelligence-planning-with-strips-a-gentle-introduction/

Beek, P. (n.d.). Max level costg i admissible level sum heuristic. Waterloo, Ontario, Canada.

Breadth-first search. (2017, May 9). Retrieved May 17, 2017, from Wikipedia: https://en.wikipedia.org/wiki/Breadth-first_search

Depth-first search. (2017, May 9). Retrieved May 17, 2017, from Wikipedia: https://en.wikipedia.org/wiki/Depth-first_search

Grastien, A. (n.d.). Planning (05) Planning graph. Australia.

Stuart Russell, P. N. (2014). Heuristics for planning. In *Artificial Intelligence A Modern Approach* (pp. 383-384). Pearson.