Project 3 - Implement a Planning Search

I. Optimal plan for Air Cargo Problems 1, 2, and 3.

a. Air Cargo Problem 1

```
Init(At(C1, SFO) ^ At(C2, JFK) ^ At(P1, SFO) ^ At(P2, JFK)
          ^ Cargo(C1) ^ Cargo(C2)
          ^ Plane(P1) ^ Plane(P2)
          ^ Airport(JFK) ^ Airport(SFO))
Goal(At(C1, JFK) ^ At(C2, SFO))
```

OPTIMAL PLAN LENGTH FOR PROBLEM 1 IS = 6

Here are the 6 actions:

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)
3. Fly(P1, SFO, JFK)
4. Fly(P2, JFK, SFO)
5. Unload(C1, P1, JFK)
6. Unload(C2, P2, SFO)

b. Air Cargo Problem 2

```
Init(At(C1, SFO) ^ At(C2, JFK) ^ At(C3, ATL)
     ^ At(P1, SFO) ^ At(P2, JFK) ^ At(P3, ATL)
       ^ Cargo(C1) ^ Cargo(C2) ^ Cargo(C3)
       ^ Plane(P1) ^ Plane(P2) ^ Plane(P3)
       ^ Airport(JFK) ^ Airport(SFO) ^ Airport(ATL))
Goal(At(C1, JFK) ^ At(C2, SFO) ^ At(C3, SFO))
```

OPTIMAL PLAN LENGTH FOR PROBLEM 2 IS = 9

Here are the 9 actions:

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)

3. Load(C3, P3, ATL)
4. Fly(P1, SFO, JFK)
5. Fly(P2, JFK, SFO)
6. Fly(P3, ATL, SFO)
7. Unload(C2, P2, SFO)
8. Unload(C1, P1, JFK)
9. Unload(C3, P3, SFO)

c. Air Cargo Problem 3

```
Init(At(C1, SFO) ^ At(C2, JFK) ^ At(C3, ATL) ^ At(C4, ORD)
        ^ At(P1, SFO) ^ At(P2, JFK)
        ^ Cargo(C1) ^ Cargo(C2) ^ Cargo(C3) ^ Cargo(C4)
        ^ Plane(P1) ^ Plane(P2)
        ^ Airport(JFK) ^ Airport(SFO) ^ Airport(ATL) ^ Airport(ORD))
Goal(At(C1, JFK) ^ At(C3, JFK) ^ At(C2, SFO) ^ At(C4, SFO))
```

OPTIMAL PLAN LENGTH FOR PROBLEM 3 IS= 12

Here are the 12 actions:

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)
3. Load(C3, P1, ATL)
4. Load(C4, P2, ORD)
5. Fly(P1, SFO, ATL)
6. Fly(P2, JFK, ORD)
7. Fly(P2, ORD, SFO)
8. Fly(P1, ATL, JFK)
9. Unload(C4, P2, SFO)
10. Unload(C3, P1, JFK)
11. Unload(C2, P2, SFO)
12. Unload(C1, P1, JFK)

II Uninformed Non-heuristic search result metrics

Air Cargo Problem 1						
Search algo	Node Expansions	Goal Tests	New nodes	Time elapsed(seconds)	Plan length	Optimal Plan Len? Y/N
Breadth First Search	43	56	180	**0.033**	**6**	Y
Depth First Graph Search	21	22	84	0.015	20	N
greedy_best_first_graph_search h_1	**7**	**9**	**28**	**0.006**	**6**	**Y**
uniform_cost_search	55	57	224	0.036	6	Y

Air Cargo Problem 2						
Search algo	Node Expansions	Goal Tests	New nodes	Time elapsed(seconds)	Plan length	Optimal Plan Len? Y/N
Breadth First Search	3343	4609	30509	**14.18**	**9**	Y
Depth First Graph Search	**624**	625	5602	**3.18**	619	N
greedy_best_first_graph_search h_1	998	1000	8982	6.59	13	N
uniform_cost_search	4853	4855	44041	49.024	9	Y

Air Cargo Problem 3						
Search algo	Node Expansions	Goal Tests	New nodes	Time elapsed(seconds)	Plan length	Optimal Plan Len? Y/N
Breadth First Search	**14663**	**18098**	**129631**	**100.793**	**12**	**Y**
Depth First Graph Search	**408**	409	3364	**1.63**	392	N
greedy_best_first_graph_search h_1	**5578**	5580	49150	116.48	22	N
uniform_cost_search	18223	18225	159618	417.16	12	Y

Best non-heuristic search:

The best non-heuristic search that is not problem specific here is one that consistently results in an optimal plan length for all three problems and takes least amount of time to arrive at the goal. **Breadth First search** always results in an optimal plan length for all three problems in least time. Greedy Best first graph resulted in an optimal plan length with least amount of time only for problem 1 but does not consistently produce optimal plan length for all problems.

III Informed heuristic search result metrics

Air Cargo Problem 1						
Search algo	Node Expansions	Goal Tests	New nodes	Time elapsed(seconds)	Plan length	Optimal Plan Len? Y/N
Breadth First Search	43	56	180	0.033	6	Y
Depth First Graph Search	21	22	84	0.015	20	N
greedy_best_first_graph_search h_1	7	9	28	0.006	6	Y
uniform_cost_search	55	57	224	0.036	6	Y
astar_search h_1	55	57	224	0.039	6	Y
astar_search h_ignore_preconditions	**41**	**43**	**170**	**0.034**	**6**	**Y**
astar_search h_pg_levelsum	11	13	50	1.44	6	Y

Air Cargo Problem 2						
Search algo	Node Expansions	Goal Tests	New nodes	Time elapsed(seconds)	Plan length	Optimal Plan Len? Y/N
Breadth First Search	3343	4609	30509	14.18	9	Y
Depth First Graph Search	624	625	5602	3.18	619	N
greedy_best_first_graph_search h_1	998	1000	8982	6.59	13	N
uniform_cost_search	4853	4855	44041	49.024	9	Y
astar_search h_1	4853	4855	44041	45.946	9	Y

astar_search h_ignore_preconditions	**1506**	**1508**	**13820**	**13.3897**	**9**	**Y**
astar_search h_pg_levelsum	86	88	839	155.179	9	Y

Air Cargo Problem 3						
Search algo	Node Expansions	Goal Tests	New nodes	Time elapsed(seconds)	Plan length	Optimal Plan Len? Y/N
Breadth First Search	14663	18098	129631	100.793	12	Y
Depth First Graph Search	408	409	3364	1.71	392	N
greedy_best_first_graph_search h_1	5578	5580	49150	116.48	22	N
uniform_cost_search	18223	18225	159618	417.16	12	Y
astar_search h_1	18223	18225	159618	385.649	12	Y
astar_search h_ignore_preconditions	**5118**	**5120**	**45650**	**87.03**	**12**	**Y**
astar_search h_pg_levelsum	404	406	3718	1068.75	12	Y

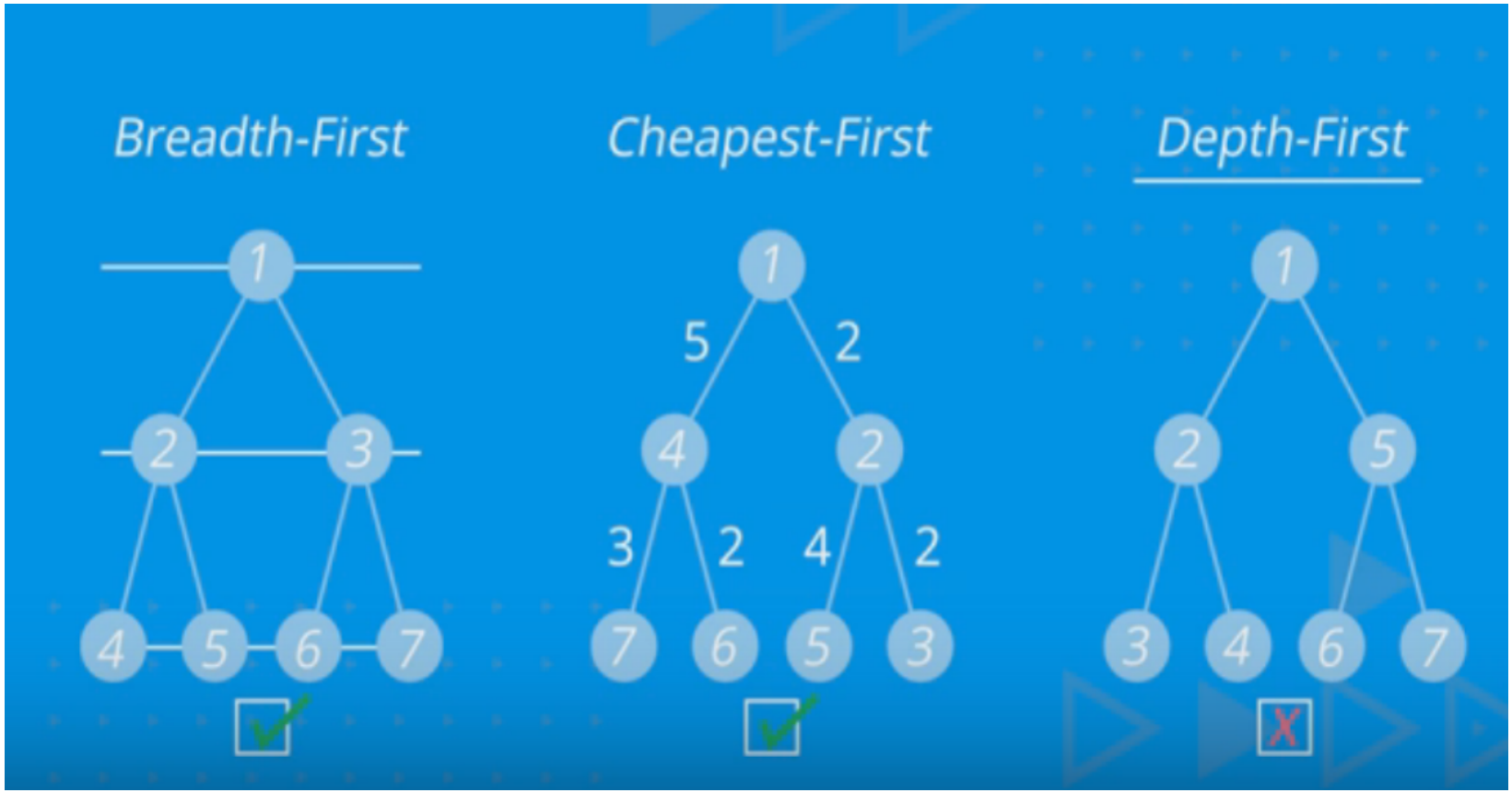
Best heuristic search:

All three A* searches reach the optimal plan length 6, 9, and 12 for problems 1, 2 and 3 respectively.
A* search that ignores preconditons does best for all three air cargo problems.
A* search with level sum heuristic uses the least memory expanding fewer nodes, but takes longer to arrive at the optimal plan. So **A* search that ignores preconditons** is the best heuristic used.

IV Conclusions:

In the case of non-heuristic search, **Breadth first search (BFS)** is the clear winner. Both BFS and Uniform cost search arrive at the optimal plan length but BFS does so faster. Both breadth wise search and least cost searches will return the optimal value for even an infinite graph.

However Depth First search fails to find the optimal length because it works through the tree. It goes through the nodes, deeper nodes first and returns the first goal state value for even an infinite graph that may not be optimal. This is clearly explained in Search Lecture 7, quiz 23 of Udacity AI[7] class.

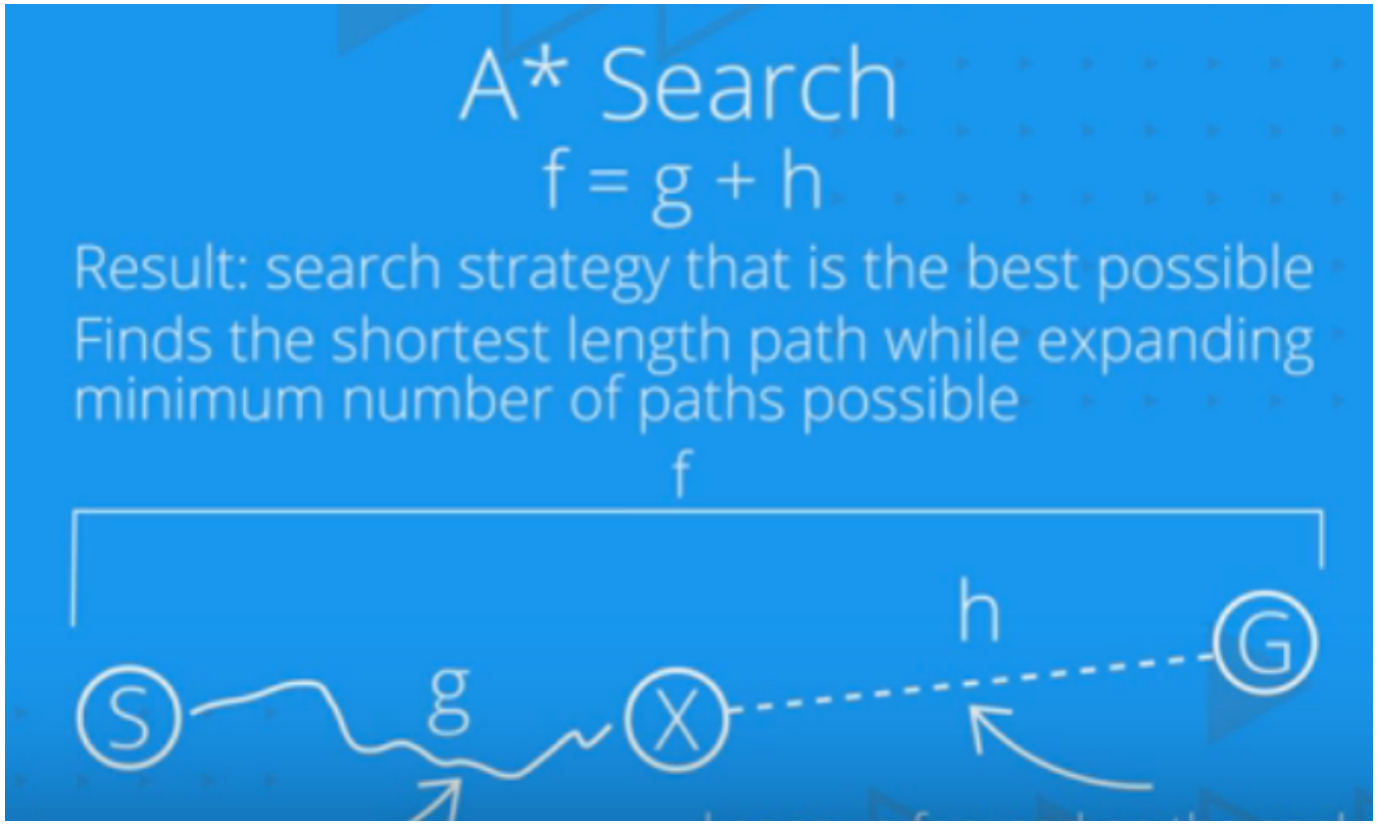


And regarding the heuristic search, all heuristics of an A* search found the optimal plan length for all three problems, because they are all admissible. Each h function doesn't overestimate the distance to the goal. So we evaluated the best heuristic based on the time it took to arrive at the optimal plan. See Lecture 27 in Udacity AI lecture[2].

h_1 returns a constant value of 1. So the path cost g is going to have a bigger weight than the estimated cost h to the goal. Heuristic h is same for all nodes.

h_pg_levelSum following the subgoal independence assumption, returns the sum of the level costs of the goals. This is inadmissible but works well for problems that are largely decomposable[3]. Node expansions are fewer, but the search time is higher.

h_ignore_preconditions every action is applicable in every step. With fewer node expansions than h1 and faster searches it is the clear winner.



Reference:

1. Search Lecture 23 - https://classroom.udacity.com/nanodegrees/nd889/parts/6be67fd1-9725-4d14-b36e-ae2b5b20804c/modules/f719d723-7ee0-472c-80c1-663f02de94f3/lessons/36fc5b2f-6367-4808-b87c-0faa42744994/concepts/4e06a0da-40b1-48a4-a4d3-a95ceebfb468
2. Search Lecture 27 - https://classroom.udacity.com/nanodegrees/nd889/parts/6be67fd1-9725-4d14-b36e-ae2b5b20804c/modules/f719d723-7ee0-472c-80c1-663f02de94f3/lessons/36fc5b2f-6367-4808-b87c-0faa42744994/concepts/fa389fff-33fc-4e9e-be01-4baeaad46b4d
3. AIMA Chapter 11, Page 398 - http://aima.cs.berkeley.edu/2nd-ed/newchap11.pdf