Planning Search

We evaluate the performances of six search algorithms on cargo-transport problems air_cargo_p1, air_cargo_p2, and air_cargo_p3. Three uniformed search and three A* heuristic search algorithms are selected. The results are summarized in the following:

(1) Number of Expansions:

	air_cargo_p1	air_cargo_p2	air_cargo_p3
Uninformed Search			
breadth_first_search	43	3343	14663
depth_first_graph_search	21	624	408
uniform_cost_search	55	4853	18151
A* Heuristic Search (astar_search)			
h_1	55	4853	18151
h_ignore_preconditions	41	1506	5118
h_pg_levelsum	11	86	404

(2) Number of Goal Tests:

	air_cargo_p1	air_cargo_p2	air_cargo_p3
Uninformed Search			
breadth_first_search	56	4609	18098
depth_first_graph_search	22	625	409
uniform_cost_search	57	4855	18153
A* Heuristic Search (astar_search)			
h_1	57	4855	18153
h_ignore_preconditions	43	1508	5120
h_pg_levelsum	13	88	406

(3) Time Elapsed (measured in seconds):

	air_cargo_p1	air_cargo_p2	air_cargo_p3
Uninformed Search			
breadth_first_search	0.0450	29.7545	143.7793
depth_first_graph_search	0.0248	6.9916	2.4530
uniform_cost_search	0.0747	100.4771	680.3170
A* Heuristic Search (astar_search)			
h_1	0.0618	63.7143	617.4484
h_ignore_preconditions	0.0714	22.1625	136.8808
h_pg_levelsum	3.0630	322.8597	2223.2931

(4) Optimality (Plan Length):

Before evaluating the optimality of the algorithm, we would like to identify a lower bound estimate on the optimal plan length (number of actions required to achieve the goal) on each of the problem.

A. air_cargo_p1:

initial: At(C1, SFO) ∧ At(C2, JFK) ∧ At(P1, SFO) ∧ At(P2, JFK)

goal: At(C1, JFK) ∧ At(C2, SFO)

Initially, the two cargos are not located at their final destinations and the two cargos are located in different airports. Therefore, the optimal plan should at least include the Load and Unload actions for each cargo (4 actions) and one Fly action for each cargo (2 actions). In this case, the optimal plan length should be at least 6.

An example of the optimal plan is:

Load(C1, P1, SFO), Load(C2, P2, JFK), Fly(P2, JFK, SFO), Unload(C2, P2, SFO), Fly(P1, SFO, JFK), Unload(C1, P1, JFK)

B. air_cargo_p2:

initial: At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ At(P3, ATL)

goal: At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO)

Initially, the three cargos are not located at their final destinations and the three cargos are located in different airports. Therefore, the optimal plan should at least include the Load and Unload actions for each cargo (6 actions) and one Fly action for each cargo (3 actions). In this case, the optimal plan length should be at least 9.

An example of the optimal plan is:

Load(C1, P1, SFO), Load(C2, P2, JFK), Load(C3, P3, ATL), Fly(P2, JFK, SFO), Unload(C2, P2, SFO), Fly(P1, SFO, JFK), Unload(C1, P1, JFK), Fly(P3, ATL, SFO), Unload(C3, P3, SFO)

C. air_cargo_p3:

initial: At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(C4, ORD) ∧ At(P1, SFO) ∧ At(P2, JFK)

Goal: At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO)

Initially, the four cargos are not located at their final destinations and the four cargos are located in different airports. Therefore, the optimal plan should at least include the Load and Unload actions for each cargo (8 actions) and one Fly action for each cargo (4 actions). In this case, the optimal plan length should be at least 12.

An example of the optimal plan is:
Load(C1, P1, SFO), Load(C2, P2, JFK), Fly(P2, JFK, ORD), Load(C4, P2, ORD), Fly(P1, SFO, ATL), Load(C3, P1, ATL), Fly(P1, ATL, JFK), Unload(C1, P1, JFK), Unload(C3, P1, JFK), Fly(P2, ORD, SFO), Unload(C2, P2, SFO), Unload(C4, P2, SFO)

The plan lengths for each of the algorithm are reported in the following. It is noted that the plan lengths of some of the search algorithms achieve the estimated lower bounds. Therefore, the optimality of these search algorithms is claimed. Other searching algorithms having a longer plan length are suboptimal.

Plan Length:

	air_cargo_p1	air_cargo_p2	air_cargo_p3
Uninformed Search			
breadth_first_search	6	9	12
depth_first_graph_search	20	619	392
uniform_cost_search	6	9	12
A* Heuristic Search (astar_search)			
h_1	6	9	12
h_ignore_preconditions	6	9	12
h_pg_levelsum	6	9	12

Discussions:

1. **Comparison among uninformed search methods:**
   Among the three uniformed search methods and for all three cargo-transport problems, depth_first_graph_search expands the fewest number of nodes and performs the least number of goal tests. It also spends the least amount of time in finding solutions for all three problems since it only expands a small portion of the search tree and fortunately the goal state resides in the branch where it

selects for searching. However, depth_first_graph_search only achieves suboptimal solutions. It provides plan length equal to 20, 619, and 392 for problems air_cargo_p1, air_cargo_p2, and air_cargo_p3, respectively.

The optimality of uniform_cost_search is guaranteed. breadth_first_search also achieves an optimal solution since the costs of the actions are the same. Nevertheless, compared with depth_first_graph_search, it takes more time and node expansions for the two algorithms to find the solution. uniform_cost_search takes more time than tbreadth_first_search as it needs to find the action with the minimum cost from a priority queue. On the other hand, breadth_first_search simply selects the action at the head of the queue.

2. **Comparison among A* search methods with different heuristic functions:**
   The A* search guarantees the optimality of the solution if the underlying heuristics are admissible. Hence, it is not surprising that h_1 and h_ignore_preconditions achieves the optimal solution. We also notice that although h_pg_levelsum is not admissible in general, it still provides an optimal plan for the three problems.

   h_pg_levelsum takes the longest time to find the optimal solution among the three heuristics since it is more time-consuming to evaluate the level sum heuristic. For air_cargo_p3, h_pg_levelsum search method takes over 30 minutes to find the solution.

   From the memory efficiency point of view, h_pg_levelsum expands the fewest number of nodes since this heuristic is closer to the actual cost. In this case, nodes inducing longer plan lengths are not expanded. On the other hand, the h_1 search method is the fastest one for air_cargo_p1. However, as the problem size grows, it expands a total of 18151 nodes, which is over 40 times larger than that expanded by h_pg_levelsum.

   h_ignore_preconditions provides a heuristic easier to evaluate than h_pg_levelsum and more accurate than h_1. While the number of nodes expanded by h_ignore_preconditions is still 10 times larger than h_pg_levelsum for air_cargo_p2 and air_cargo_p3, it is only 1/3 of that expanded by h_1. For air_cargo_p1, h_ignore_preconditions is faster than h_pg_levelsum by 3 seconds. For air_cargo_p2 and air_cargo_p3, h_ignore_preconditions is the fastest among all the three A* search based algorithms. It even finds the optimal solution 10 times faster than h_pg_levelsum. As a result, considering the tradeoffs between time to expand new nodes and time to evaluate the heuristics, h_ignore_preconditions seems to be a promising selection of heuristics.

3. **Comparing the A* search methods with uninformed search methods:**
   In this section, we select the best heuristic h_ignore_preconditions for A* search method and compare it with uninformed search methods. For air_cargo_p1, the number of node expanded by h_ignore_preconditions is close to that expanded by breadth_first_search (smallest number of expanded nodes among all uninformed search methods that provide optimal solutions). The elapsed times of the two algorithms are approximately the same. However, for larger problem sizes (air_cargo_p2 and air_cargo_p3), h_ignore_preconditions expands only 1/3 of the nodes when compared with breadth_first_search. The elapsed time of h_ignore_preconditions is slightly better than breadth_first_search for air_cargo_p2 (22 secs v.s. 29 secs) and air_cargo_p3 (136 secs v.s. 143 secs). In conclusion, the A* search with heuristic h_ignore_preconditions outperforms all the other three uninformed heuristics for air_cargo_p2 and air_cargo_p3.