Air Cargo Heuristic Analysis

By Will Russell

The problem addressed in this project was a deterministic logistics problem for Air Cargo Transport based on the problem presented in Chapter 10 of *Artificial Intelligence* by Peter Norvig and Stuart Russell.

Discussion

Of the non-heuristic search strategies breadth first search, uniform cost search, and depth first search were able to find the optimal solution to all cargo problems. For the heuristic strategies, only A* search utilizing the base heuristic 'h_1' and 'h_ignore_preconditions' were able to run to completion for all 3 problems. 'h1_pg_levelsum' was the slowest of the 3 and was not able to run to completion within the allotted 10 minute time window. In the case of problem 1, it seems that due to the limited complexity of the problem, breadth first was best for finding the optimal solution in the shortest time window. As complexity increased in poblems 2 and 3, the heuristic functions were able to perform much better and A* ignore_preconditions seemed to perform the best on these problems.

Breadth first search was able to find an optimal solution in a reasonable time frame for all solutions, though its performance lagged behind the heuristic methods as complexity increased. Finding the shortest path work guaranteed the optimal solution, however it may not be advantageous to take the same approach as the number of possibilities increase due to the time loss.

Depth first graph search found a solution to the problem quickly, however it lacked the optimality of the other options in all problems. This is due to the way in which it explores the nodes, going as deep as possible to the left, even though the goal may be to the right.

Uniform cost search found the optimal solution in all cases, though it was not the fastest to the goal in any solution. An interesting characteristic of this search was that the amount of time that it took to reach the goal did not increase as quickly as happened with breadth first search. In both problems 2 and 3, uniform cost search was faster than breadth first search. This is probably due to the way in which uniform-cost search expands nodes in order of their optimal cost [1].

Greedy best first graph search was optimal only in the first case, though it did manage to find the solution the fastest for problems 2 and 3. This is due to the nature in which it tries to expand the node closes to the goal, in hopes of achieving a solution quickly. Due to its inconsistent nature it does not seem like this would be a good choice when the optimal solution is required, however it may be ideal for situations in which a quick path to the goal is needed.

Between the heuristic functions A* h_1 and A* ignore_preconditions, the latter seemed to perform the best. This is most likely due to the dropping of the preconditions for each action in the seach, allowing the goal condition to be achieved much more rapidly.

Planning Problems

```
Action(Load(c, p, a),
        PRECOND: At(c, a) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
        EFFECT: ¬ At(c, a) ∧ In(c, p))
Action(Unload(c, p, a),
        PRECOND: In(c, p) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
        EFFECT: At(c, a) ∧ ¬ In(c, p))
Action(Fly(p, from, to),
        PRECOND: At(p, from) ∧ Plane(p) ∧ Airport(from) ∧ Airport(to)
        EFFECT: ¬ At(p, from) ∧ At(p, to))
```

Problem 1 Initial State and Goal:

```
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))
```

Problem 2 Initial State and Goal:

```
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))
```

Problem 3 Initial State and Goal:

```
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 lengths of 6, 9, and 12 actions exist for the respective problems 1, 2, and 3. Due to the number of expansions, some of the search implementations would not run to completion for problems 2 and 3. The results which would not run to completion have been excluded from the following tables.

Results

Problem 1 Table

Approach	Expansions	Goal Tests	New Nodes	Min Solutions	Time Elapsed
breadth_first_search	43	56	180	6	0.03489247
breadth_first_tree_search	1458	1459	5960	6	0.989501679
depth_first_graph_search	12	13	48	12	0.007512835
depth_limited_search	101	271	414	50	0.092639288
uniform_cost_search	55	57	224	6	0.03620735
recursive_best_first_search	4229	4230	17029	6	2.74271472
greedy_best_first_graph_search	7	9	28	6	0.004805446
astar_search: h_1	55	57	224	6	0.041226259
astar_search : ignore_precond	41	43	170	6	0.045083694
astar_search: pg_levelsum	55	57	224	6	3.007666228

Problem 2 Table

Approach	Expansions	Goal Tests	New Nodes	Min Solutions	Time Elapsed
breadth_first_search	3346	4612	30534	9	13.40096497
depth_first_graph_search	1391	1392	12432	1288	10.27986239
uniform_cost_search	4605	4607	41839	9	11.0179724
greedy_best_first_graph_search	479	481	4306	20	1.140722635
astar_search: h_1	4605	4607	41839	9	11.06463196
astar_search: ignore_precond	1311	1313	11989	9	4.705543095

Problem 3 Table

Approach	Expansions	Goal Tests	New Nodes	Min Solutions	Time Elapsed
breadth_first_search	14491	17947	128184	12	103.6023143
depth_first_graph_search	1948	1949	16253	1878	18.97564004
uniform_cost_search	17783	17785	155920	12	53.97837301
greedy_best_first_graph_search	4031	4033	35794	22	12.30699183
astar_search: h_1	17783	17785	155920	12	52.73391892
astar_search: ignore_precond	5003	5005	44586	12	20.79267321

Optimal Series of Actions

Problem 1:
Breadth First Search

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

Problem 2:
A* Search with ignore_preconditions heuristic.

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

Problem 3:
A* Search with ignore_preconditions heuristic.

- Load(C2, P2, JFK)
- Fly(P2, JFK, ORD)
- Load(C4, P2, ORD)
- Fly(P2, ORD, SFO)
- Unload(C4, P2, SFO)
- Load(C1, P1, SFO)
- Fly(P1, SFO, ATL)
- Load(C3, P1, ATL)
- Fly(P1, ATL, JFK)
- Unload(C3, P1, JFK)
- Unload(C2, P2, SFO)
- Unload(C1, P1, JFK)

Citations

[1] S. J. Russell, P. Norvig(2010*), Artificial Intelligence: A Modern Approach(3rd Edition).* Pearson.