Written Analysis of Planning Problems

Planning Problems

Air Cargo Action Schema:

```
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))
```

Non-heuristic Search Result Metrics

Three different non-heuristic searches were run on three different problems: breadth first search, depth first graph search, and uniform cost search. The results of these runs are in the tables below.

Air Cargo Problem 1:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
breadth_first_search	43	56	180	**6**	0.0467
depth_first_graph_search	**12**	**13**	**48**	12	**0.0137**
uniform_cost_search	55	57	224	**6**	0.0433

Air Cargo Problem 2:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
breadth_first_search	3343	4609	30509	**9**	21.0615
depth_first_graph_search	**582**	**583**	**5211**	575	**5.2377**
uniform_cost_search	4761	4763	43206	**9**	19.984

Air Cargo Problem 3:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time

breadth_first_search	14491	17947	128184	**12**	154.82
depth_first_graph_search	**1948**	**1949**	**16253**	1878	**30.706**
uniform_cost_search	17797	17799	156081	**12**	78.546

The results demonstrate that depth first graph search takes the least amount of time to execute, however, it also provides the worst plan lengths. Breadth first search and uniform cost search both found an optimal plan length, with uniform cost search executing faster and breadth first search searching less nodes.

Breadth first search is optimal and will always find the optimal path because it expands the shortest paths first, whereas depth first graph search will attempt the farthest paths first and might miss out on the shorter paths because it is not complete [1]. Uniform cost search also finds the optimal path because it will not search branches that cost more than the shortest path [1].

Based on these results, uniform cost search appears to be the best choice for non-heuristic search, since it not only executes faster, it also will always provide an optimal plan.

Heuristic Search Result Metrics

Next, three different heuristic searches were applied to the same problems: A* search with h1 heuristic, A* search with ignore precondition heuristic, and A* search with level sum heuristic.

Air Cargo Problem 1:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
astar_search with h_1	55	57	224	**6**	0.063
astar_search with h_ignore_precondtions	41	43	170	**6**	**0.06**
astar_search with h_pg_levelsum	**11**	**13**	**50**	**6**	1.321

Air Cargo Problem 2:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
astar_search with h_1	4761	4763	43206	**9**	16.492

astar_search with h_ignore_precondtions	1450	1452	13303	**9**	**6.8644**
astar_search with h_pg_levelsum	**86**	**88**	**841**	**9**	236.206

Air Cargo Problem 3:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
astar_search with h_1	17797	17799	156081	**12**	82.528
astar_search with h_ignore_precondtions	5034	5036	44886	**12**	**26.354**
astar_search with h_pg_levelsum	**315**	**317**	**2904**	**12**	1592

All searches were able to return an optimal plan, however some took a significant period of time to execute. Level sum performed the least amount of expansions, but the execution time was far longer than the other heuristics. Based on these results, A* search with the ignore preconditions heuristic is the best choice, as it took the least amount of time to execute.

Optimal Plans:

Problem 1:

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

Problem 2:

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

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

Problem 3:

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

References:

1. AIND Video Lessons. Lesson 10 - Search.