Logic and Planning Heuristic Analysis

The first part of this assignment was to run various uninformed (non-heuristic) searches on three different problems. Before I begin my analysis, I'm going to list the optimal plans of each problem, so that I can reference them in comparison to various search methods.

Problem 1: Plan length: 6
Load(C1, SFO, P1)
Fly(P1, SFO, JFK)
Load(C2, JFK, P2)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Problem 2: Plan length: 9
Load(C1, SFO, P1)
Fly(P1, SFO, JFK)
Load(C2, JFK, P2)
Fly(P2, JFK, SFO)
Load(C3, ATL, P3)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Problem 3: Plan length: 12
Load(C2, JFK, P2)
Fly(P2, JFK, ORD)
Load(C4, ORD, P2)
Fly(P2, ORD, SFO)
Load(C1, SFO, P1)
Fly(P1, SFO, ATL)
Load(C3, ATL, P1)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)

In the first problem, breadth first search found the optimal plan (with length 6), and the time it took was 0.0588 seconds, which was one of the fastest times for problem 1. I think this was due to the low complexity of the problem. Depth first graph search also found a solution, but it wasn't the optimal one. In this case, it took 0.0273 seconds, but had a solution of

length 12. This difference in time complexity is common, since a depth first search doesn't find guarantee an optimal solution unless iterative deepening is used. The third uninformed search, uniform cost, was the best in terms of time complexity + optimal solution. However it should be noted that there were 55 expansions (10 more than breadth first), even though it was faster. The .01 difference in time is very neglectable for such a simple problem, but this will likely become an issue as complexity increases.

In the second problem, our breadth first search jumped to 33.3 seconds to complete, expanding 3,343 times. At this point, it's already easy to tell that expanding almost every node is taking a tax on our system. Depth first search did loads better with a time of 5.77 seconds, but showed its true weakness as its plans length was 575, which is over 60 times our optimal solution of 9. Once again, uniform cost was outperformed the other two overall, finding an optimal solution and completing in 16 seconds.

In our third problem, the cost of breadth first was most apparent, expanding over 14,000 nodes and taking 324 seconds. This is expected because even though there is just one more cargo and airport, and one less plane than problem two, this results in exponentially more branches to have to cover. As expected, depth first search took a fraction of the time, but had an unusable plan of 1,451 steps, compared to the optimal 12. Once again, uniform cost search provided the best of both worlds, with a shorter time and optimal solution.

It's worth noting at this point that I originally tried running breadth first tree searches and depth limited searches on problems 2 and 3, but had to stop them because it was taking too long. It was clear from early on that this would be the case, since they both expand exponentially more nodes than both breadth first search and depth first search. In problem 1, BFTS took 1 second, but ran 1,450 expansions, compared to BFS expanding 43 times.

My next comparison is with heuristic search using A*. With the ignore preconditions heuristic, problem 1 took 0.13 seconds with 41 expansions, while the level sum heuristic took 0.9 seconds with 11 expansions. At first I thought that this discrepancy would disappear as the problem got more complex. However, it only got more prominent in problem two. Ignoring preconditions took 7.81 seconds with 1,450 expansions, while level sum took 173 seconds with only 86 expansions. For completeness sake, I'll include that ignore preconditions took 17 seconds vs. level sum's 825, with 5,040 vs 316 expansions. The trend here is obvious, but why? Both are returning optimal plans, but the one with way more expansions and nodes seems to be way faster. Well in this case, level sum has to create a planning graph, looping through every single goal, level and state every node we explore. Although there are many more nodes in with the ignore preconditions heuristic, the time it takes to compute its heuristic is significantly less than that of level sum.

All things considered, having the ignore preconditions heuristic is significantly better than the best non-heuristic, uniform cost search. Assuming the plan is more complex than our first problem, the A* search added with any relatively low constant cost should outperform a search using no heuristic. I think one of the important take-aways is that heuristics help reduce time complexity by reducing the number of nodes that have to be searched, but if the computation of the heuristic is too expensive (especially when not constant), then the algorithm will take too long, even though there are so many less nodes to explore. Lastly, I'll conclude that although keeping time complexity low, while achieving an optimal plan was the goal in this case, there

exists plenty of trees in which the ignore preconditions heuristic could be limited by memory. In those situations, the tradeoff of completion time vs required memory allocation could easily be worth it.