heuristic analysis

Problem 1

Problem 1 Results

Search Alg	Expansions	Goal Tests	New Nodes	Plan Length	Time
Breadth First Search	43	56	180	6	0.036
Breadth First Tree Search	1458	1459	5960	6	0.947
Depth First Graph Search	21	22	84	20	0.0175
Depth First Limited	101	271	414	50	0.0927
Uniform Cost Search	55	57	224	6	0.052
Recursive Best First W/H1	4229	4230	17023	6	2.749
Greedy Best First W/H1	7	9	28	6	0.0058
AStar Search W/H1	55	57	224	6	0.0575
AStar Search W/H*Ignore*Precon	41	43	170	6	0.0800
AStar Search W/H*pg*levelsum	10	12	42	6	1.045

Problem 1 Optimal Path

Plan length 6

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

Problem 1 Comparison

1. BFS - Breadth First Search looks liks to be the winner out of the non-heuristic search algororithms. It

acheieved the shortest pathway in a short amount of time. However since the problem was small, I think it may be too early to celebrate BFS.

2. DFS - Depth First Search is much faster than BFS, completing in a fraction of the time. However we may have to ignore this feat because the pathway it came up with is not near the optimal plan. In fact is is more than triple.
3. Uniform Cost Search achieves the optimal plan as well as being cost effective. I think this may be our winner for non-heuristic.
4. Both Astar with ignore*preconditions and level*sum heuristics acheived the optimal plan, however ignore precondition completed with a much faser time. As a takeaway though, the pg*level*sum search used much less space. This may make more of an impact with larger problems.
5. The best heuristic algorithm looks to be astar with ignore_preconditions. However uniform cost search seems to be the best search overall for this problem.

Problem 2 Results

Search Alg	Expansions	Goal Tests	New Nodes	Plan Length	Time
Breadth First Search	3343	4609	30509	9	13.157
Breadth First Tree Search	--	--	--	--	--
Depth First Graph Search	624	625	5602	619	3.336
Depth First Limited	--	--	--	--	--
Uniform Cost Search	4852	4854	44030	9	44.200
Recursive Best First W/H1	--	--	--	--	--
Greedy Best First W/H1	990	992	8910	17	6.697
AStar Search W/H1	4852	4854	44030	9	44.243
AStar Search W/H*Ignore*Precon	1506	1508	13820	9	14.888
AStar Search W/H*pg*levelsum	724	726	6819	9	983.415

Problem 2 Optimal Path

PLan length 9

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

Problem 2 Comparison

1. BFS - Breadth First Search creates much more new nodes to call it an optimal search technique, especialy when it takes 13 seconds to finish. It does however eventually come to the optimal path. I get the feeling that the more complex the problem, the cost to run BFS will increase almost exponentially in both space and time.
2. DFS - Depth First Search completed in roughly a quarter of the time of BFS, however the path length is 619! Thats alot of flying and redundant cargo handling. I feel like I need to dismiss DFS almost as a default because of the terrible results, at this point speeds almost doesnt even matter.
3. Uniform Cost Search actually did worse than BFS this time, taking roughly 3 times as long. Even though BFS performed terrible, amongist BFS, DFS and UCS, I think BFS did the better job.
4. Both Astar with ignore*preconditions and level*sum heuristics acheived the optimal plan, however ignore precondition completed with a much faser time. This time around, the Astar with pg*level*sum took 983 seconds! I think this is going to be a big problem scaling any larger
5. The best heuristic algorithm looks to be a tie between BFS and A*/ignore_precondition with the former being slighly faster however the latter consuming less space.

Problem 3 Results

Search Alg	Expansions	Goal Tests	New Nodes	Plan Length	Time
Breadth First Search	14663	18098	129631	12	100.44
Breadth First Tree Search	--	--	--	--	--
Depth First Graph Search	408	409	3364	392	1.766
Depth First Limited	--	--	--	--	--
Uniform Cost Search	18235	18237	159716	12	405.808
Recursive Best First W/H1	--	--	--	--	--
Greedy Best First W/H1	5614	5616	49429	12	104.201
AStar Search W/H1	18235	18237	159716	12	466.430
AStar Search W/H*Ignore*Precon	5118	5120	45650	12	100.09
AStar Search W/H*pg*levelsum	--	--	--	--	--

Problem 3 Optimal Path

Plan length 12

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

Problem 3 Comparison

1. BFS - Breadth First Search creates almost 130k new nodes however does get to the optimal path of 12 in a time of roughly 100 seconds. I feel that this may be a contender just like the last problem .
2. DFS - Depth First Search, once again is super fast however gives us a plan length of 392. I wonder if there is a way to take the path that DFS produces and trim off the fat in another iteration in some fancy optimzation operation. But for now the path is just huge.
3. Uniform Cost Search did much worse that BFS this time and even has the largest space requirements.
4. Only Astar with ignore*preconditions completed operation within the 10 minute alotted time window. However it did complete as the fasted search on the grid as well as the second smallest space requirements, bravo. pg*level_sum A* did not complete within that time window and must forfeit.

5. The best heuristic algorithm looks to be a tie between BFS and A*/ignore_precondition again with the latter inching ahead as the sligtly faster but also much less space requirements.

Conclusion

Starting out with a small problem, it looks like the BFS and UCS searches are to be reckoned with, however as the problems get more involved with more requirements to solve, the space requirements on BFS and UCS just explodes and UCS seems to get much worse in processing time. Depth first search is always very fast however leads us to a path that is orders of magnitude higher than the optimal path. Depending on the problem, which I assume to be a large marjority, would be unusable. The AStar Search with h*ignore*precondition heuristic looks to be the most promising search as it looks to be consistently doing a reasonable performance in both time and space. I forecast that if we were to make higher complex problems that this search would continue to perform reasonably well.