Nicholas Pickering – AIND Project 3

Optimal Plans for Each Cargo Problem

	Optimal Execution Plan
Problem 1	Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK)
Problem 2	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)
Problem 3	Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P1, ATL, JFK) Fly(P2, ORD, SFO) Unload(C4, P2, SFO) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)

Non-Heuristic Uninformed Planning Searches Performance Comparison

	Search Method	Expansions	Goal Tests	New Nodes	Plan Length	Elapsed Time
Problem 1	breadth_first	43	56	180	6	0.02 s
	depth_first	21	22	84	20	0.01 s
	uniform_cost	55	57	224	6	0.02 s
Problem 2	breadth_first	3343	4609	30509	9	7.52 s
	depth_first	624	625	5602	619	1.88 s
	uniform_cost	4835	4837	43877	9	6.52 s
Problem 3	breadth_first	144663	18098	129631	12	56.17 s
	depth_first	408	409	3364	392	0.98 s
	uniform_cost	18223	18225	159618	12	28.67 s

In each problem, the depth-first search returned its result in the fastest amount of time – however, the execution plan generated is far from optimal, with an order of magnitude greater execution plan lengths.

Between breadth-first and uniform-cost searches, it seems that uniform-cost searches result in similar execution plans, while taking less time. The uniform-cost search seems to perform more poorly with regards to expansions, goal tests and new nodes for the simpler problems – 1, and 2. However, for Problem 3's complexity the uniform-cost search was able to drastically reduce the number of expansions required.

According to *Artificial Intelligence: A Modern Approach[1]*, if the path cost is a non-decreasing function of the depth of the goal nodes – resulting in breadth-first guaranteeing an optimal solution. The time and space complexity of the breadth-first search is generally poor.

A* Search with Automatic Heuristic Performance Comparison

	A* Heuristic	Expansions	Goal Tests	New Nodes	Plan Length	Elapsed Time
Problem 1	Constant	55	57	224	6	0.02 s
	Ignore Preconditions	41	43	170	6	0.02 s

	Level Sum	11	13	50	6	0.30 s
Problem 2	Constant	4835	4837	43877	9	6.42 s
	Ignore Preconditions	1450	1452	13303	9	2.33 s
	Level Sum	86	88	841	9	26.11 s
Problem 3	Constant	18223	18225	159618	12	29.43 s
	Ignore Preconditions	5040	5042	44944	12	9.43 s
	Level Sum	312	314	2872	12	124.12 s

Each A* heuristic search resulted in an optimal execution plan, as to be expected. According to *Artificial Intelligence: A Modern Approach[2]*, an admissible heuristic is one which does not overestimate the cost of a transition between two states.

The Level Sum heuristic takes advantage of a Planning Graph. Planning Graphs are a rich source of information about the problem being solved. The Planning Graph is constructed in such a way that estimating how far any goal literal is away from a given state can be done admissibly and relatively accurately.

The Level Sum heuristic significantly outperforms every other search approach in every metric but time performance.

References

[1] *Artificial Intelligence: A Modern Approach* (Russell, Norvig) Chapter 3: Solving Problems By Search (pg 81-83)
[2] *Artificial Intelligence: A Modern Approach* (Russell, Norvig) Chapter 10: Classical Planning (pg 376-382)