PLANNING SEARCH HEURISTIC ANALYSIS

In this project, we materialized classes and functions to solve deterministically logistics planning problems for Air_Cargo transport system using the search algorithms like those that we use for navigation in order to find out the optimal solution plans for each problem. Three problems are sorted in this project

- 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 aand 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))

We have implemented functions like:load_actions(),unload_action(),fly_actions()

With this 3-problem set, Breadth First Search and Uniform Cost Search are the only two uninformed search strategies that systematically yield an optimal action plan.

For problems 2 and 3, the Depth First Graph Search plan lengths are so much longer than the optimal path length that it wouldn't make sense to use this search strategy. Greedy Best First Graph Search is the best alternative. In problems 1 and 2, it manages to find the optimal path.

CONCLUSION

Among the non-heuristic search functions, breadth first search and uniform cost search algorithms are the most optimal.

Though Depth first graph search is faster, it doesn't give an optimal solution. Similarly, we can see that depth limited

search is not optimal because many nodes are visited multiple times as it doesn't keep track of the visited paths/nodes.

The Planning problem is solved with A* search using 2 automatic heuristics - Ignore Preconditions and Level-sum heuristics. The comparison table is laid out below. It is very clear that though both give plan length of equal

size, the Level Sum (planning graph) heuristic is better in terms of node expansions and goal tests.

The most optimal solution in terms of node expansions is found using A-star search with planning graph and Level Sum heuristic.