PROJECT REPORT

Date: April 29, 2020
Project: Build a Forward Planning Agent
School: Udacity
Written By: Hardik Trivedi

AIR CARGO TRANSPORT PROBLEM

This project deals with an air cargo transport problem that involves loading and unloading cargo onto and from planes that fly between airports. Planning ("the task of coming up with a sequence of actions that will achieve a goal") is the central concept for this project and the air cargo transport problem is a complex planning problem. The main task of the project is to implement a planning graph and heuristics. A planning graph is a special data structure that consists of a sequence of levels that correspond to time steps in the plan. Each level consists of a set of literals and a set of actions. Planning graphs work only for propositional planning problems.

AIR CARGO TRANSPORTATION PROBLEM USING PROPOSITIONAL LOGIC

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))

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))

FUNCTIONS TO IMPLEMENT

Mutual Exclusion: A mutually exclusive (or mutex) relation holds between two *actions or literals* at a given level if any of the following conditions holds for the respective layers:

Action Layer:

- *Inconsistent effects:* one action negates an effect of the other.
- *Interference:* one of the effects of one action is the negation of a precondition of the other.
- *Competing needs:* one of the preconditions of one action is mutex with a precondition of the other.

Literal Layer:

- *Inconsistent support:* each possible pair of actions that could achieve the two literals is mutex.
- *Negation: a* mutex relation holds between two *literals* at the same level if one is the negation of the other.

Heuristics: To estimate costs of a conjunction of goals, there are three simple approaches.

Level sum heuristic: following the sub-goal independence assumption, returns the sum of the level costs of the goals.

Max-level heuristic: takes the maximum level cost of any of the goals.

Set-level heuristic: finds the level at which all the literals in the conjunctive goal appear in the planning graph without any pair of them being mutually exclusive

TABLE OF SEARCHES FOR ALL PROBLEMS

Problem 1	Algorithm	Actions	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (sec)
Uninformed Search	BFS	20	43	56	178	6	0.00636384
Uninformed Search	DFS	20	21	22	84	20	0.00350680
Uninformed Search	UCS	20	60	62	240	6	0.01040059
Informed Search - Heuristic	GFS - Unmet Goals	20	7	9	29	6	0.00170123
Informed Search - Heuristic	GFS - Level Sum	20	6	8	28	6	0.26251569
Informed Search - Heuristic	GFS - Max Level	20	6	8	24	6	0.08370497
Informed Search - Heuristic	GFS - Set Level	20	6	8	24	6	0.19574776
Informed Search - Heuristic	A* - Unmet Goals	20	50	52	206	6	0.01008470
Informed Search - Heuristic	A* - Level Sum	20	28	30	122	6	0.67632271
Informed Search - Heuristic	A* - Max Level	20	43	45	180	6	0.30692420
Informed Search - Heuristic	A* - Set Level	20	38	40	158	6	0.68168465

Problem 2	Algorithm	Actions	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (sec)
Uninformed Search	BFS	72	3343	4609	30503	9	2.05877544
Uninformed Search	DFS	72	624	625	5602	619	3.15469722
Uninformed Search	UCS	72	5154	5156	46618	9	3.41987499
Informed Search - Heuristic	GFS - Unmet Goals	72	17	19	170	9	0.01927482
Informed Search - Heuristic	GFS - Level Sum	72	9	11	86	9	5.58580563
Informed Search - Heuristic	GFS - Max Level	72	27	29	249	9	3.72387778
Informed Search - Heuristic	GFS - Set Level	72	27	29	250	9	12.19014081
Informed Search - Heuristic	A* - Unmet Goals	72	2467	2469	22522	9	2.27893911
Informed Search - Heuristic	A* - Level Sum	72	357	359	3426	9	143.21350876
Informed Search - Heuristic	A* - Max Level	72	2887	2889	26594	9	360.79754371
Informed Search - Heuristic	A* - Set Level	72	1832	1834	16780	9	762.18624372

Problem 3	Algorithm	Actions	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (sec)
Uninformed Search	BFS	88	14663	18098	129625	12	10.88364890
Uninformed Search	UCS	88	18510	18512	161936	12	14.79693428
Informed Search - Heuristic	GFS - Unmet Goals	88	25	27	230	15	0.03660094
Informed Search - Heuristic	GFS - Level Sum	88	14	16	126	14	12.50949403
Informed Search - Heuristic	A* - Unmet Goals	88	7388	7390	65711	12	8.57417962
Informed Search - Heuristic	A* - Level Sum	88	369	371	3403	12	227.87222197

Problem 4	Algorithm	Actions	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (sec)
Uninformed Search	BFS	104	99736	114953	944130	14	98.22587385
Uninformed Search	UCS	104	113339	113341	1066413	14	119.03683616
Informed Search - Heuristic	GFS - Unmet Goals	104	29	31	280	18	0.06107276
Informed Search - Heuristic	GFS - Level Sum	104	17	19	165	17	22.54479486
Informed Search - Heuristic	A* - Unmet Goals	104	34330	34332	328509	14	56.52757967
Informed Search - Heuristic	A* - Level Sum	104	1208	1210	12210	15	1267.53386178

Certain things stand out as we look at the results above:

1. Depth First Search (DFS) is not only too expensive and inefficient, but gives incorrect plans such as Fly prior to Load and Fly prior to Unload. DFS can be discarded for this problem.
2. Uniform Cost Search (UCS) creates highest number of nodes and expansions, therefore this cannot be an ideal algorithm to use.
3. Greedy First Search (GFS) is the most optimal choice for problems with less amount of actions. However, after crossing a certain threshold of actions, it appears to be less accurate and generates a less optimal plan. Although it appears that with a better heuristic, the performance can improve.
4. Breadth First Search (BFS) is not the most optimal with any problem and hence is not an ideal choice. Better heuristic = better performance as number of actions grow.
5. A* Search (A*) with better heuristics appears most suited for real world if time is not a constraint.

ANALYSIS

Search Complexity - Number of Nodes Expanded vs. Number of Actions

As number of actions increase, the number of nodes expanded drastically increases. For uninformed searches, this trend seems to be very prevalent. For informed searches the number of nodes expanded is on a downward trend as the heuristics improve. GFS expands the least amount of nodes as number of actions increases - most computationally beneficial. UCS is the most expensive and generates the most amount of nodes compared to any other algorithm.

Search Time - Search Time vs Number of Actions

Looking purely at the search time (and not any other factor), it increases for all search algorithms as the number of actions increase. A big exception is GFS with unmet-goals heuristics, which is the best in terms of search time for all four problems tested. A* is the most expensive in terms of time as number of actions increase. A* with unmet-goals heuristic is still acceptable for very large number of actions.

Optimality - Length of Plans by each Algorithm

DFS is an anomaly and needs to be discarded for this problem. The remaining algorithms are fairly optimal with respect to finding the optimal plan. For uninformed searches (except DFS) plan lengths are optimal from small to large problems. For informed searches, as the size of the actions grow, better heuristics will provide better optimality in terms of the plan length. GFS is less optimal at finding plan length as number of actions grow.

QUESTIONS

Which algorithm or algorithms would be most appropriate for planning in a very restricted domain (i.e., one that has only a few actions) and needs to operate in real time?

Answer: GFS with max-level or level-sum heuristic seems to be an ideal and most optimal algorithm to use for this scenario.

Which algorithm or algorithms would be most appropriate for planning in very large domains (e.g., planning delivery routes for all UPS drivers in the U.S. on a given day)

Answer: GFS does not give optimal plans for problems with very large domains. Performance might increase by using either the max-level or set-level heuristic. If computing resources and time are a constraint, then I will go with GFS with better of max-level or set-level heuristic. If computing resources and time are not a constraint, then I will go with A* as a more optimal solution.

Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?

Answer: A* will be the most optimal given that time and computing resources are not a constraint.