Artificial Intelligence Nanodegree Alexandru Korotcov

Project: Implement a Planning Search April 15th, 2017

Experiments and analysis for non-heuristic and heuristic planning solution searches

This project is an implementation of different classes and functions needed to solve deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. With progression search algorithms like those in the navigation problem the optimal plans for each problem have been computed. Unlike the navigation problem, there is no simple distance heuristic to aid the agent. Instead, I have implemented domain-independent heuristics and analyze performance of different non-heuristic and heuristic planning solution searches.

Planning problems

All problems are in the Air Cargo domain. They have the same action schema defined, but different initial states and goals.

- Air Cargo Action Schema:

```
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))
```

- 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 and 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))

The optimal plan lengths to achieve the problem 1, 2, 3 goals are 6, 9, and 12 respectively. In our problem, for simplicity and clarity, the optimal search plan will be defined as a plan with optimal length, minimum elapsed time, and minimum number of expanded nodes.

Uninformed planning searches

Uninformed search algorithms use highly generic search strategies generating the search tree without using any domain specific knowledge. In general, those group of algorithms are only capable to generate successor states and evaluate costs and goal conditions. Classic examples of uninformed searches are breadth-first (explore states at the same search tree level first), depth-first (explores nodes as far as possible along each branch first), and uniform cost search (explores states with more or less cost from stating node to current state node) algorithms. These algorithms are good for some small problems and could be very expensive to solve complex problems.

The result of uninformed planning searches for all three problems are summarized in the Table 1. The optimal search plan for each problem is highlighted in the table.

Table 1. Uninformed planning search for all three problems.

Problem 1

Algorithm	expansions	goal tests	time elapsed, s	plan length	plan optimality
breadth_first_search	43	56	0.101	6	Yes
depth_first_graph_search	21	22	0.029	20	No
uniform_cost_search	55	57	0.114	6	Yes
greedy_best_first_graph_search h_1	7	9	0.014	6	Yes

Problem 2

Algorithm	expansions	goal tests	time elapsed, s	plan length	plan optimality
breadth_first_search	3343	4609	19.6	9	Yes
depth_first_graph_search	624	625	4.6	619	No
uniform_cost_search	4853	4855	57.3	9	Yes
greedy_best_first_graph_search h_1	998	1000	9.4	21	No

Problem 3

Algorithm	expansions	goal tests	time elapsed, s	plan length	plan optimality
breadth_first_search	14663	18098	145	12	Yes
depth_first_graph_search	408	409	2	408	No
uniform_cost_search	18223	18225	565	12	Yes
greedy_best_first_graph_search h_1	5578	5580	165	22	No

In the simple problem 1, all the searches, except depth first graph search, found optimal plan. With the increased complexity (problem 2 and 3), only the breadth first and uniform cost searches were able to find an optimal plan. Nevertheless, number of node expansions and elapsed time were minimal for the breadth first search, thus we may conclude, that for the given problem the breadth first search is the optimal uninformed search plan.

Domain-independent heuristics

A main disadvantage of uninformed search algorithms is that they don't use any problem-specific information to guide the search. The informed or heuristic search techniques are applying the same general search principles, but selecting the most promising states first to expand in the search tree by using the problem-specific information given by an evaluation function (heuristic). The evaluation function incorporates an estimate of the path cost from the state to a goal. A* search is an informed search algorithm which uses a heuristic to estimate the distance from a node to a goal state.

The result of A* planning search problems using different heuristics are summarized in the Table 2. The *h_1* heuristic is not the actual informed heuristic, it returns cost equal one all the time, thus behaves like the uniform cost search. The *h_ignore_preconditions* estimates the minimum number of actions that must be carried out from the current state in order to satisfy all of the goal conditions by ignoring the preconditions required for an action to be executed. The *h_pg_levelsum* estimates the sum of all actions that must be carried out from the current state in order to satisfy each individual goal condition. All the informed searches found an optimal plan. The optimal search plan for each problem is highlighted in the table.

Table 2. A* planning search using different heuristics for all three problems.

Problem 1

Algorithm	expansions	goal tests	time elapsed, s	plan length	plan optimality
astar_search h_1	55	57	0.102	6	Yes
astar_search h_ignore_preconditions	41	43	0.056	6	Yes
astar_search h_pg_levelsum	11	13	4.610	6	Yes

Problem 2

Algorithm	expansions	goal tests	time elapsed, s	plan length	plan optimality
astar_search h_1	4853	4855	58.8	9	Yes
astar_search h_ignore_preconditions	1506	1508	19.1	9	Yes
astar_search h_pg_levelsum	86	88	1001.4	9	Yes

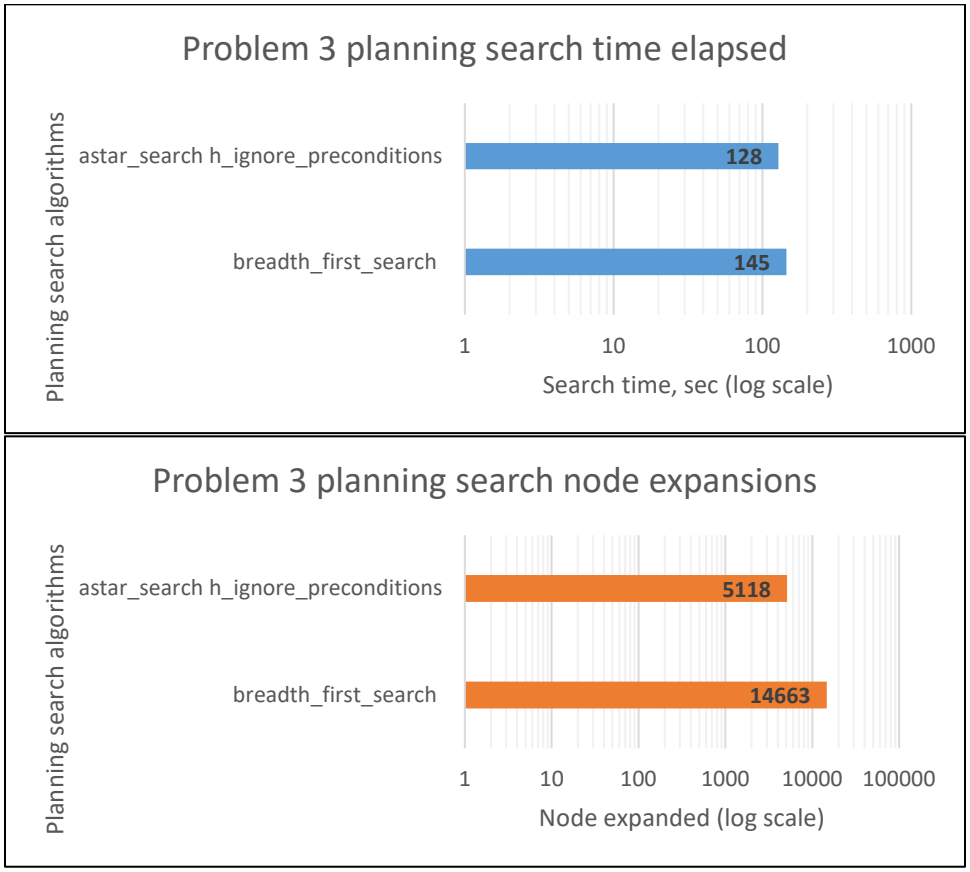
Problem 3

Algorithm	expansions	goal tests	time elapsed, s	plan length	plan optimality
astar_search h_1	18223	18225	594	12	Yes
astar_search h_ignore_preconditions	5118	5120	128	12	Yes
astar_search h_pg_levelsum	414	416	9099	12	Yes

All used heuristics are admissible, thus the A* guarantees the optimal search plan solution. The level sum heuristic is a very good in term of number of node expanded, therefore, the computing time of using this heuristic is too long. The ignore preconditions heuristic is the most optimal heuristic for the A* planning search for the given air cargo problems.

Summary:

The figures below show search time elapsed and number of node expanded during most complex problem 3 search planning for the most optimal informed and heuristic based search plans. The best optimal characteristics in planning search problems were achieve using A* search with ignore preconditions.



The ***ignore all preconditions*** is the most simple heuristic which is admissible and always guarantee the optimal solution. Moreover, because heuristic function guides selecting of the next state by estimating the path with minimum cost from the current state to a goal and expending the most optimal node, the heuristic search for complex problems outperforms any uninformed search, which is just expanding nodes and checking if the current state is goal state or not. In general, the problem-specific information knowledge implemented via heuristic function always lead search faster toward a goal state and choosing a good heuristic function is a very important for solving planning search problems.