Heuristic Function Analysis | Planning Search

Udacity Artificial Intelligence Nanodegree
Brittany Martin | 1/24/2018

Abstract

In this project, we are tasked to solve deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. With progression search algorithms, optimal plans for each problem are computed. There is no simple distance heuristic to aid the agent. Instead, we implemented domain-independent heuristics.

Action Schema

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

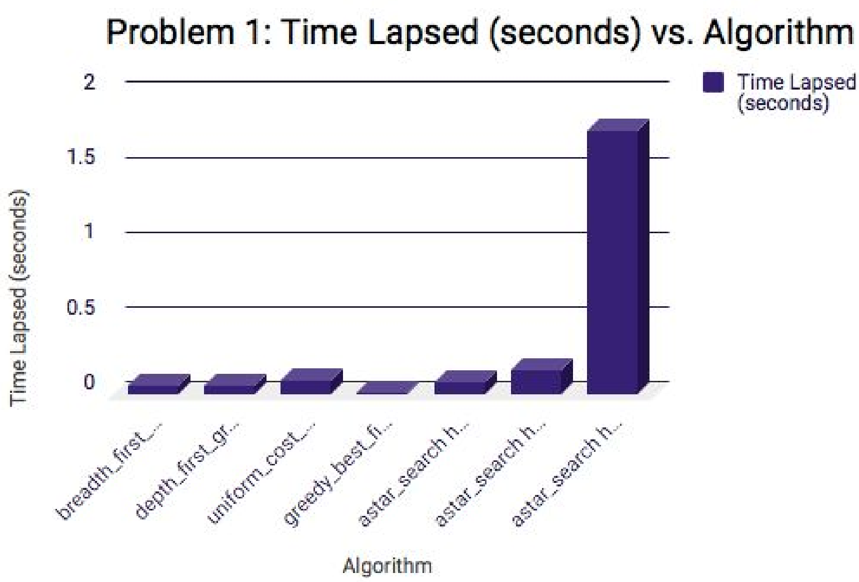
Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(C4, ORD)
∧ At(P1, SFO) ∧ At(P2, JFK)
∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3) ∧ Cargo(C4)
∧ Plane(P1) ∧ Plane(P2)
∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL) ∧ Airport(ORD))
Goal(At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO))

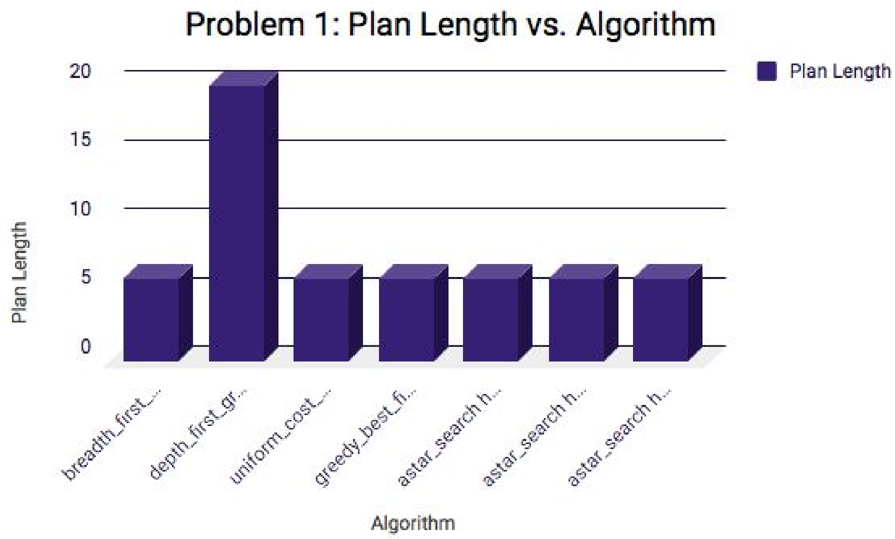
Results

Per the recommendation of Udacity, I utilized PyPy for running the searches.

Problem 1 Initial State and Goal

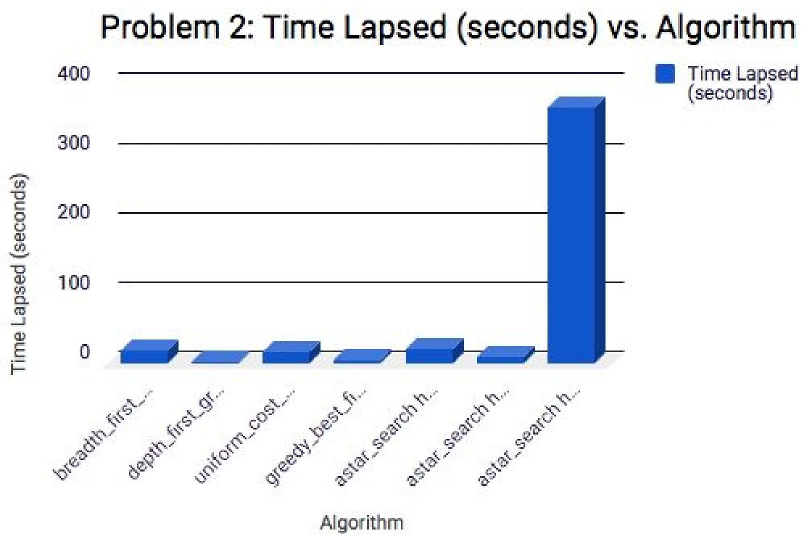
PROBLEM 1					
Algorithm	Expansions	Goal Tests	New Nodes	Time Lapsed (seconds)	Plan Length
breadth_first_search	43	56	180	0.05459801201	6
depth_first_graph_search	21	22	84	0.05979764694	20
uniform_cost_search	55	57	224	0.08602173103	6
greedy_best_first_graph_search h_1	7	9	28	0.009701812989	6
astar_search h_1	55	57	224	0.07332619798	6
astar_search h_ignore_preconditions	51	53	208	0.162204607	6
astar_search h_pg_levelsum	55	57	224	1.759035398	6

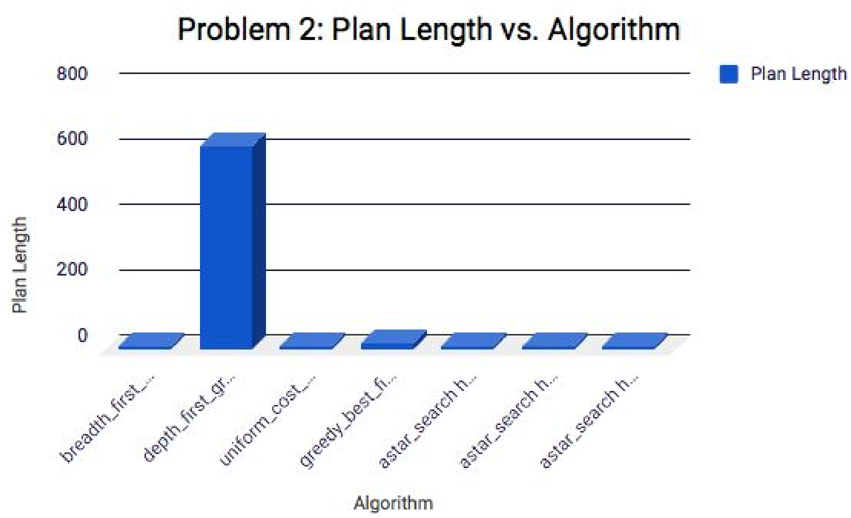




Problem 2 Initial State and Goal

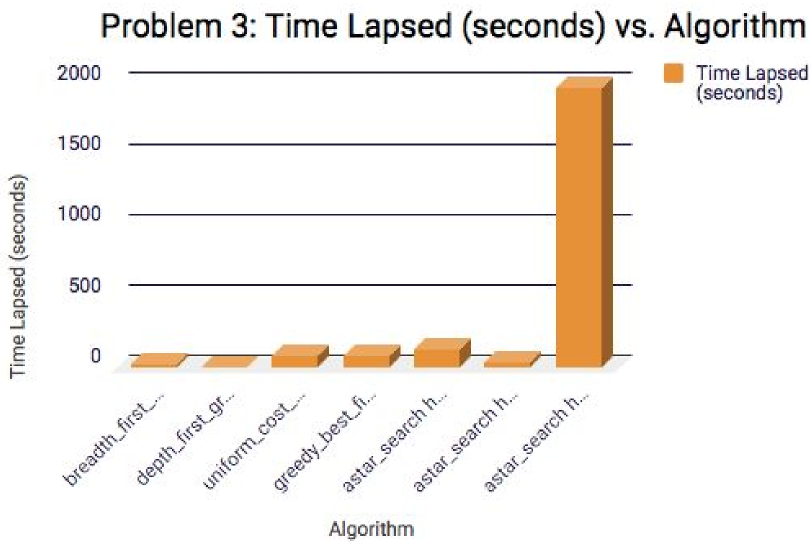
PROBLEM 2					
Algorithm	Expansions	Goal Tests	New Nodes	Time Lapsed (seconds)	Plan Length
breadth_first_search	3343	4609	30509	17.23337885	9
depth_first_graph_search	624	625	5602	1.410260129	619
uniform_cost_search	4852	4854	44030	15.87766461	9
greedy_best_first_graph_search h_1	990	992	8910	3.668398147	17
astar_search h_1	4852	4854	44030	21.41859974	9
astar_search h_ignore_preconditions	4297	4299	39110	8.666144078	9
astar_search h_pg_levelsum	4852	4854	44030	366.9104339	9

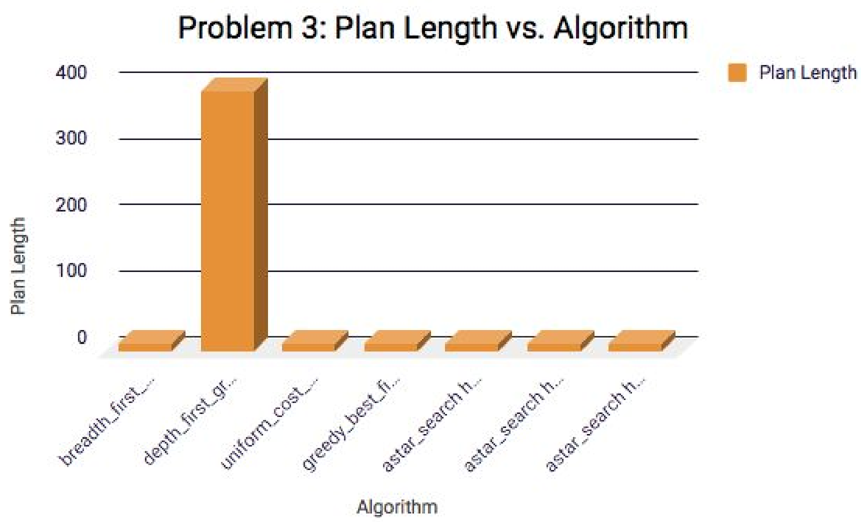




Problem 3 Initial State and Goal

PROBLEM 3					
Algorithm	Expansions	Goal Tests	New Nodes	Time Lapsed (seconds)	Plan Length
breadth_first_search	14663	18098	129631	19.49652567	12
depth_first_graph_search	408	409	3364	0.648579923	392
uniform_cost_search	18223	18225	159618	78.76316557	12
greedy_best_first_graph_search h_1	18223	18225	159618	78.70390829	12
astar_search h_1	16993	16995	149302	120.3517763	12
astar_search h_ignore_preconditions	16993	16995	149302	35.15960233	12
astar_search h_pg_levelsum	18223	18225	159168	1972.053674	12





Optimal Plans

Optimal Plan for Problem 1

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

Greedy Best First Graph Search was the clear winner for locating the goal state in Problem 1 by completing in the lowest amount of time and accurately locating the optimal plan. The algorithms were almost all equal in time to find the goal state except for A* search with the ‘h_pg_levelsum’ option (400%

longer). All algorithms except Depth First search located the minimum plan length (6) instead of DPL's return of 20.

Optimal Plan for 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)

Greedy Best First Graph Search is the winner again by finding the optimal plan in the lowest amount of time with the least amount of expansions. Since Problem 2 increases in complexity, DPL's outlier of plans (619) versus the optimal plan (12) becomes more exacerbated.

Optimal Plan for 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(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Finally, for the most complex problem, Problem 3, the recommended algorithm is **A* Search (Ignore Preconditions)** clause because it was the most performant, located the goal state and minimized the nodes explored.

Analysis

<u>Non-heuristic Planning Solution Searches</u>

Breadth First Search (BFS) could always find optimal solution with an average number of expansions. Since BFS does not keep track of explored nodes, the number of expansions were significantly higher than uninformed graph search.

Depth First Search (DFS) kept track of all explored nodes and ended up expanding a lower number of nodes for each problem. Since DPL expands the deepest nodes first without any heuristic, it ends up going down a non-optimal path (Problem 2 and 3 were significantly sub-optimal).

Uniform Cost Search (UCS) always found the optimal path. It expanded significantly more nodes than Depth First Search (and was often far slower). but since it keeps track of explored nodes, it was faster than Breadth First Search. In a choice between BFS, DFS and USC, USC will deliver a correct result in a reasonable amount of time.

<u>Automatic Heuristics</u>

A* Search (Ignore Preconditions) - In this search, all preconditions are dropped. This can lead to operations achieving multiple goals and can undo the effects of others. For the Cargo problems, this algorithm always found the optimal path to the goal state.

A* Search (h_pg_levelsum) - In this search, the level sum heuristic is used which follows the subgoal independence assumption and returned the sum of the level costs of the goals. For all of the problems, it took the most amount of time to complete.

Bibliography

Russell, S. and Norvig, P. Artificial Intelligence, A Modern Approach, Third Edition, Sec 10.3.1 Planning graphs for heuristic estimation, LEVEL SUM HEURISTIC

Russell, S. and Norvig, P. Artificial Intelligence, A Modern Approach, Third Edition, Sec 10.2.3 Heuristics for planning, IGNORE PRECONDITIONS HEURISTIC