

Q1: Provide an optimal plan for Problems 1, 2, and 3.

P1	P2	P3
Load(C1, P1, SFO) Fly(P1, SFO, JFK) UnLoad(C1, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) UnLoad(C2, P2, SFO)	Load(C1, P1, SFO) Fly(P1, SFO, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Load(C3, P3, ATL) Fly(P3, ATL, SFO) UnLoad(C3, P3, SFO) UnLoad(C1, P1, JFK) UnLoad(C2, P2, SFO)	Load(C1, P1, SFO) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) UnLoad(C3, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) UnLoad(C4, P2, SFO) UnLoad(C1, P1, JFK) UnLoad(C2, P2, SFO)

Q2: Compare and contrast non-heuristic search result metrics (optimality, time elapsed, number of node expansions) for Problems 1,2, and 3. Include breadth-first, depth-first, and at least one other uninformed non-heuristic search in your comparison; Your third choice of non-heuristic search may be skipped for Problem 3 if it takes longer than 10 minutes to run, but a note in this case should be included.

Non-heuristic search

	P1	P2	P3
depth-first	21 22 84 20 0.04s	624 625 5602 619 6.0s	408 409 3364 392 3.3s
breadth-first	43 56 180 6 0.1s	3343 4609 30509 9 28.7s	14663 18098 129631 12 163.6s
depth-limit	101 271 414 50 0.2s	222719 2053741 2054119 50 2065.0s	Takes too long to run

Data presented in the format of “expansions | goal tests | new nodes | plan length | time elapsed

Q3: Compare and contrast heuristic search result metrics using A* with the "ignore preconditions" and "level-sum" heuristics for Problems 1, 2, and 3.

Heuristic search

	P1	P2	P3
A* - h1	55 57 224 6 0.1s	4841 4843 43929 9 32.4s	18181 18183 159281 12 122.6s
A* - ignore preconditions	41 43 170 6 0.1s	1438 1440 13188 9 10.0s	4931 4933 43869 12 37.7s
A* - level-sum	11 13 50 6 0.9s	85 87 831 9 70.4s	250 252 2286 12 259.0s

Data presented in the format of "expansions | goal tests | new nodes | plan length | time elapsed"

Q4: What was the best heuristic used in these problems? Was it better than non-heuristic search planning methods for all problems? Why or why not?

Q: What was the best heuristic used in these problems?

A: First of all, we need to define what does "best" mean for a planning solution. Obviously a must-have criteria is that the plan has to take the minimal step possible to reach the goal, this means the fewer steps a solution takes the better it is. In this case, there are several solutions produce the optimal number of steps, so additional criterias are needed.

How to find the next best criteria in our metric space (expansions, goal tests, new nodes, time elapsed)? Answer is that it depends. If we assume computer resources are infinite, time elapsed is the best pick because time complexity is usually how people judge the performance of an algorithm. But in the old days, computer resources is very limited, so space complexity is sometimes a higher priority. In A* search, every node expansion needs to put all its children on the stack which takes memory, in a problem with high branching factor, it can be very big. So when we have limited memory that's less than enough to hold the entire graph, we should favor solutions that visit least amount of nodes while solving the problem.

In this particular case, computer memory are way sufficient to hold the entire graph for all three problems, so I pick time elapsed + number steps as the criteria to judge the solutions.

The optimal solution is more obvious once we visualize all search results onto a 2-d space (show below). The x-axis represents the number of steps to the plan and y-axis represents the

duration in second the algorithm takes to reach a result. The solution that is closest to (0,0) is the optimal. Here's the result:

P1: A* - ignore precondition

P2: A* - ignore precondition

P3: A* - ignore precondition

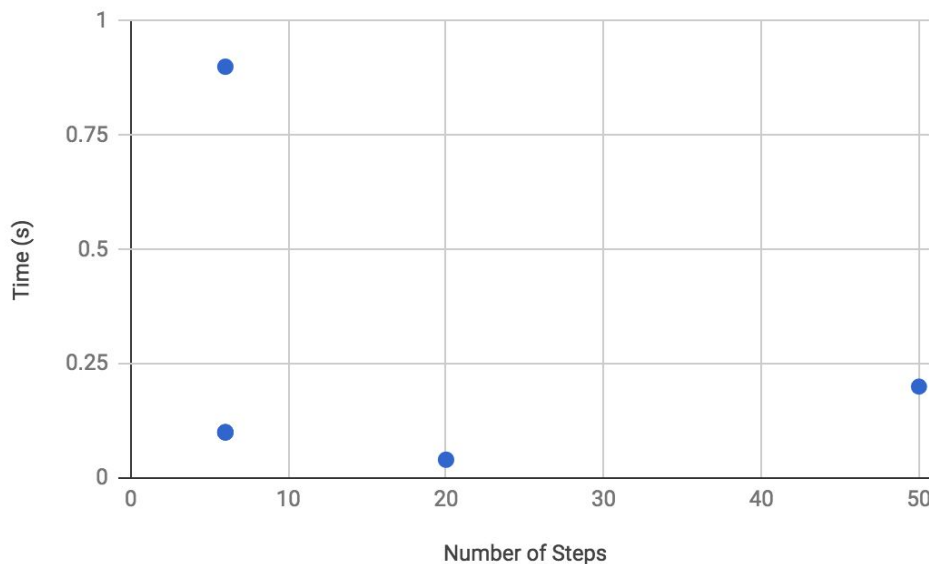
Q: Was it better than non-heuristic search planning methods for all problems? Why or why not?

A: In this case Yes, heuristic search outperforms non-heuristic for all 3 problems. However in general heuristic search is not ALWAYS better than non-heuristic search. According to AIND Lesson 8 - 32. A* Search 5, Heuristic search, A* search only finds the lowest cost path if $h(s) < \text{true cost}$. In other words, h should never overestimate distance to the goal. When h overestimates the cost to the goal, it performs worse than breadth-first search (non-heuristic) because breadth-first search is always complete (according to AIND Lesson 8, 25. Search Comparison 3)

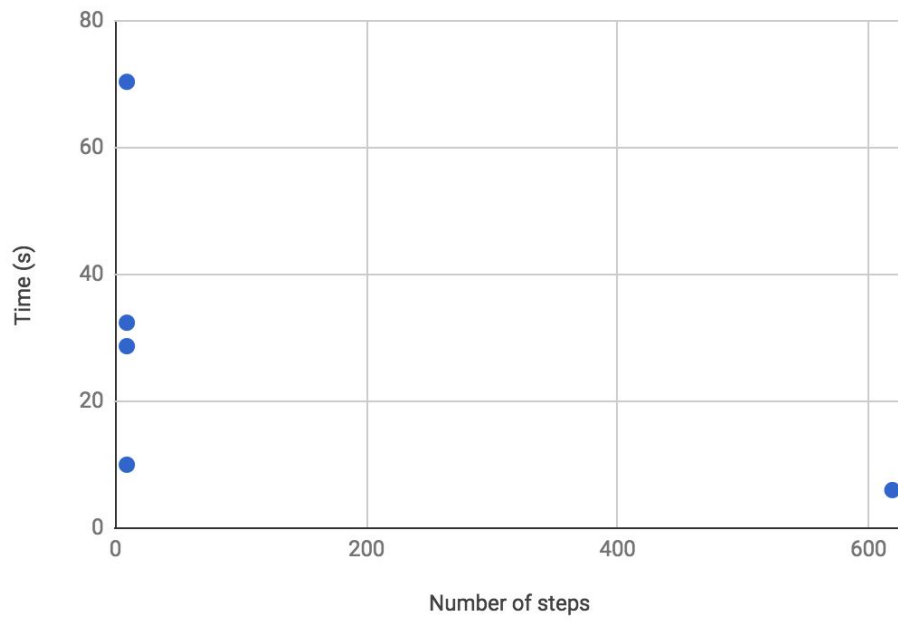
So back to the 3 problems we have in this report, because the h function ignores preconditions for each actions, and thus makes it admissible. It's able to perform better than non-heuristic search.

P1:

P1 analysis



P2:



P3:

