# Heuristic analysis

Xiaolong Yang

The target of heuristic analysis is to find the best algorithms for each problem, with the smallest path and lowest time. There are exhibits in the end show the performance of different algorithm on different problems.

We can easily find the **Breath First Search** worked best on Problem 1 and Problem 2, and **A\* search h ignore preconditions** worked best on Problem 3, here is the summary from the exhibits:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P | Best Search Type | Expansions | Goal Tests | New Nodes | Length | Time | Optimal Sequence of Actions |
| 1 | Breadth first search | 43 | 56 | 180 | 6 | 0.0304 | |  | | --- | | **Load(C1, P1, SFO)** | | **Load(C2, P2, JFK)** | | **Fly(P2, JFK, SFO)** | | **Unload(C2, P2, SFO)** | | **Fly(P1, SFO, JFK)** | | **Unload(C1, P1, JFK)** | |
| 2 | Breadth first search | 3343 | 4609 | 30509 | 9 | 12.8157 | |  | | --- | | **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)** | |
| 3 | A\* search h ignore preconditions | 5040 | 5042 | 44944 | 12 | 33.9542 | |  | | --- | | **Load(C2, P2, JFK)** | | **Fly(P2, JFK, ORD)** | | **Load(C4, P2, ORD)** | | **Fly(P2, ORD, SFO)** | | **Unload(C4, P2, SFO)** | | **Load(C1, P1, SFO)** | | **Fly(P1, SFO, ATL)** | | **Load(C3, P1, ATL)** | | **Fly(P1, ATL, JFK)** | | **Unload(C3, P1, JFK)** | | **Unload(C2, P2, SFO)** | | **Unload(C1, P1, JFK)** | |

We can easily find the “**Optimal sequence of actions**” for each problem in **rightmost** column of the above table.

There are two categories of Search Strategy: One is Strategy with no heuristic, such as breadth first search, uniform cost search, and depth first search; Another is Strategy with heuristic, such as A\* Search with h-ignore-preconditions or h-pg-levelsum. Non-heuristic strategy works well in simple problem, and heuristic strategy works well in complex problem.

Breath First Search[1]:

It always tried to find shortest path first, thus provides optimal plan. However, as problem got complex, the time for search would also increase.

Depth First Search:

It always tried to minimize the search time, but not consider the shortest path.

Uniform Cost Search and A\* Search with h1:

The goal is to find the cheap path, even when they find the goal. Both guarantee the optimal paths, but consume more time then Breath First Search.

A\* search with h-ignore-preconditions or h-pg-level-sum:

These are search strategy with heuristics. The heuristic of h-ignore-preconditions estimate the min number of actions need from current state to satisfy all the goal conditions, ignore the preconditions need for action executed. The heuristic of h-pg-level-sum sum the level costs of individual goal. Heuristic Search strategy works worse in simple problem such as 1 and 2, but works better in complex situation such as 3. However, the h-ignore-preconditions works better then h-pg-level-sum, since the latter is far more complex in heuristic and take too much time for calculations.

## Exhibits1-Problem1



Exhibits2-Problem2

Exhibits3-Problem3

## Reference

[1] Stuart J. Russell, Peter Norvig (2010), Artificial Intelligence: A Modern Approach (3rd Edition).