Heuristic Analysis of Planning Solution Searches

This paper contains an analysis of heuristic and non-heuristic planning solution searches by comparing and contrasting their search metrics for 3 sets of problems.

Problems

For this experiment, we search for a solution for the following 3 sets of problems:

Air Cargo Problem Possible Actions

```
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))
```

Air Cargo Problem 1

```
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))
```

Air Cargo Problem 2

```
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))
```

Air Cargo Problem 3

```
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))
```

Optimal Plans

For each of the problem above, the following are the optimal solutions (solutions that require the least number of actions to reach the goal state):

Optimal Solution to Air Cargo Problem 1

Since there are 1 cargo and 1 plane for each of the 2 airports, an optimal plan would load cargo onto the airplane once, fly each cargo to its destination once, and unload cargo from the airplane once, for a total of 3 actions for each cargo.

The optimal number of actions would be 2 cargos x (1 load + 1 unload + 1 flight) = 6 actions.

Example:

```
[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(C3, P3, ATL), Fly(P3, ATL, SFO),
Unload(C3, P3, SFO)]
```

Optimal Solution to Air Cargo Problem 2

Since there are 1 cargo and 1 plane for each of the 3 airports, an optimal plan would be the same as the first problem.

The optimal number of actions would be 3 cargos x (1 load + 1 unload + 1 flight) = 9 actions.

Example:

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

Optimal Solution to Air Cargo Problem 3

Since there are 1 cargo at each of the 4 airports but only 2 planes and a goal to deliver them to 2 airports, an optimal plan would be to load and unload each container once, while each airplane flying twice (once to pick up a load at a non-goal location, then once to drop off both loads at the goal location.)

The optimal number of actions would be 4 loads + 4 unloads + (2 flights x 2 airplanes) = 12 actions.

Example:

```
[Load(C1, P1, SFO) , Fly(P1, SFO, ATL), Load(C3, P1, ATL), Fly(P1, ATL, JFK),
Unload(C1, P1, JFK), Unload(C3, P1, JFK), Load(C2, P2, JFK) , Fly(P2, JFK, ORD),
Load(C4, P2, ORD), Fly(P2, ORD, SFO), Unload(C2, P2, SFO), Unload(C4, P2, SFO),]
```

Search Performance

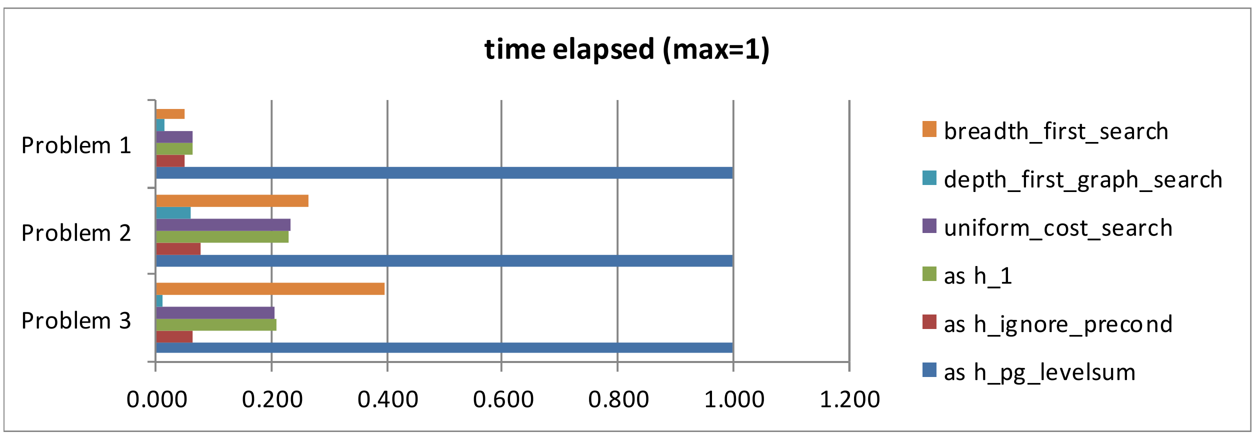
Non-Heuristic Search Performance

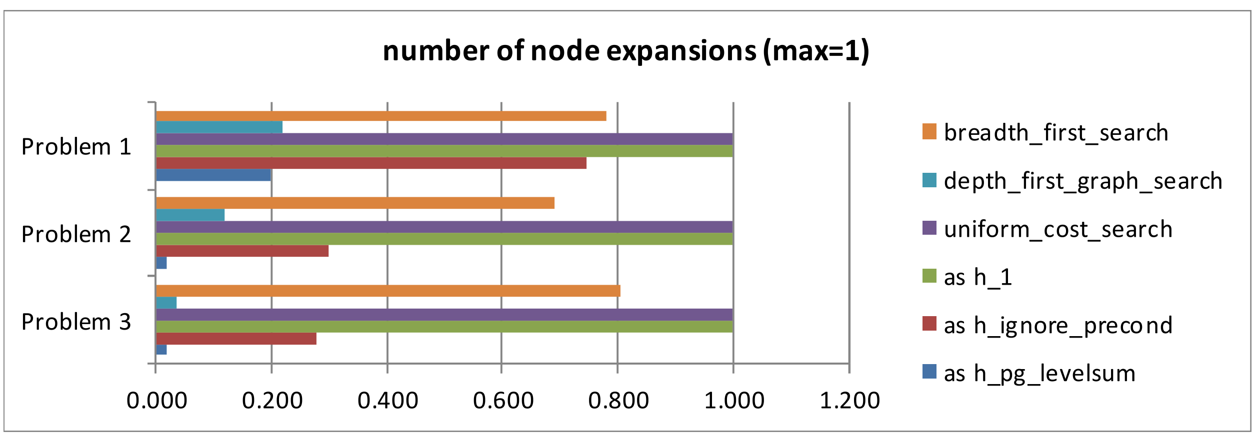
Metric	optimality (plan length)			time elapsed (sec)			number of node expansions		
Problem	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
breadth_first_search	6	9	12	0.029	12.144	87.838	43	3343	14663
depth_first_graph_search	12	575	596	0.008	2.695	2.803	12	582	627
uniform_cost_search	6	9	12	0.037	10.684	45.623	55	4835	18225

Heuristic Search Performance

Metric	optimality (plan length)			time elapsed (sec)			number of node expansions		
Problem	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
as h_1	6	9	12	0.036	10.517	46.419	55	4835	18225
as h_ignore_precond	6	9	12	0.028	3.465	14.070	41	1450	5040
as h_pg_levelsum	6	9	12	0.576	45.904	222.294	11	86	314

Side-by-Side Comparison (comparison of relative values when max=1)





Conclusion

Of the 3 non-heuristic search methods, depth first search was the fastest to complete the search with the least number of node expansions. However, it failed to find optimal solutions for all 3 problems. Breadth first search and uniform cost search have both found optimal solutions. Uniform cost search required less time to complete the search while expanding more nodes compared to the breadth first search.

Both heuristic search methods succeeded in finding an optimal solution. Heuristic to ignore preconditions required less time to complete the search while expanding more nodes compared to the heuristic that used level sum.

For all 3 problems and for all search methods that found optimal solutions, the search method that used heuristic to ignore preconditions found the optimal solution in the least amount of time.

Looking at the metrics, there are two ways to reduce the execution time: reduce the number of expansions required and reduce the time required to perform each expansion.

Comparing the results of the two heuristic search methods, we observe that the level sum heuristic expands far less nodes compared to the heuristic to ignore preconditions. However, the former still takes significantly longer to complete the search than the latter. Looking at the implementation of the two heuristics and how they are used, level sum heuristic constructs a new planning graph each time a node is expanded. Construction of the planning graph is a significantly expensive operation compared to ignoring preconditions. The experiment suggests that both the accuracy of the heuristic and its execution cost must be weighed when selecting a heuristic that allows the search to complete faster.