Planning Search Heuristic Analysis

by Teruyuki Takahashi

Planning Problems

I was given three planning problems in the Air Cargo domain that use the same **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))
```

The three problems have the following initial states and goals:

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))
```

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))
```

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))
```

The goals above can be reached using different plans, but the **optimal plan lengths** for problems 1,2, and 3 are **6, 9, and 12 actions**, respectively. Below are sample plans with optimal length:

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)
```

Problem 2:

```
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
Unload(C2, P2, SFO)
Unload(C1, P1, JF
```

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

Uninformed Search Strategies Analysis

Performance measures were collected using the following commands:

```
python run_search.py -p 1 -s 1 2 3 4 5 6 7 >> run_uninformed_search_results_p1.txt
python run_search.py -p 2 -s 1 3 5 7 >> run_uninformed_search_results_p2.txt
python run_search.py -p 3 -s 1 3 5 7 >> run_uninformed_search_results_p3.txt
```

For Problem 2, because their execution time exceeded 10 minutes, we cancelled data collection for Breadth First Tree Search, Depth Limited Search, and Recursive Best First Search (per Udacity staff instruction). For the same reason, with Problem 3 we did not collect any data for Breadth First Tree Search, Depth Limited Search, Uniform Cost Search, and Recursive Best First Search.

Problem 1 Results

Search Strategy	*Optimal*	*Path Length*	*Execution Time(s)*	*Node Expantions*
Breadth First Search	*Yes*	*6*	*0.024*	*43*
Breadth First Tree Search	*Yes*	*6*	*0.772*	*1458*
Depth First Graph Search	*No*	*12*	*0.006*	*12*
Depth Limited Search	*No*	*50*	*0.068*	*101*
Uniform Cost Search	*Yes*	*6*	*0.029*	*55*
Recursive Best First Search	*Yes*	*6*	*2.174*	*4229*
Greedy Best First Graph Search	*Yes*	*6*	*0.005*	*7*

Problem 2 Results

Search Strategy	*Optimal*	*Path Length*	*Execution Time(s)*	*Node Expantions*
Breadth First Search	*Yes*	*9*	*6.863*	*3401*
Breadth First Tree Search	----	----	----	----
Depth First Graph Search	*No*	*346*	*1.309*	*350*
Depth Limited Search	----	----	----	----
Uniform Cost Search	*Yes*	*9*	*9.211*	*4761*
Recursive Best First Search	----	----	----	----
Greedy Best First Graph Search	*Yes*	*9*	*1.064*	*550*

Problem 3 Results

Search Strategy	*Optimal*	*Path Length*	*Execution Time(s)*	*Node Expantions*
Breadth First Search	*Yes*	*12*	*35.743*	*14491*
Breadth First Tree Search	*----*	*----*	*----*	*----*
Depth First Graph Search	*No*	*1878*	*17.985*	*1948*
Depth Limited Search	*----*	*----*	*----*	*----*
Uniform Cost Search	*Yes*	*12*	*42.705*	*17783*
Recursive Best First Search	*----*	*----*	*----*	*----*
Greedy Best First Graph Search	*No*	*22*	*9.861*	*4031*

Analysis
I didn't execute 2,4,6 at problem2,3. Because need much time.
And in my opinion Breadth First Search is better way. Because can take optimal answer at all problem, and need less time and less node.

Informed (Heuristic) Search Strategies Analysis

Performance measures were collected using the following commands:

```
python run_search.py -p 1 -s 8 9 10 >> run_informed_search_results_p1.txt
python run_search.py -p 2 -s 8 9 10 >> run_informed_search_results_p2.txt
python run_search.py -p 3 -s 8 9 >> run_informed_search_results_p3.txt
```

Problem 1 Results

Search Strategy	*Optimal*	*Path Length*	*Execution Time(s)*	*Node Expantions*
*A*Search with h1 heuristic*	*Yes*	*6*	*0.030*	*55*
*A*Search with Ignore PreConditions Heuristic*	*Yes*	*6*	*0.028*	*41*
*A*Search with Level Sum Heuristic*	*Yes*	*6*	*0.786*	*11*

Problem 2 Results

Search Strategy	*Optimal*	*Path Length*	*Execution Time(s)*	*Node Expantions*
*A*Search with h1 heuristic*	*Yes*	*9*	*9.169*	*4761*
*A*Search with Ignore PreConditions Heuristic*	*Yes*	*9*	*3.347*	*1506*
*A*Search with Level Sum Heuristic*	*Yes*	*9*	*142.696*	*86*

Problem 3 Results

Search Strategy	*Optimal*	*Path Length*	*Execution Time(s)*	*Node Expantions*
*A*Search with h1 heuristic*	*Yes*	*12*	*40.072*	*17783*
*A*Search with Ignore PreConditions Heuristic*	*Yes*	*12*	*13.493*	*5081*
*A*Search with Level Sum Heuristic*	*----*	*----*	*----*	*----*

Analysis
I didn't execute 10 at problem3 because need much time.
8,9,10 can get optimal answer and don't use much node, but need long time.