HEURISTIC ANALYSIS

Implement a Planning Search, AI nanodegree, Udacity

Developed by Maha Ezzat

Implement a Planning Search

We are required to plan a set of actions to reach the goal state starting from the initial state.

The possible actions are:

- 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))

We have three problems to be solved:

- 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)

Optimal Solution

Using the following search strategies:

- Breadth First Search
- Depth Limited Search
- Depth First Graph Search
- Uniform Cost Search
- A* h1 Heuristic Function
- A* h_Ignore Precondition Function
- A* h_Levelsum Function

We find the optimal solution for each problem is:

Problem 1

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

Problem 2

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

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

Searching Results

Problem 1

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
Breadth First Search	**Yes**	**6**	**0.05 sec**	**44**	**57**	**184**
Depth Limited Search	No	50	0.14 sec	96	248	291
Depth First Graph Search	No	20	0.03 sec	21	22	84
Uniform Cost Search	Yes	6	0.06 sec	55	57	224

Problem 2

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
Breadth First Search	**Yes**	**9**	**11.6 sec**	**3346**	**4612**	**30534**
Depth Limited Search	No	50	24 min	213491	1967093	1967471
Depth First Graph Search	No	1085	11.3 sec	1124	1125	10017
Uniform Cost Search	Yes	9	17.2 sec	4778	4780	43379

Note: Depth Limited Search takes more than 10 min to find the result in the second problem.

Problem 3

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
Breadth First Search	**Yes**	**12**	**56.5 sec**	**14120**	**17673**	**124926**
Depth Limited Search	-	-	-	-	-	-
Depth First Graph Search	No	660	5.3 sec	677	678	5608
Uniform Cost Search	Yes	12	72.2 sec	17792	17794	156001

Note: Depth Limited Search takes more than 10 min to find the result in the third problem.

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes[a]	Yes[a,b]	No	No	Yes[a]	Yes[a,d]
Time	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(b\ell)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Yes[c]	Yes	No	No	Yes[c]	Yes[c,d]

Figure 3.21 Evaluation of tree-search strategies. b is the branching factor; d is the depth of the shallowest solution; m is the maximum depth of the search tree; l is the depth limit. Superscript caveats are as follows: [a] complete if b is finite; [b] complete if step costs $\geq \epsilon$ for positive ϵ; [c] optimal if step costs are all identical; [d] if both directions use breadth-first search.

As illustrated in the AIMA book [1] :

- Depth First Graph search is the faster search and requires the least memory but unfortunately it does not give an optimal solution.

- Depth Limited Search is the worst search with respect to the speed and memory.

- The best search is the Breadth First Search according to speed and memory.

Best Heuristic Function

Problem 1

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
A* h1 Heuristic Function	Yes	6	0.06 sec	55	57	224
A* h_Ignore Precondition Function	Yes	6	0.06 sec	34	36	142
A* h_Levelsum Function	Yes	6	0.9 sec	9	11	43

Problem 2

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
A* h1 Heuristic Function	Yes	9	16.4 sec	4778	4780	43379
A* h_Ignore Precondition Function	Yes	9	5.9 sec	1369	1371	12539
A* h_Levelsum Function	Yes	9	64.4 sec	78	80	763

Problem 3

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
A* h1 Heuristic Function	Yes	12	75.5 sec	17792	17794	156001
A* h_Ignore Precondition Function	Yes	12	21.9 sec	4555	4557	40238
A* h_Levelsum Function	Yes	12	295.8 sec	253	255	2309

- A* h_Ignore Precondition Function has the minimum elapsed time of execution since it is required no-time to be executed, but it expands the highest number of nodes.

- A* h_Levelsum Function requires minimum memory since it expands the least number of nodes, however it requires the maximum execution time.

- A* h_Ignore Precondition is the best heuristic function since it’s significantly has the minimum execution time.

Best Search Strategy

Problem 1

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
Breadth First Search	Yes	6	0.05 sec	44	57	184
A* h_Ignore Precondition Function	**Yes**	**6**	**0.06 sec**	**34**	**36**	**142**

Problem 2

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
Breadth First Search	Yes	9	11.6 sec	3346	4612	30534
A* h_Ignore Precondition Function	**Yes**	**9**	**5.9 sec**	**1369**	**1371**	**12539**

Problem 3

Search Strategy	Optimal	Path length	Time elapsed	Expansions	Goal Tests	New Nodes
Breadth First Search	Yes	12	56.5 sec	14120	17673	124926
A* h_Ignore Precondition Function	**Yes**	**12**	**21.9 sec**	**4555**	**4557**	**40238**

- The best searching strategy is A* h_ignore prediction Function, because it has the least elapsed time and requires the least memory.

References

[1] Russell, S., & Norvig, P. (2010). Artificial intelligence (3rd ed., pp. 91). New Jersey: PEARSON.