Author: Rafal Biegacz

Report on Planning Search Implementation & Achieved Results

Introduction

The topic of the project is to implement air cargo transportation planning system and then check how it works based on the results achieved for three problems (presented below). Problems are defined using simplified version of PDDL (Planning Domain Definition Language) and need to be resolved using three actions: *Load*, *Unload* and *Fly*.

The project implements 10 different search strategies belonging to two categories as it was defined in Chapter 3 of "Artificial Intelligence A Modern Approach" book [1].

- Category I – uninformed search
  - Breadth First Graph
  - Breadth First Tree
  - Depth First graph
  - Depth Limited
  - Uniform Cost
- Category I – informed search
  - Recursive Best First
  - Greedy Best First Graph
  - A*
  - A* with "ignore preconditions" heuristic
  - A* with "level sum" heuristic

You can read more about informed search strategies in Chapter 3.5 of "Artificial Intelligence A Modern Approach" book [1] and about uninformed search strategies in Chapter 3.4 of [1].

In addition to that, as it was written in Chapter 3.7 of [1] the difference between general Three Search algorithms (e.g. Breadth First Graph) and Graph Search algorithms (e.g. Breadth First Tree) is that the former consider all possible solutions while the latter avoid redundant paths.

Problem 1

This problem deals with transporting C1 cargo from SFO airport to JFK airport and C2 cargo from JFK airport to SFO airport, using two airplanes P1 and P2. Definition of this problem expressed in PDDL looks as follows:

```
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))
```

The obvious solution to the above defined problem is the following series of actions:

- 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 plan length for this problem is 6 and we expect that our planning search agent is able to find such a path.

Solutions for Problem 1

Below are the results.

Search Strategy	Path Length	Optimal	Execution Time [s]	Expansions	Goal Tests	New Nodes
Breadth First	6	YES	0.028	43	56	180
Breadth First Tree	6	YES	0.856	1458	1459	5960
Depth First Graph	20	NO	0.013	21	22	84
Depth Limited	50	NO	0.079	101	271	414
Uniform Cost	6	YES	0.035	55	57	224
Recursive Best First	6	YES	2.659	4229	4230	17023
Greedy Best First Graph	6	YES	0.004	7	9	28
A*	6	YES	0.035	55	57	224
A* h_ignore_preconditions	6	YES	0.033	41	43	224
A* h_pg_levelsum	6	YES	1.253	55	57	224

Optimal strategies from the ‘path length’ perspective:

- Breadth first, Breadth First Tree, Uniform Cost and all informed search strategies

The best strategy which calculated ‘path length’ optimally & its calculations took the smallest amount of time is:

- Uninformed strategies: Breadth First
- Informed strategies: Greedy Best First Graph

The best strategy that calculated ‘path length’ optimally & generated smallest expansions (i.e. memory footprint) is:

- Uninformed strategies: Breadth First
- Informed strategies: Greedy Best First Graph

Problem 2

This problem deals with transporting:

- C1 cargo from SFO aiport to JFK airport
- C2 cargo from JFK airport to SFO airport
- C3 cargo from ATL airport to SFO

using three airplanes: P1, P2 and P3. Definition of this task expressed in PDDL looks as follows

```
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))
```

The obvious solution to the above defined problem is the following series of actions:

- 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, P2, SFO)

Optimal plan length for this problem is 9 and we expect that our planning search agent is able to find such a path.

Solutions for Problem 2

Below are the results. For some strategies search calculations took longer than 10 min that's why they were omitted.

Search Strategy	Path Length	Optimal	Execution Time [s]	Expansions	Goal Tests	New Nodes
Breadth First	9	YES	11.0260	3401	4672	31049
Breadth First Tree	---	---	---	---	---	---
Depth First Graph	1138	NO	6.508	1192	1193	10606
Depth Limited	---	---	---	---	---	---
Uniform Cost	9	YES	9.503	4779	4781	43370
Recursive Best First	---	---	---	---	---	---
Greedy Best First Graph	13	NO	1.177	590	592	5310
A*	9	YES	9.596	4779	4781	43370
A* h_ignore_preconditions	9	YES	3.378	1450	1452	13303
A* h_pg_levelsum	9	YES	1375.285	4779	4781	43370

Optimal strategies from the 'path length' perspective:

- Breadth first, Uniform Cost and A* - based strategies

The best strategy which calculated 'path length' optimally & its calculations took the smallest amount of time is:

- Uninformed strategies: Uniform Cost
- Informed strategies: A* with "ignore preconditions" heuristics

The best strategy that calculated 'path length' optimally & generated smallest expansions (i.e. memory footprint) is:

- Uninformed strategies: Breadth First
- Informed strategies: A* with "ignore preconditions" heuristics

Problem 3

This problem deals with transporting:

- C1 cargo from SFO aiport to JFK airport
- C2 cargo from JFK airport to SFO airport
- C3 cargo from ATL airport to JFK
- C4 cargo from ORD airport to SFO

using two airplanes: P1 and P2. Definition of this task expressed in PDDL looks as follows

```
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 obvious solution to the above defined problem is the following series of actions:

- 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(C3, P1, JFK)
- Unload(C1, P1, JFK)
- Unload(C4, P2, SF)
- Unload(C2, P2, SF)

Optimal plan length for this problem is 12 and we expect that our planning search agent is able to find such a path.

Solutions for Problem 3

Below are the results. For some strategies search operation took longer than 10 min that's why they were omitted.

Search Strategy	Path Length	Optimal	Execution Time [s]	Expansions	Goal Tests	New Nodes
Breadth First	12	YES	77.556	14491	17947	128184
Breadth First Tree	---	---	---	---	---	---
Depth First Graph	2014	NO	16.500	2099	2100	17558
Depth Limited	---	---	---	---	---	---
Uniform Cost	12	YES	41.532	18223	18225	159618
Recursive Best First	---	---	---	---	---	---
Greedy Best First Graph	22	NO	12.723	5578	5580	49150
A*	12	YES	41.712	18223	18225	159618
A* h_ignore_preconditions	12	YES	13.077	5040	5042	44944
A* h_pg_levelsum	12	YES	8682.550	18223	18225	159618

Optimal strategies from the 'path length' perspective:

- Breadth first, Uniform Cost and A* - based strategies

The best strategy which calculated 'path length' optimally & its calculations took the smallest amount of time is:

- Uninformed strategies: Uniform Cost
- Informed strategies: A* with "ignore preconditions" heuristics

The best strategy that calculated 'path length' optimally & generated smallest expansions (i.e. memory footprint) is:

- Uninformed strategies: Breadth First
- Informed strategies: A* with "ignore preconditions" heuristics

Conclusions

Taking into account the solutions for Problem 1, 2 and 3 it seems once can draw the following conclusions:

1. Informed search strategies are generally better than uninformed search strategies.
2. The best informed search strategy was A* search strategy with "ignore precondition" heuristics – both from the execution standpoint as well as from the memory footprint standpoint.
3. The best uninformed search strategy from the execution standpoint was Uniform Cost search strategy. Uniform cost strategy was only slightly worse than Breadth First search strategy taking into account memory footprint (both strategies are comparable from this standpoint).

   Chapter 3.4.7 in [1] summarizes properties of uninformed search strategies in the following way.

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.

(the table above is a quote from [1]).

The results achieved in the experiments agree with properties of Breadth First, Uniform Cost and Depth First search algorithms. Especially, that Breadth First and Uniform Cost are the optimal algorithms.

References

[1] Artificial Intelligence A Modern Approach, 3rd Edition by Stuart Russell and Peter Norvig