Planning Search Written Analysis

Manasi Kulkarni

This project was implemented to solve deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. We used planning garph and automatic domain-independent heuristics with A* search and their results were compared against different non heuristic search methods like braedth-first, depth-first, and uniform cost serach.

We were given three problems in the Air Cargo domain. They have the same action schema defined, but different initial states and goals.

Air Cargo 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	Problem 2	Problem 3
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))	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))	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 problems 1, 2, and 3 can be given as follows and have plan lengths 6, 9 and 12 respectively.

Problem 1 optimal plan	Problem 2 optimal plan	Problem 3 optimal plan
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) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK)	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)

Results and Comparison for Uninformed Searches :

The following tables provide brief comparison of uninformed non-heuristic searches. Please note that solutions using Breadth First Tree Search and Depth Limited Search are not going to be used for the overall comparison, given that they took longer than 10 minutes to run in problem 3. (Advised in the project instructions) This means, only Breadth First Search (BFS), Depth First Graph Search (DFS) and Uniform Cost Search (UCS) will be used for analysis. Also, for execution time and node expansions, smaller numbers are considered better.

Problem1

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Breadth First Search	**43**	**56**	**0.045**	**6**	**Yes**
Breadth First Tree Search	1458	1459	1.373	6	Yes
Depth First Graph Search	12	13	0.011	12	No
Depth Limited Search	101	271	0.131	50	No
Uniform Cost Search	55	57	0.0581	6	Yes

Problem 2

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Breadth First Search	**3401**	**4672**	**20.57**	**9**	**Yes**
Breadth First Tree Search	----------	----------	----------	----------	----------
Depth First Graph Search	350	351	2.23	346	No
Depth Limited Search	254020	2344879	1579.25	50	No
Uniform Cost Search	4761	4763	27.088	9	Yes

Problem 3

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Breadth First Search	**14491**	**17947**	**153.7**	**12**	**Yes**
Breadth First Tree Search	----------	----------	----------	----------	----------
Depth First Graph Search	3491	3492	93.45	3335	No
Depth Limited Search	----------	----------	----------	----------	----------
Uniform Cost Search	17783	17785	173.86	12	Yes

Analysis

Uninformed Search Strategies

Search strategies that are considered uninformed search also known as blind search have no additional information about states beyond that provided in the problem definition. They generate successors and distinguish a goal state from a non-goal state. Here, we are comparing the performance of strategies based on speed (time elapsed for execution, measured in seconds), **memory usage** (measured in search node expansions) and **optimality** (depending upon optimal length).
Note: Breadth First Tree Search and Depth Limited Search are omitted from this analysis as they took longer than 10 minutes to run in problem 2 and 3. (Advised in the project instructions)

According to the results obtained for the given problems, **Breadth First Search** and **Uniform Cost Search** are the only two uninformed search strategies that **yield an optimal action plan** under the 10min time limit. **Depth First Graph Search** is the **fastest** amongst all with the **low memory usage**. The number of node expansions for Depth First Graph Search was also lower than the numbers got for the other two searches. However, it failed to generate optimal action plan (plan length of 12 instead of 6 for problem 1, plan length of 346 instead of 9 for problem 2, plan length of 1878 instead of 12 for problem 3). Knowing this, DFS could be used as an algorithm for fast execution, to check if a route exists between two nodes, though it wouldn't return the shortest one.

Hence, if the plan length is critical criterion, both Breadth First Search and Uniform Cost Search are optimal and have same plan length in all 3 problems. However, Breadth First Search uses less memory and performs faster than Uniform Cost Search, it is the recommended search algorithm for all 3 problems.

Results and Comparison for Informed (Heuristic) Searches:

Problem1

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Recursive Best First Search	4229	4230	4.03	6	Yes
Greedy Best First Graph Search	7	9	0.007	6	Yes
A*Search with h1 heuristic	55	57	0.053	6	Yes
A* search with Ignore Preconditions heuristic	**41**	**43**	**0.068**	**6**	**Yes**
A* search with Level-Sum heuristic	11	13	3.024	6	Yes

Problem 2

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Recursive Best First Search	------	---------	----------	----------	----------
Greedy Best First Graph Search	550	552	4.62	9	Yes
A*Search with h1 heuristic	4761	4763	17.33	9	Yes
A* search with Ignore Preconditions heuristic	**1450**	**1452**	**8.65**	**9**	**Yes**
A* search with Level-Sum heuristic	86	88	614.77	9	Yes

Problem 3

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Recursive Best First Search	---------	---------	---------	----------	----------
Greedy Best First Graph Search	4031	4033	45.46	22	No
A*Search with h1 heuristic	17783	17785	72.81	12	Yes
A* search with Ignore Preconditions heuristic	**5003**	**5005**	**29.71**	**12**	**Yes**
A* search with Level-Sum heuristic	311	313	4225.43	12	Yes

Analysis

Informed (Heuristic) Search Strategies

Informed search strategies use problem specific knowledge beyond the definition of problem itself and can find solutions more efficiently than uninformed strategies. Here, we are comparing the performance of the informed searches in terms **of speed, memory usage** and **optimality**. We are also going to compare and contrast search result metrics using A* with the 'ignore-preconditions' and 'level-sum' heuristics for all 3 problems.

Note: We did not include the results for Recursive Best First Search and A* search with Level-Sum heuristic as they took longer than 10 minutes to run in problem 2 and 3. (Advised in the project instructions)

Greedy Best First Search is fast and uses less memory. If having an optimal plan length is not the primary criterion it would be the best alternative in terms of execution time and memory usage. For **A* Search using three different heuristics,** only A* search with h1 heuristic and A* search with ignore precondition heuristic were able to execute within 10 mins execution time for problem 3. Though A* search with level-sum heuristic is memory efficient is very slow as compare to others**. A* with ignore preconditions heuristic** is the **fastest** and **optimal** amongst all.

Comparison of Informed vs Uninformed Searches:

The uninformed search strategies that generate optimal plan lengths are Breadth First Search and Uniform Cost Search. We chose Breadth First Strategy because it was more efficient than Uniform Cost Search in terms of execution time and memory usage.
As for informed strategies A* Search with all three heuristics are optimal A* Search with Ignore Preconditions heuristic is the fastest and uses the least memory. So it comes to the choice between Breadth First Search and A* Search with Ignore Preconditions heuristic. We compare their results against our 3-problem set as shown below:

Problem 1

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Breadth First Search	43	56	0.045	6	Yes
A* with Ignore Preconditions Heuristic	41	43	0.068	6	Yes

Problem 2

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Breadth First Search	3401	4672	20.57	9	Yes
A* with Ignore Preconditions Heuristic	1450	1452	8.65	9	Yes

Problem 3

Search	Node Expansions	Goal Tests	Time Elapsed (in seconds)	Plan Length	Optimal
Breadth First Search	14491	17947	153.7	12	Yes
A* with Ignore Preconditions Heuristic	5003	5005	29.71	12	Yes

Conclusion:

From above comparison, it is fair to say that **A* Search with Ignore Preconditions heuristic** would be the best choice overall for our Air Cargo problem as it is faster and uses less memory. The results above show the benefits of using informed search strategies with custom heuristics over uninformed search techniques when searching for an optimal plan. The benefits are substantial both in terms of execution speed and memory usage.