Planning Search Heuristic Analysis

A. Uninformed Search Strategies Analysis

a. Optimal sequence of actions

1) Problem 1

1	Load(C1, P1, SFO)
2	Load(C2, P2, JFK)
3	Fly(P1, SFO, JFK)
4	Fly(P2, JFK, SFO)
5	Unload(C1, P1, JFK)
6	Unload(C2, P2, SFO)

2) Problem 2

1	Load(C3, P3, ATL)
2	Fly(P3, ATL, SFO)
3	Unload(C3, P3, SFO)
4	Load(C2, P2, JFK)
5	Fly(P2, JFK, SFO)
6	Unload(C2, P2, SFO)
7	Load(C1, P1, SFO)
8	Fly(P1, SFO, JFK)
9	Unload(C1, P1, JFK)

3) Problem 3

1	Load(C2, P2, JFK)
2	Fly(P2, JFK, ORD)
3	Load(C4, P2, ORD)
4	Fly(P2, ORD, SFO)
5	Unload(C4, P2, SFO)
6	Load(C1, P1, SFO)
7	Fly(P1, SFO, ATL)
8	Load(C3, P1, ATL)
9	Fly(P1, ATL, JFK)
10	Unload(C3, P1, JFK)
11	Unload(C2, P2, SFO)
12	Unload(C1, P1, JFK)

b. Results

1) Problem 1

Search Strategy	Path Length	Execution Time(s)	Node Expansion
Breadth First Search	**6**	0.027	43
Breadth First Tree Search	**6**	0.762	1458
Depth Firth Graph Search	12	0.006	12
Depth Limited Search	50	0.073	101
Uniform Cost Search	**6**	0.031	55
Recursive Best First Search	**6**	2.245	4229
Greedy Best First Graph Search	**6**	**0.004**	**7**

2) Problem 2

Search Strategy	Path Length	Execution Time(s)	Node Expansion
Breadth First Search	**9**	12.726	3401
Breadth First Tree Search*	---	---	---
Depth Firth Graph Search	346	1.352	**350**
Depth Limited Search*	---	---	---
Uniform Cost Search	**9**	9.813	4761
Recursive Best First Search*	---	---	---
Greedy Best First Graph Search	**9**	**1.107**	550

**The search strategy cannot be completed within 10 minutes*

3) Problem 3

Search Strategy	Path Length	Execution Time(s)	Node Expansion
Breadth First Search	**12**	93.624	14491
Breadth First Tree Search*	---	---	---
Depth Firth Graph Search	1878	17.549	**1878**
Depth Limited Search*	---	---	---
Uniform Cost Search	**12**	41.519	17783
Recursive Best First Search*	---	---	---
Greedy Best First Graph Search	22	**9.443**	4031

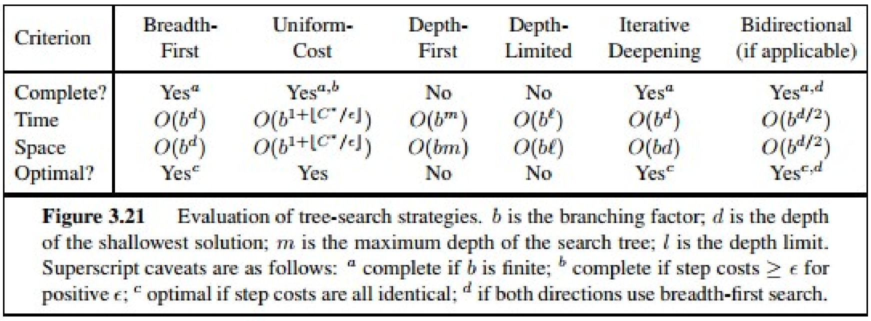
**The search strategy cannot be completed within 10 minutes*

c. Analysis

For Problem 1, 2 and 3, the optimal path length should be 6,9 and 12 respectively. Only **Breadth First Search** and **Uniform Cost Search** can achieve optimal results with less than 10 min time constraint. When considering execution speed and node expansion, **Depth First Graph Search** and **Greedy Best First Graph Search** usually expand fewer nodes within a reasonable time range. Fewer nodes imply less memory consumption in general. However, they do not guarantee to generate an optimal action plan.

Breadth First Search is the recommended search strategy if find optimal path length is the highest priority in all three problems. Comparing to **Uniform Cost Search**, it generates fewer nodes and have similar executive time level.

The observation is aligned with the statement that Breadth First Search is optimal and complete. The only downside is the over-expansion of nodes, which consumes a lot of memory if the problem's branching factor is high. Below is the comparison of uninformed search strategy from textbook (section 3.4.7).



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.

Apart from the two search strategies above, the Depth First Graph Search plan lengths are so much longer than the optimal path length that it wouldn't make sense to use this search strategy, especially for complex problems like 2 and 3. **Greedy Best First Graph Search** is worthwhile to try given its impressive searching time on all three problems. In problems 1 and 2, it manages to find the optimal path. In problem 3, it does not find the optimal path but the path length it generates is 22.

B. Heuristic Search Strategies Analysis

a. Results

1) Problem 1

Search Strategy	Path Length	Execution Time(s)	Node Expansion
A* Search with h1 heuristic	**6**	0.034	55
A* Search with Ignore Preconditions heuristic	**6**	**0.033**	41
A* Search with Level Sum heuristic	**6**	0.883	**11**

2) Problem 2

Search Strategy	Path Length	Execution Time(s)	Node Expansion
A* Search with h1 heuristic	**9**	10.455	4761
A* Search with Ignore Preconditions heuristic	**9**	**4.168**	1450
A* Search with Level Sum heuristic	**9**	162.869	**86**

3) Problem 3

Search Strategy	Path Length	Execution Time(s)	Node Expansion
A* Search with h1 heuristic	**12**	42.499	17783

A* Search with Ignore Preconditions heuristic	**12**	**14.118**	**5003**
A* Search with Level Sum heuristic#	---	---	---

# the search strategy cannot be completed within 10 minutes

b. Analysis

For all three problems, **A* Search with h1 heuristic** and **Ignore Preconditions heuristic** can achieve the optimal path length within 10 minutes time limit. In addition, **A* Search with Ignore Preconditions heuristi**c is the best when considering execution time and node expansion. For A* Search with Level Sum heuristic, it expands the fewest number of nodes, but the execution time is too long, i.e. 162 seconds for problem 2.

C. Uninformed vs. Heuristic Search Strategies

Combining the above two analysis, the search strategies below can achieve optimal path length within 10 minutes.

1. Breadth First Search
2. Uniform Cost Search
3. A* Search with h1 heuristic
4. A* Search with Ignore Preconditions heuristic

For all three problems, we can re-compare the results.

1) Problem 1

Search Strategy	Path Length	Execution Time(s)	Node Expansion
Breadth First Search	**6**	**0.027**	43
Uniform Cost Search	**6**	0.031	55
A* Search with h1 heuristic	**6**	0.034	55
A* Search with Ignore Preconditions heuristic	**6**	0.033	**41**

2) Problem 2

Search Strategy	Path Length	Execution Time(s)	Node Expansion
Breadth First Search	**9**	12.726	3401
Uniform Cost Search	**9**	9.813	4761
A* Search with h1 heuristic	**9**	10.455	4761
A* Search with Ignore Preconditions heuristic	**9**	**4.168**	**1450**

3) Problem 3

Search Strategy	Path Length	Execution Time(s)	Node Expansion
Breadth First Search	**12**	93.624	14491
Uniform Cost Search	**12**	41.519	17783
A* Search with h1 heuristic	**12**	42.499	17783
A* Search with Ignore Preconditions heuristic	**12**	**14.118**	**5003**

From the results above, because it is faster and expands fewer nodes, **A* Search with Ignore Preconditions heuristic** would be the best choice overall for our Air Cargo problem

D. Conclusion

The benefits of using informed search strategies with heuristics over uninformed search are significant both in terms of speed and memory usage, which is determined by the node expansion. Moreover, the combination of A* Search with different heuristics can balance between the execution time and memory usage in a custom way.