Heuristic Analysis: **Planning Search**

In this project, we define a deterministic Air Cargo transport domain and use planning search to solve three different logistics problems. In the following analysis, we will compare and contrast the results of several search strategies, including breadth-first search (BFS), depth-first search (DFS), uniform cost search (UCS), and two variations of A* search.

Uninformed Search Strategies:

- ***Problem 1:***

Search Strategy	Path Length	Execution Time (s)	Nodes	Expansions	Goal Tests
Breath-First Search	**6**	0.0447	180	**43**	56
Depth-First Search	20	**0.0186**	**84**	84	**22**
Uniform Cost Search	**6**	0.0447	224	55	57

- ***Problem 2:***

Search Strategy	Path Length	Execution Time (s)	Nodes	Expansions	Goal Tests
Breath-First Search	**9**	15.38	31,049	3,401	4,672
Depth-First Search	1,138	**9.85**	**10,606**	**1,192**	**1,193**
Uniform Cost Search	**9**	13.44	43,206	4,761	4,763

- ***Problem 3:***

Search Strategy	Path Length	Execution Time (s)	Nodes	Expansions	Goal Tests
Breath-First Search	**12**	127.67	128,184	14,491	17,947
Depth-First Search	2,014	**27.04**	**17,558**	**2,099**	**2,100**
Uniform Cost Search	**12**	59.63	155,920	17,783	17,785

Analysis:

- *Only BFS & UCS consistently find the optimal path.*
  - Per the lectures, **BFS** will explore the **shortest paths first**. The search gradually expands all possible paths until one of them, at the optimal length, creates the goal state. This is acceptable for the problem set; however, the search will have **difficulty scaling** given problems with a longer optimal path and additional complexity.

- **UCS** will explore paths with the **lowest total cost first**. Similar to BFS, the search finds the optimal path by gradually expanding all possible paths; however, it may expand one path before another if it is cheaper to do so. This is the reason UCS **consistently has more new nodes and expansions** than BFS & DFS. It also assists in finding an optimal solution faster than BFS.

➢ *DFS will find a solution fastest, but it will not find the optimal path.*

- **DFS** will explore the **longest path first**. In this instance, the search will continue to add actions to one path until it stumbles into the goal state. For these problems, this brute force approach will **find a solution fastest**; however, **does not find the optimal path and will struggle with more complex problems**.

A* Search Strategies:

➢ ***Problem 1:***

A* Search Strategy	Path Length	Execution Time (s)	Nodes	Expansions	Goal Tests
H_1 Heuristic	**6**	0.0402	224	55	57
Ignore Preconditions	**6**	**0.0365**	170	41	43
Level-Sum Heuristic	**6**	0.9671	**50**	**11**	**13**

➢ ***Problem 2:***

A* Search Strategy	Path Length	Execution Time (s)	Nodes	Expansions	Goal Tests
H_1 Heuristic	**9**	11.90	43,206	4,761	4,763
Ignore Preconditions	**9**	**4.24**	13,303	1,450	1,452
Level-Sum Heuristic	**9**	226.55	**841**	**86**	**88**

➢ ***Problem 3:***

A* Search Strategy	Path Length	Execution Time (s)	Nodes	Expansions	Goal Tests
H_1 Heuristic	**12**	54.97	155,920	17,783	17,785
Ignore Preconditions	**12**	**17.99**	44,586	5,003	5,005
Level-Sum Heuristic	**12**	1071.84	**3,002**	**325**	**327**

Analysis:

➢ *All A* search strategies find the optimal path*

- Given the structure of A* Search, all heuristics will find the optimal path eventually

- *Ignoring preconditions will find the optimal path fastest*
  - By **intentionally disregarding non-essential information**, ignoring preconditions executes **~3x faster than the H_1 heuristic** and **50–60x faster than Level-Sum**
- *Level-Sum Heuristic utilizes the least memory but executes significantly slower*
  - The Level-Sum heuristic utilizes **~15x less memory than ignoring preconditions** and **~50x less memory than the H_1 heuristic**; however, the complexity of the heuristic causes the search to be significantly slower than its counterparts

Recommendation: A* Search – Ignoring Preconditions

- Will always yield the optimal path
- Fastest execution time of all A* search strategies
- Utilizes less memory than Breadth-First Search

Sample Paths of Optimal Length:

- **Problem 1:**
  - Load(C1, P1, SFO)
  - Fly(P1, SFO, JFK)
  - Unload(C1, P1, JFK)
  - Load(C2, P2, JFK)
  - Fly(P2, JFK, SFO)
  - Unload(C2, P2, SFO)
- **Problem 2:**
  - 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)

- **Problem 3:**
  - 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)