Planning Algorithm Analysis Report

Analysis Charts

1. Nodes Expanded vs. Number of Actions in Domain

Algorithm	20 Actions	72 Actions	88 Actions	104 Actions
Uniform Cost Search	240	46,618	161,936	1,066,413
Greedy Best-First (h_unmet_goals)	29	170	230	280
Greedy Best-First (h_pg_levelsum)	28	86	126	165
A* (h_unmet_goals)	206	22,522	65,711	328,509
A* (h_pg_levelsum)	122	3,426	3,403	12,210

2. Search Time vs. Number of Actions in Domain

Algorithm	20 Actions	72 Actions	88 Actions	104 Actions
Uniform Cost Search	0.0050s	0.2349s	0.5955s	3.5885s
Greedy Best-First (h_unmet_goals)	0.0007s	0.0038s	0.0071s	0.0105s
Greedy Best-First (h_pg_levelsum)	0.1146s	0.3027s	0.8909s	1.1658s
A* (h_unmet_goals)	0.0046s	0.2632s	0.5215s	1.8458s
A* (h_pg_levelsum)	0.0822s	7.4680s	12.1527s	67.4415s

3. Plan Length vs. Problem Size

Algorithm	Problem 1	Problem 2	Problem 3	Problem 4
Uniform Cost Search	6	9	12	14
Greedy Best-First (h_unmet_goals)	6	9	15	18
Greedy Best-First (h_pg_levelsum)	6	9	14	17
A* (h_unmet_goals)	6	9	12	14
A* (h_pg_levelsum)	6	9	12	15

Analysis and Recommendations

1. Planning in a Very Restricted Domain with Real-Time Constraints

For a domain with few actions and real-time requirements, the most appropriate algorithms would be:

1. Greedy Best-First Search with h_unmet_goals
2. A* Search with h_unmet_goals

Rationale: These algorithms consistently showed the fastest execution times across all problem sizes, especially in smaller domains. They provide a good balance between speed and solution quality, which is crucial for real-time operations.

2. Planning in Very Large Domains

For planning in very large domains, like UPS delivery route planning, the most appropriate algorithms would be:

1. Greedy Best-First Search with h_pg_levelsum
2. A* Search with h_unmet_goals

Rationale: These algorithms demonstrated better scalability as the problem size increased. Greedy Best-First Search with h_pg_levelsum showed the least growth in node expansions, while A* with h_unmet_goals provided a good balance between optimality and performance in larger domains.

3. Planning Problems Requiring Optimal Plans

For problems where finding optimal plans is crucial, the most appropriate algorithms would be:

1. A* Search with h_pg_levelsum
2. Uniform Cost Search

Rationale: A* Search with h_pg_levelsum consistently found optimal or near-optimal solutions across all problem sizes. While it's slower, it ensures optimality. Uniform Cost Search, being complete and optimal, is also suitable but becomes impractical for larger problems due to its exponential time complexity.

Conclusion

The choice of algorithm depends on the specific requirements of the planning problem. For small, time-sensitive domains, greedy approaches work well. For large-scale problems, more informed heuristics like h_pg_levelsum show better scalability. When optimality is crucial, A* with strong heuristics or exhaustive searches like Uniform Cost Search are necessary, despite their higher computational cost.