Heuristic Analysis

Chenguang Yang

This report compares multiple search algorithms without any heuristic, and Planning-Graph-based search with multiple heuristic functions, in the air cargo problem settings. Four performance metrics are used for comparison

* Optimality: is the solution taking least number of actions to achieve the goal?
* Time Elapsed: how long does it take to arrive at a feasible solution?
* # Node Expansion: how many states does the search process expand?
* # Goal Test: how many states does the search process check if goal is met?

# Optimal Plan

Three air cargo problems in this project are all easy enough to be solved optimally by hand. But be aware that the sequence of the plan doesn’t reflect exact sequence of operation in real situation, since time duration or parallel operation constraints, etc. are out of the scope of planning problem here. For example, the following 2 solutions are treated equivalent

Here are optimal plans (with possible variations as shown above) for each problem, and optimality of solutions are checked against them.

|  |  |  |
| --- | --- | --- |
| Air Cargo Problem 1 | Air Cargo Problem 2 | Air Cargo Problem 3 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Performance Comp – Non Heuristic

I picked Uniform Cost Search as the 3rd algorithm to compare against Breadth First Search and Depth First Graph Search

|  |  |
| --- | --- |
|  |  |
|  |  |

Main take-away from the 3 graphs above is that Depth First Graph Search takes way less node expansions and much less time to find a solution, while Breadth First and Uniform Cost are on par with each other. Such difference becomes bigger when the planning problem size increases.

However, the solution from Depth First Graph are not optimal, while both Breadth First and Uniform Cost achieved optimality in all 3 air cargo problems. These optimality behavior generally holds true per video lecture #24 of (Udacity, n.d.). And the charts on the next page show us solutions from Depth First are much (so much I have to use data labels) more costly than optimal plan.

Another interesting observation is that Uniform Cost took about half the time as Breadth First (They both achieved optimality) although Uniform Cost expanded more nodes and did more goal tests. I think it’s mainly due to their difference in the data structure to store “frontier” nodes.

|  |  |  |
| --- | --- | --- |
| Search Algorithm | Frontier Nodes | Without Heuristics |
| Breadth First Search | FIFO Queue | Still maintains FIFO order |
| Uniform Cost Search | Priority Queue | No order required on queue |

|  |  |
| --- | --- |
|  |  |
|  |  |

# Performance Comp – Heuristic

For more revealing result, I include not only A\*-IgnorePrecondition, A\*-LevelSum, but also A\*-NonHeuristic in this section. And note that A\* is equivalent to Uniform Cost without heuristic function

return best\_first\_graph\_search(problem, lambda n: n.path\_cost + h(n))

return best\_first\_graph\_search(problem, lambda node: node.path\_cost)

|  |  |
| --- | --- |
|  |  |
|  |  |

All A\* Search algorithms find optimal plans for all three problems. This is because IgnorePrecondition and LevelSum are both admissible and consistent heuristics, and A\* search is complete and optimal when its heuristic function is admissible and consistent (Russell & Norvig, 2010, pp. 93-95).

Main take-away from the 3 graphs above is, A\*-IgnorePrecondition takes less time, less node expansion and less goal test than A\*-NonHeuristic to reach optimality. A\*-LevelSum has even less node expansion and goal tests than A\*-IgnorePrecondition, but costs significantly more time than A\*-NonHeuristic.

Quick clarification on Air Cargo Problem 2 and 3 chart where **“node expansion” line** is not visible. For all 3 search algorithms here, there is always # node expansion = # goal test – 2, so the blue lines overlap with red lines. I think the constant difference of 2 comes from specific procedure of node expansion, goal testing, frontier updates, etc. of Best Graph Search. And A\* Searches + any heuristics are all special cases of it.

# Conclusion

**I determine A\*-IgnorePrecondition to be the best search algorithm** based on several practical considerations, and the concept of “domination”. More specifically, algorithm X is dominated by another algorithm Y, if X is not better than Y in all 3 metrics of all 3 planning problems. And better or worse is evaluated by the ranking (1 being winner) among search algorithms per metric per problem.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Problem | Metric | B 1st | D 1st Graph | A\*H=None | A\*Precond | A\*LevelSum |
| 1 | Node | 4 | 2 | 5 | 3 | 1 |
| 1 | Time | 2 | 1 | 3 | 3 | 5 |
| 1 | Optimality | 1 | 5 | 1 | 1 | 1 |
| 2 | Node | 4 | 2 | 5 | 3 | 1 |
| 2 | Time | 4 | 1 | 3 | 2 | 5 |
| 2 | Optimality | 1 | 5 | 1 | 1 | 1 |
| 3 | Node | 4 | 2 | 5 | 3 | 1 |
| 3 | Time | 4 | 1 | 3 | 2 | 5 |
| 3 | Optimality | 1 | 5 | 1 | 1 | 1 |

* A\*-NoneHeuristic is dominated by A\*-IgnorePrecondition, that’s eliminated first.
* The quality of the solution from Depth First Search is far worse than optimality, and in practice when planning is more complicated than Air Cargo problem 3, it’s obviously unacceptable.
* Breadth First Search is very close to be dominated by A\*-IgnorePrecondition, and the exception of time elapsed in the 1st problem, 0.025s vs. 0.030s is minor enough to think of Breadth First Search as practically dominated by A\*-IgnorePrecondition.

Despite Breadth First Search is eliminated, the **situation where heuristic search planning is not necessarily better than non-heuristic search planning exists,** which can be generalized as situations where the scale of the problem is small enough that improved efficiency of heuristic functions doesn’t compensate additional calculation of heuristic values.

Only A\*-IgnorePrecondition and A\*-LevelSum are left to compete for winning heuristic. One is faster, the other have smaller # of node expansions, but takes time to build Planning Graph. There is no straightforward numerical weight or score to assign, therefore my conclusion at the beginning of the section is from my qualitative argument. I think current trend in practice is storage being larger and cheaper while business decision expected to be more and more responsive. That’s why I recommend A\*-IgnorePrecondition as the best heuristic used in these planning problems.

# References

Russell, S. J., & Norvig, P. (2010). *Artificial Intelligence-A Modern Approach.* Prentice Hall.

Udacity. (n.d.). *AI Nanodegree Lession 8: Search*. Retrieved from classroom.udacity.com.