Heuristic Analysis

The following evaluations are executed with PyPy 5.8.0-beta0 on MacBook Pro (15-inch, Mid 2015).

Non-heuristic Search

Three non-heuristic search approaches are evaluated. Namely, breadth-first search (BFS), depth-first search (DFS), and uniform cost search (UCS).

Number of Node Expansions

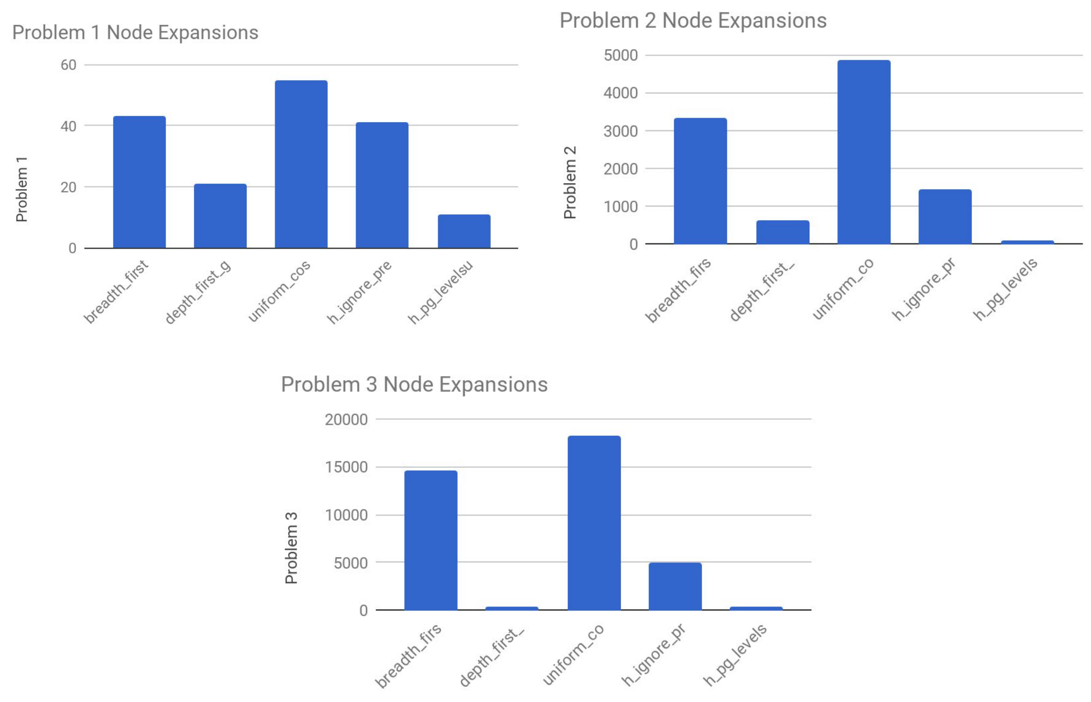

Chart A: Number of Node Expansions

As is shown in Chart A, the order of number of node expansions are consistent across three problems - UCS > BFS > DFS.

Time Elapsed

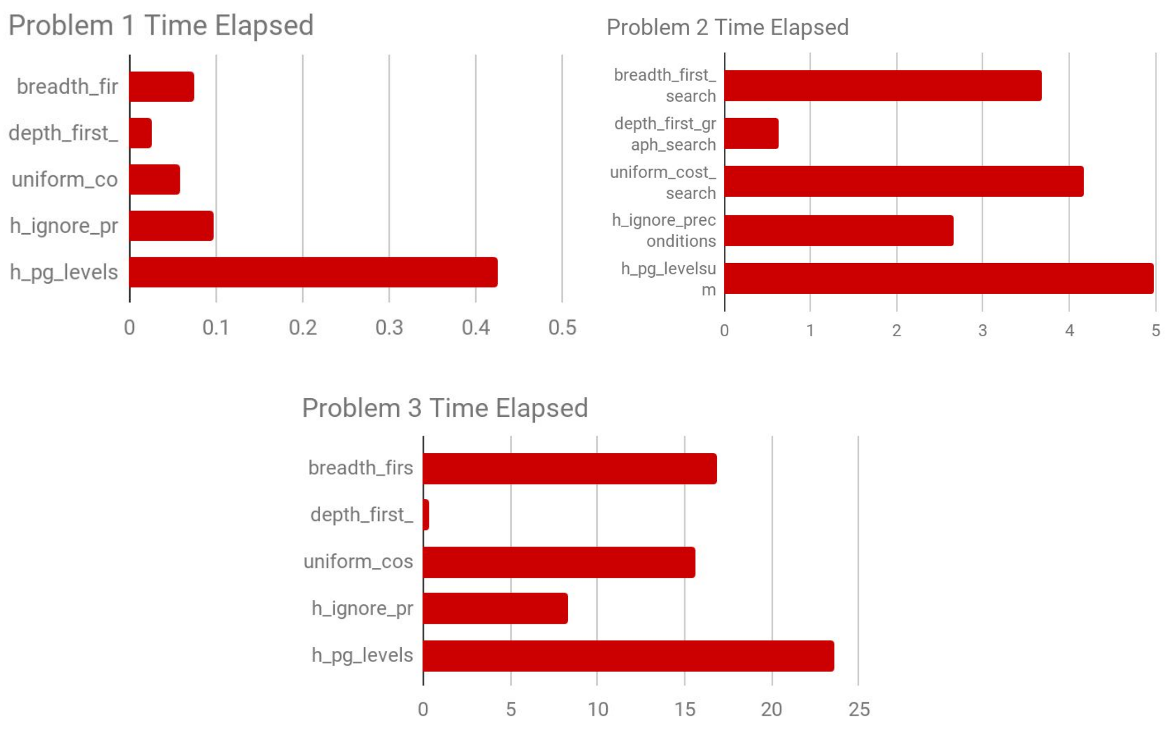

Chart B: Time Elapsed

Chart B shows that DFS takes the least amount of time. The elapsed time of UCS and BFS is similar.

Optimality

BFS and UCS are guaranteed to produce optimal result, while DFS is not.

Heuristic Search

A* search is evaluated with two heuristic functions - “ignores preconditions” and “level-sum”.

Number of Node Expansions

As is shown in Chart A, “ignore preconditions” expands more nodes than “level-sum” does.

Time Elapsed

Chart B shows that “level-sum” takes more time to execute than “ignore preconditions”.

Optimality

Both heuristic searches guarantee optimality.

Discussion

1. What was the best heuristic used in these problems?

The answer depends entirely on the definition of “best” based on the context of the problem domain. If “best” means the fastest, the faster heuristic function is “ignore preconditions”. If memory is a scarce resource when solving a problem, “level-sum” is more suitable.

2. Was it better than non-heuristic search planning methods for all problems?

It was better than some non-heuristic search planning in some regards but not in all. For example, on one hand, the two heuristic search approaches guarantee optimality and DFS does not; on the other hand, DFS outperforms them in terms of time elapsed.