Overview

Algorithm

Features:

Queue lookup:

Using sets greatly increases the time taken to verify if the node is already in the queue

Using the first search to find the other searches

Probability that the best solutions are close to the end node:

Since Dijkstra finds the most optimal path. We can work with the fact that the path to Nodei in the solution is optimal. Like Yens algorithm we can find alternative sub graphs as we remove connections in the original graph. Unlike Yen’s algorithm that is aimed at optimality we can speed up the process significantly by removing nodes from the goal node rather than the first node. This is somewhat of a heuristic considering most optimal paths will alter towards the end since we have found the most optimal path through most of the graph.

Insert some sort of diagram

Backtracking until number of solutions are complete:

The idea is to avoid unnecessary iterations that guarantee optimality and provide a more generalised approach. To solve issues where the original path contains single node options for the last x nodes we need to implement a sort of backtracking where we can step back enough nodes that we have at least k options to get to the path

Retaining the information from the first search to continue other avenues for searching:

To best make use of the first search we should store or implement path permutations within the Dijkstra search to make use of the already computed best paths. More so, once the search is complete we can save the path, remove the weight from the final nodes and continue the search as though that option didn’t exist

Results:

Show times, costs, k, alternative data to prove optimality

Analysis:

Modify Dijkstras complexity

Space complexity

Compare to Yens implementation