Combinatorial Optimization by Shortest Path Algorithms

1. Introdution:

\_In graph theory, the shortest path problem is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized.

\_The problem of finding the shortest path between two intersections on a road map may be modeled as a special case of the shortest path problem in graphs, where the vertices correspond to intersections and the edges correspond to road segments, each weighted by the length of the segment.

\_In graphs, the distance between two nodes s and t (source and target, respectively) is defined as follows. If s and t are connected by an edge, their distance is 1. If they are not directly connected, the distance is defined by the length of a shortest path between s and t, which is a sequence of adjacent edges. In a weighted graph, the length of a path is defined by the sum of the weights of the edges on the path. Consequently, shortest paths are defined with respect to these weights. Note that, in weighted graphs, even if two nodes are connected by an edge, depending on its weight, the edge is not necessarily part of any shortest path.

\_So far, in the new graph only consider the weights of the edges, the vertices independently, in which the length of the path is merely the sum of the weights of the edges and vertices on that path. However, in many practical problems, the weight at a vertex is not the same for every path that passes that peak, but also depends on the edge going to and from the top. In the paper, the general graph model is defined. This paper’s main objective is to develop and demonstrate the algorithm to find the shortest path between two peaks and the algorithm to find the shortest path from one peak to the other peaks on the general transport network.

\_In this paper, we will use Dijkstra’s Algorithm to solve the shortest path problem.

1. Related Work:

• Implement the algorithm that can be adaptive to any input data change.

• Read input data from the nodes and edges, and predict the shortest path.

• Design the interface for input from user by representing the feature like: Add, Insert, Delete, Dropout, Change parameters, ...

• Visualize the networks.

1. Approaches:

\_As mentioned earlier, a graph can be used to represent a map where the cities are represented by vertices and the routes or roads are represented by edges within the graph. In this section, a graph representation of a map is explained further, and brief descriptions and implementations of Dijkstra’s Algorithm being studied are presented.

\_Two vertices are adjacent when they are both incident to a common edge. A path in an undirected graph is a sequence of vertices {\displaystyle P=(v\_{1},v\_{2},\ldots ,v\_{n})\in V\times V\times \cdots \times V} {\displaystyle P=(v\_{1},v\_{2},\ldots ,v\_{n})\in V\times V\times \cdots \times V} such that {\displaystyle v\_{i}} v\_{i} is adjacent to {\displaystyle v\_{i+1}} v\_{i+1} for {\displaystyle 1\leq i<n} 1\leq i<n. Such a path {\displaystyle P} P is called a path of length {\displaystyle n-1} n-1 from {\displaystyle v\_{1}} v\_{1} to {\displaystyle v\_{n}} v\_{n}. (The {\displaystyle v\_{i}} v\_{i} are variables; their numbering here relates to their position in the sequence and needs not to relate to any canonical labeling of the vertices.)

\_Let {\displaystyle e\_{i,j}} e\_{i,j} be the edge incident to both {\displaystyle v\_{i}} v\_{i} and {\displaystyle v\_{j}} v\_{j}. Given a real-valued weight function {\displaystyle f:E\rightarrow \mathbb {R} } f:E\rightarrow \mathbb {R} , and an undirected (simple) graph {\displaystyle G} G, the shortest path from {\displaystyle v} v to {\displaystyle v'} v' is the path {\displaystyle P=(v\_{1},v\_{2},\ldots ,v\_{n})} P=(v\_{1},v\_{2},\ldots ,v\_{n}) (where {\displaystyle v\_{1}=v} v\_{1}=v and {\displaystyle v\_{n}=v'} v\_{n}=v') that over all possible {\displaystyle n} n minimizes the sum {\displaystyle \sum \_{i=1}^{n-1}f(e\_{i,i+1}).} \sum \_{i=1}^{n-1}f(e\_{i,i+1}). When each edge in the graph has unit weight or {\displaystyle f:E\rightarrow \{1\}} f:E\rightarrow \{1\}, this is equivalent to finding the path with fewest edges.

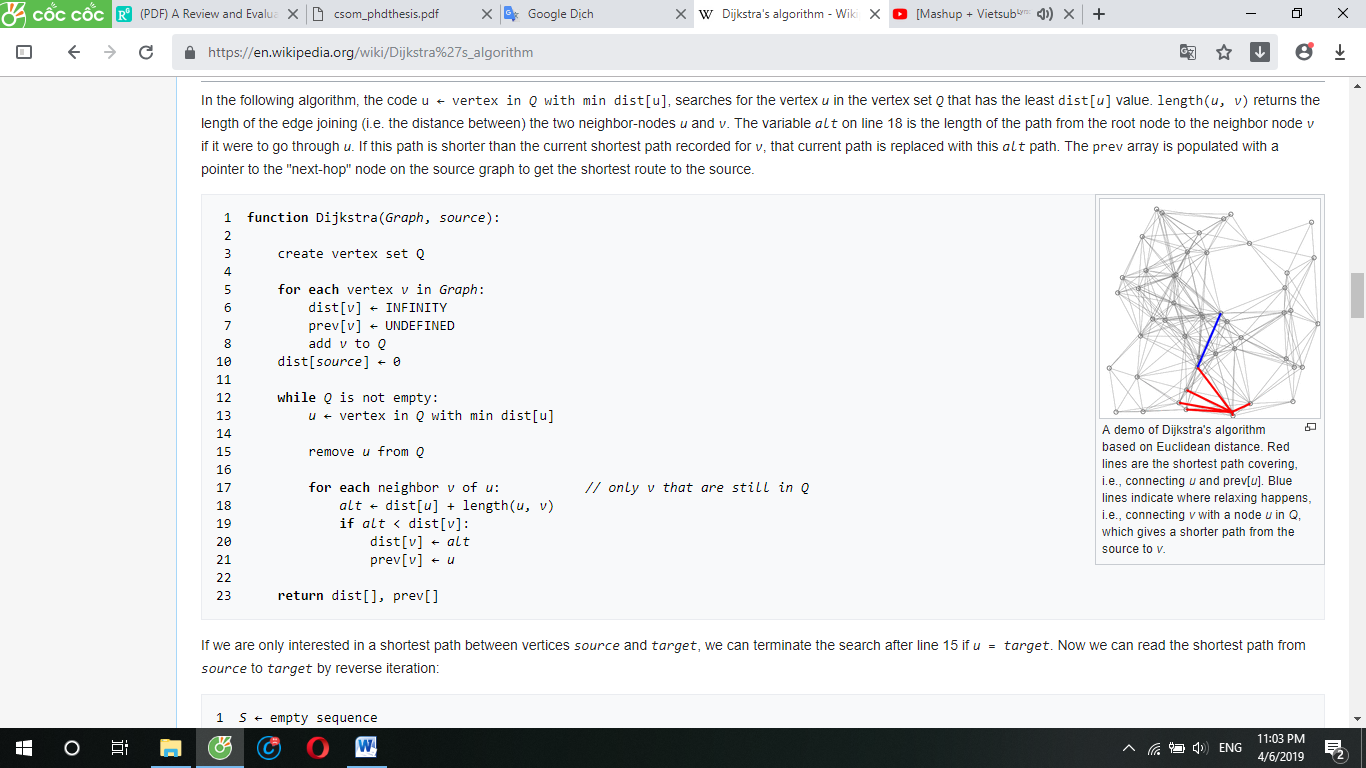
\_The problem is also sometimes called the single-pair shortest path problem, to distinguish it from the following variations:

\_The single-source shortest path problem, in which we have to find shortest paths from a source vertex v to all other vertices in the graph.

The single-destination shortest path problem, in which we have to find shortest paths from all vertices in the directed graph to a single destination vertex v. This can be reduced to the single-source shortest path problem by reversing the arcs in the directed graph.

The all-pairs shortest path problem, in which we have to find shortest paths between every pair of vertices v, v' in the graph.

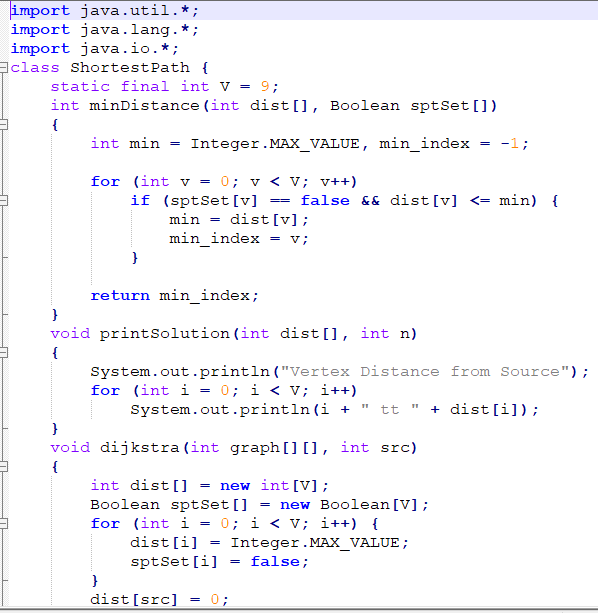
\_In the following algorithm, we have the pseudocode of Dijkstra’s algorithm. The code u ← vertex in Q with min dist[u], searches for the vertex u in the vertex set Q that has the least dist[u] value, length(u, v) returns the length of the edge joining (i.e. the distance between) the two neighbor-nodes u and v. The variable alt on line 18 is the length of the path from the root node to the neighbor node vif it were to go through u. If this path is shorter than the current shortest path recorded for v, that current path is replaced with this alt path. The prev array is populated with a pointer to the "next-hop" node on the source graph to get the shortest route to the source.

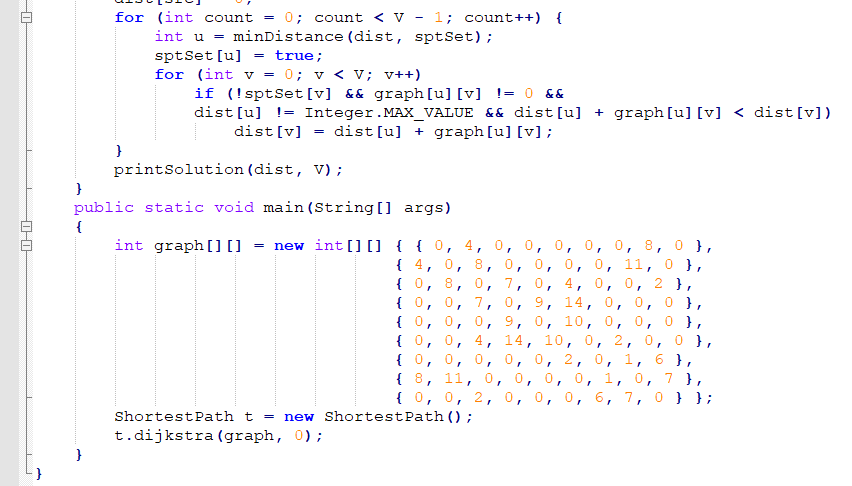


**Figure 3.1**: Pseudocode of Dijkstra’s algorithm

\_When the algorithm completes, prev[] data structure will actually describe a graph that is a subset of the original graph with some edges removed. Its key property will be that if the algorithm was run with some starting node, then every path from that node to any other node in the new graph will be the shortest path between those nodes in the original graph, and all paths of that length from the original graph will be present in the new graph. Then to actually find all these shortest paths between two given nodes we would use a path finding algorithm on the new graph, such as depth-first search.

1. Experiments and Results:





1. Conclusion:

References:

<https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm>

<https://en.wikipedia.org/wiki/Shortest_path_problem>