Dijkstra’s Algorithm

Dijkstra’s Algorithm assigns to every node \( j \) a pair of labels \((p_j, d_j)\), where \( p_j \) is the node preceding node \( j \) in the existing shortest path from 1 to \( j \), \( d_j \) is the length of this shortest path. Some of the labels are called temporary, i.e. they could change at a future step; some labels are called permanent, i.e. they are fixed and the shortest path from 1 to a node that is permanently labeled has been found.

We denote by \( d_{jk} \) the length of arc \((j, k)\).

**Step 1.** Label node 1 with the permanent labels \((\emptyset, 0)\). Label every node \( j \), such that \((1, j)\) is an arc in the graph, with temporary labels \((1, d_{1j})\). Label all other nodes in the graph with temporary labels \((\emptyset, \infty)\).

**Step 2.** Let \( j \) be a temporarily labeled node with the minimum label \( d_j \), i.e. \( d_j = \min\{d_l : \text{node } l \text{ is temporarily labeled}\} \). For every node \( k \), such that \((j, k)\) is in the graph, if \( d_k > d_j + d_{jk} \), relabel \( k \) as follows:

\[
p_k = j, d_k = d_j + d_{jk}.
\]

Consider the labels of node \( j \) to be permanent.

**Step 3.** Repeat step 2 until all nodes in the graph are permanently labeled.

The shortest paths can be found by reading labels \( p_j \) backwards, i.e. from node \( j \) to node 1.