DIJKSTRA'S ALGORITHM

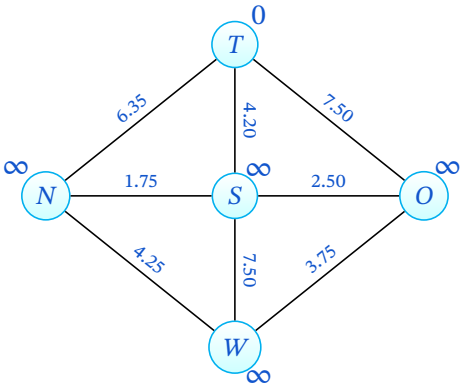
Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks.

1. Mark all nodes unvisited.
2. Set *tentative distance* for all nodes.
   - 2.1 0 for initial node.
   - 2.2 ∞ for all other nodes.
3. **Select node with lowest *tentative distance***.
4. If it target node, we are done!
5. For every node connected to this node:
   - 5.1 **Calculate new *tentative distance***:

   Current node's distance + path length

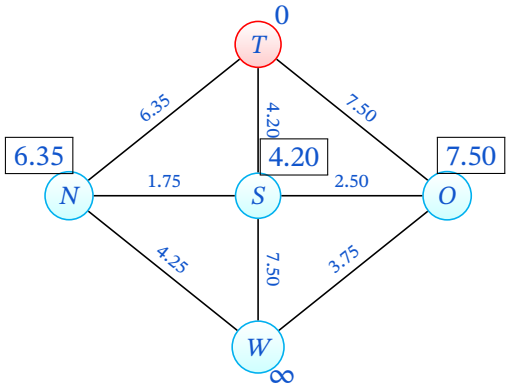
   - 5.2 Set if it lower then current *tentative distance*.
6. Mark selected node as visited.
   We no longer consider visited nodes.
7. Go to step 3.

Step 1: set an initial *tentative distance* for all nodes.

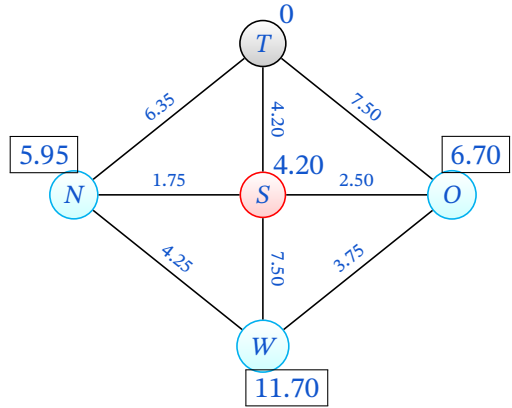


DIJKSTRA'S ALGORITHM

Step 2: select a node with lowest *tentative distance* (T) and update all connected nodes.
For S: $0 + 4.20 = 4.20 < \infty$ - update
For N: $0 + 6.35 = 6.35 < \infty$ - update
For O: $0 + 7.50 = 7.50 < \infty$ - update

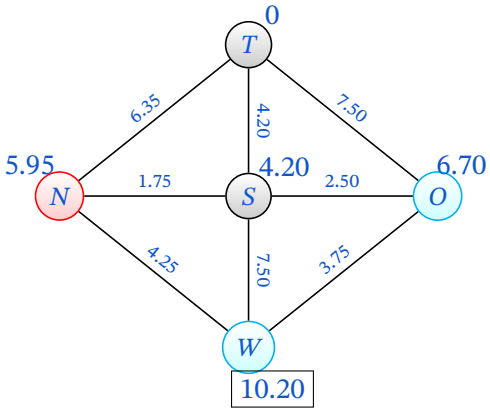


Step 3: Mark current node (T) as visited, select a node with lowest *tentative distance* (S) and update all connected nodes.
For N: $4.20 + 1.75 = 5.95 < 6.35$ - update
For O: $4.20 + 2.50 = 6.70 < 7.50$ - update
For W: $4.20 + 7.50 = 11.70 < \infty$ - update

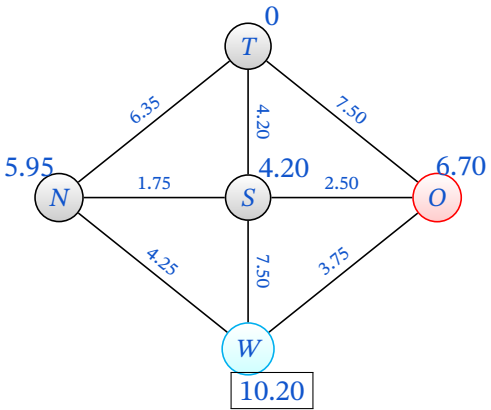


DIJKSTRA'S ALGORITHM

Step 4: Mark current node (S) as visited, select a node with lowest *tentative distance* (N) and update all connected nodes.
For W: $5.95 + 4.25 = 10.20 < 11.70$ - update

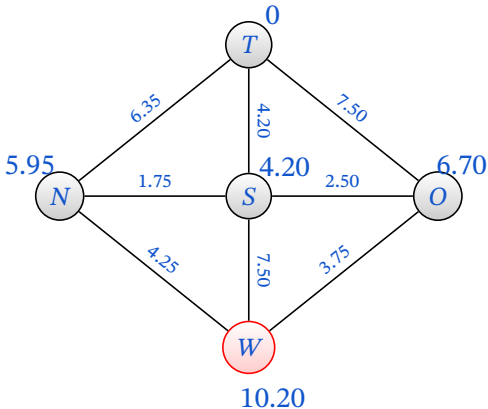


Step 5: Mark current node (N) as visited, select a node with lowest *tentative distance* (O) and update all connected nodes.
For W: $6.70 + 3.75 = 10.45 > 10.20$ - **DON'T update**



DIJKSTRA'S ALGORITHM

Step 6: The final node becomes current (S), end of algorithm.
So the shortest distance from T to W is 10.20.



Dijkstra's algorithm consider each node *no more then once* and consider every road *no more then once*.
To restore the shortest path, the algorithm need to be modified: each node needs to remember the last current node to update its distance.
If we did that, then
T is the initial node,
S remembers T,
N remembers S,
O remembers S,
W remembers N.
Walking back we restore our path: $T - S - N - W$.