**Dijkstra’s shortest path algorithm**

Dijkstra's algorithm makes use of weights of the edges for finding the path that minimizes the total distance (weight) among the source node and all other nodes. The goal is to find the shortest path from a source to all other vertices.

We can understand Dijkstra's search algorithm by taking an example of calculating the minimum time taken to travel from one town to the other via different towns. The vertices can be seen as towns and the edges as roads. The vertices and edges combined forms a graph, moreover fixing edge values forms a weighted graph. We can fix one town as a source i.e. starting point. The source town will have the value 0 and the other towns will hold the value infinity initially. Mark the source town as explored and start moving along the roads to reach other towns. As a new town is reached give it a value which represents the time taken to reach there. When lowest value is reached after travelling through various roads, mark the town as explored. Repeat this for every town until the shortest path is found for each vertex. This algorithm is also known as the single-source shortest path algorithm. The final graph containing the shortest paths is known as shortest path graph (SPG).

**Algorithm:**

* for each vertex v:

dist[v] = infinity

dist[source] = 0

set all vertices to unexplored

while destination not explored:

v = least valued unexplored vertex

set v to explored

for each edge (v, w):

If dist[v] + len(v, w) < dist[w]:

dist[w] = dist[v] + len(v, w)

**Applications of Dijkstra’s algorithm**

It is vastly effective in measuring the least possible distance and checking direction which can work pretty well in different regions and terrains, discovering map locations pointing to the vertices of a graph, calculating traffic, mapping directions like Google maps etc.

This is also hugely implemented in the data conducting domains like networking and telecommunication to detect obstacles which decrease transmission efficiency.

**Properties and Advantages**

1. One of the main advantages of Dijkstra’s algorithm is its little complexity which is almost linear.
2. It only works for directed, weighted graphs and all edges should have non-negative values.

(Directed graph: if for every pair of connected nodes, you can only go from one node to another in a specific direction.)

**Disadvantages**

1. It does an obscured exploration that consumes a lot of time while processing.
2. It is unable to handle negative edges.
3. As it heads to the acyclic graph, so can’t achieve the accurate shortest path.
4. There is a need to maintain tracking of vertices that have been visited.