Dijkstra's Algorithm

Dr. Monika Patel
Assistant Professor
Dept of Computer Science
Durga Mahavidyalaya
Raipur (CG)

Dijkstra's Algorithm is a greedy algorithm used in graph theory (a part of discrete mathematics) to compute the shortest path from a source vertex to all other vertices in a weighted graph with non-negative edge weights.

Graph (G = (V, E)):

A set of vertices (V) and edges (E) connecting pairs of vertices.

Weighted Graph:

Each edge has an associated weight or cost.

Path:

A sequence of edges that connects a sequence of vertices.

Shortest Path:

The path with the minimum total weight between two vertices.

Greedy Algorithm:

Makes the locally optimal choice at each step.

Algorithm Steps:

Let G = (V, E) be a graph with vertices V and edges E.

1. Initialize:

Set the distance to the source vertex as 0.

Set the distance to all other vertices as ∞.

Mark all vertices as unvisited.

2. Set the source as the current node.

3. For the current node:

Check all its unvisited neighbors.

Calculate their tentative distances through the current node.

If the calculated distance is smaller, update it.

4. Mark the current node as visited.

5. Select the unvisited node with the smallest tentative distance and set it as the new current node.

6. Repeat steps 3 to 5 until all vertices are visited or the destination is reached.

Pseudocode:

```
function Dijkstra(Graph, source):
  for each vertex v in Graph:
      dist[v] := infinity
      previous[v] := undefined
  dist[source] := 0
  Q := set of all vertices in Graph
   while Q is not empty:
      u := vertex in Q with smallest dist[u]
```

```
    remove u from Q
     for each neighbor v of u:
        alt := dist[u] + weight(u, v)
        if alt < dist[v]:
            dist[v] := alt
            previous[v] := u
```

Properties:

Works only with non-negative edge weights.

Guarantees optimal solution.

Uses greedy approach for selecting the next node.

Time Complexity:

Using simple array: O(V^2)

Using min-priority queue (binary heap): O((V + E) log V)

Applications:

Network routing protocols (like OSPF)

GPS navigation systems

Game development (path finding)

AI and robotics

Limitations:

Does not work with negative edge weights.

For negative weights, use Bellman-Ford Algorithm.