Graph Theory

All-to-All Shortest Path Problem

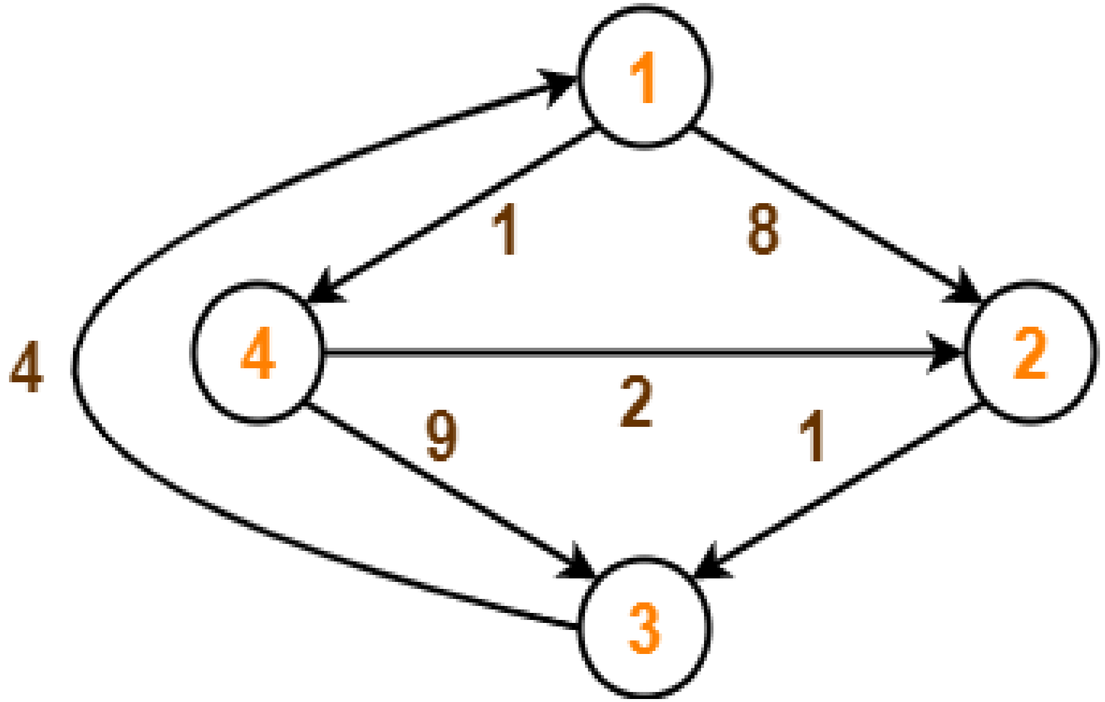
Floyd-Warshall Algorithm

- **Floyd-Warshall Algorithm** is an algorithm for finding the shortest path between all the pairs of vertices in a weighted graph.
- This algorithm works for both the directed and undirected weighted graphs.

Algorithm

```
n = no of vertices
A = matrix of dimension n*n
for k = 1 to n
    for i = 1 to n
        for j = 1 to n
            A^k[i, j] = min (A^(k-1)[i, j], A^(k-1)[i, k] + A^(k-1)[k, j])
return A
```

Example



Solution-

Step-0:

Remove all the self-loops and parallel edges (keeping the lowest weight edge) from the graph.

In the given graph, there are neither self-edges nor parallel edges.

Steps 1: Create a matrix D_0 of dimension n*n where n is the number of vertices.

Write the initial distance matrix.

It represents the distance between every pair of vertices in the form of given weights.

For diagonal elements (representing self-loops), distance value = 0.

For vertices having a direct edge between them, distance value = weight of that edge.

For vertices having no direct edge between them, distance value = ∞.

$$D_0 = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 8 & \infty & 1 \\ \infty & 0 & 1 & \infty \\ 4 & \infty & 0 & \infty \\ \infty & 2 & 9 & 0 \end{bmatrix} \end{array}$$

- **Step 2:** Now, create a matrix D_1 using matrix D_0. The elements in the first column and the first row are left as they are. The remaining cells are filled in the following way.
  Let k be the intermediate vertex in the shortest path from source to destination. In this step, k is the first vertex.
- D[i][j] is filled with (D[i][k] + D[k][j]) if (D[i][j] > D[i][k] + D[k][j]).

$$
D_1 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array}
\begin{array}{c}
\begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\
\left[\begin{array}{cccc}
0 & 8 & \infty & 1 \\
\infty & 0 & 1 & \infty \\
4 & 12 & 0 & 5 \\
\infty & 2 & 9 & 0
\end{array}\right]
\end{array}
$$

- **Step 3**: In a similar way, D_2 is created using A_1. The elements in the second column and the second row are left as they are.

  In this step, k is the second vertex (i.e. vertex 2).

$$D_2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & 9 & 1 \\ 2 & \infty & 0 & 1 & \infty \\ 3 & 4 & 12 & 0 & 5 \\ 4 & \infty & 2 & 3 & 0 \end{array}$$

- **Step 4 and 5**: Similarly, D_3 and D_4 is also created

$$D_3 = \begin{array}{c|c} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 0 & 8 & 9 & 1 \\ 5 & 0 & 1 & 6 \\ 4 & 12 & 0 & 5 \\ 7 & 2 & 3 & 0 \end{bmatrix} \end{array}$$

$$D_4 = \begin{array}{c|c} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 0 & 3 & 4 & 1 \\ 5 & 0 & 1 & 6 \\ 4 & 7 & 0 & 5 \\ 7 & 2 & 3 & 0 \end{bmatrix} \end{array}$$

- The last matrix D_4 represents the shortest path distance between every pair of vertices. For example, from vertex 1 to vertex 4, cost is 1 and 1 to 3 is 4 and so on.

Find the shortest path between each pair of vertices.