Floyd-Warshall Algorithm

Algorithm Description

The Floyd-Warshall algorithm is a dynamic programming algorithm used to find the shortest paths between all pairs of vertices in a weighted graph (both directed and undirected). The graph should not contain negative weight cycles.

—

Algorithm

Algorithm 1 Floyd-Warshall Algorithm

Input: Adjacency matrix $dist$ of size $V \times V$ representing the graph
Output: Shortest distances between all pairs of vertices
function FLOYDWARSHALL($dist$)
    for $k \leftarrow 1$ to V **do** ▷ Iterate over all intermediate vertices
        for $i \leftarrow 1$ to V **do** ▷ Iterate over all source vertices
            for $j \leftarrow 1$ to V **do** ▷ Iterate over all destination vertices
                $dist[i][j] \leftarrow \min(dist[i][j], dist[i][k] + dist[k][j])$
            end for
        end for
    end for
    return $dist$
end function

—

Explanation of the Algorithm

1. **Initialization**: - The input graph is represented as an adjacency matrix $dist$, where $dist[i][j]$ is the weight of the edge from vertex i to vertex j. If there is no edge, $dist[i][j]$ is initialized to ∞. - $dist[i][i]$ is initialized to 0 for all vertices i.

2. **Dynamic Programming Transition**: - For each vertex k (acting as an intermediate vertex), the algorithm updates the shortest distance between every pair of vertices (i, j). - The key update formula is:

$$dist[i][j] = \min(dist[i][j], dist[i][k] + dist[k][j])$$

3. **Final Output**: - After V iterations, $dist[i][j]$ contains the shortest distance from vertex i to vertex j.

—

Time Complexity Analysis

1. **Outer Loop**: - The outer loop runs V times, iterating over all intermediate vertices.

2. **Inner Loops**: - For each pair (i, j), the algorithm checks and updates the shortest path via vertex k. This takes $\mathcal{O}(1)$ time for each pair.

3. **Overall Complexity**: - The total number of iterations is $V \times V \times V = V^3$. - Therefore, the time complexity is:

$$\mathcal{O}(V^3)$$

4. **Space Complexity**: - The space complexity is $\mathcal{O}(V^2)$ to store the adjacency matrix.

—

Advantages and Limitations

- **Advantages**: - Finds shortest paths between all pairs of vertices in a single execution. - Handles graphs with negative edge weights (but not negative weight cycles).

- **Limitations**: - Inefficient for sparse graphs compared to algorithms like Dijkstra's.

—