DESIGN ANALYSIS AND ALGORITHM

HANDSON-14

RASHMITHA RAMASANI

ID:1002233393

Dijkstra's Algorithm:

- The algorithm uses a **priority queue** (implemented with a heap) to always process the node with the smallest known distance.
- It updates the shortest distance to each neighboring node if a shorter path is found through the current node.
- The distances dictionary stores the shortest distance from the start node to each node.
- The predecessors dictionary helps to reconstruct the shortest path.
- **Time Complexity**: O(E log V), where **E** is the number of edges and **V** is the number of vertices.
- **Space Complexity**: O(V + E) due to the storage of distances, predecessors, and the priority queue.

2. **Bellman-Ford Algorithm**, which is used to compute the shortest path from a single source node to all other nodes in a graph. The algorithm supports graphs with negative edge weights and detects negative weight cycles.

Features

- **Handles Negative Edge Weights**: Unlike Dijkstra's algorithm, Bellman-Ford works with graphs containing negative edge weights.
- **Detects Negative Weight Cycles**: Identifies if any negative weight cycle exists in the graph and reports it.
- **Single Source Shortest Paths**: Calculates the shortest path from the source node to all other nodes.
- **Aspect Complexity**
  time complexity: O(V·E)

  space complexity: O(V+E)

3. **Floyd-Warshall Algorithm** is an all-pairs shortest path algorithm that computes the shortest paths between all pairs of nodes in a graph. It is particularly suitable for dense graphs and can handle negative edge weights (but not negative weight cycles).

Time Complexity

- The algorithm uses three nested loops to iterate through all pairs of nodes with each possible intermediate node: O(V3)
  - V: Number of vertices in the graph.

Space Complexity

- The algorithm requires a V×VV \times VV×V matrix to store distances: O(V2)