IIIT Bhubaneswar
Imagine, Innovate, Inspire
(A University established by Government of Odisha)

MTech CSE – 1st Semester

Student Name: SIDDHARTH KUMAR
Student ID: A125021

Question 9

Explain why Dijkstra's algorithm cannot be applied to graphs containing negative edge weights.

Answer:

Short Answer

Dijkstra's algorithm cannot be used on graphs with negative edge weights because it assumes that once the shortest distance to a vertex is determined, it will never decrease. Negative edge weights violate this assumption, making the algorithm incorrect.

Working Principle of Dijkstra's Algorithm

Dijkstra's algorithm computes shortest paths from a source vertex by repeatedly:

- selecting the vertex with the minimum tentative distance,
- permanently fixing its distance,
- relaxing all outgoing edges from that vertex.

This process is based on a greedy strategy that relies on a crucial assumption.

Critical Assumption

All edge weights in the graph must be non-negative.

With non-negative weights, any alternative path to a vertex discovered later must be at least as long as the already computed shortest path. Hence, once a vertex is selected, its distance is guaranteed to be final.

Why Negative Edge Weights Cause Failure

When negative edge weights are present:

- a vertex may be finalized too early,
- a shorter path to that vertex may be found later through a negative edge.

This contradicts the core assumption that finalized distances cannot decrease.

Greedy Choice Breakdown

Dijkstra's algorithm is greedy in nature. It makes a local optimal choice by selecting the vertex with the smallest current distance.

With negative edges:

- a locally optimal choice may not be globally optimal,
- future relaxations can reduce previously fixed distances.

Thus, the greedy strategy becomes invalid.

Illustrative Example

Consider the graph with edges:

- $s \to a$ with weight 2,
- $s \to b$ with weight 5,

- $b \to a$ with weight -4.

Initially:

$$\text{dist}(a) = 2, \quad \text{dist}(b) = 5.$$

Dijkstra's algorithm finalizes vertex a first.

Later, when processing b, a new path to a is found with cost:

$$5 + (-4) = 1,$$

which is shorter than the finalized distance of 2. Since Dijkstra's algorithm does not revisit finalized vertices, it produces an incorrect result.

Theoretical Limitation

The failure of Dijkstra's algorithm with negative edges is not due to poor implementation but is a fundamental theoretical limitation.

Negative weights destroy the monotonicity property of shortest paths, which Dijkstra's algorithm critically depends upon.

Negative Cycles

If a graph contains a negative-weight cycle:

- no shortest path exists,
- path costs can be reduced indefinitely.

Dijkstra's algorithm cannot detect such cycles and is therefore unsuitable for these graphs as well.

Correct Alternatives

For graphs with negative edge weights, suitable algorithms include:

- **Bellman–Ford algorithm**: correctly handles negative weights and detects negative cycles.
- **Johnson's algorithm**: reweights edges to eliminate negative weights and then applies Dijkstra's algorithm.

Final Conclusion

Dijkstra's algorithm cannot be applied to graphs with negative edge weights because:

- it assumes distances never decrease after being finalized,
- negative edges can introduce shorter paths later,
- the greedy choice property no longer holds.

Hence, Dijkstra's algorithm is correct only for graphs with non-negative edge weights.