Methodology: <u>Reliable Routing in Unreliable Networks</u>

1. Network Modeling

The network is modeled as a graph consisting of nodes and edges.
Each node represents a communication entity such as a router, sensor, or device, while each edge represents a communication link between two nodes.

Unlike traditional network models, every link in this graph is associated with a success probability. This probability represents the likelihood that a packet transmitted over the link is delivered successfully.For simplicity, link failures are assumed to be independent of each other. This assumption is commonly used in probabilistic network analysis and allows tractable computation of end-to-end reliability.

2. Path Reliability Definition

For a given path between a source node and a destination node, the end-to-end success probability is defined as the product of the success probabilities of all links along that path.

If a path consists of multiple links, even a single unreliable link can significantly reduce the overall success probability. Therefore, the most reliable path is not necessarily the shortest path in terms of hop count.

3. Key Mathematical Insight

A major challenge in probabilistic routing is that path reliability is computed using multiplication of probabilities.
Standard shortest-path algorithms, such as Dijkstra's algorithm, are designed to minimize the sum of edge weights and cannot directly optimize multiplicative objectives.

To address this issue, a logarithmic transformation is applied to the link probabilities.
Each link probability ppp is converted into a weight using the following transformation:

$$\text{weight} = -\log(p)$$

This transformation converts the product of probabilities along a path into a sum of weights. As a result, maximizing the path success probability becomes equivalent to minimizing the total transformed weight.

4. Routing Algorithm

After transforming the probabilistic network into a weighted graph, Dijkstra's shortest-path algorithm is applied to compute the optimal route.

The algorithm operates as follows:

- The transformed weight is assigned to each edge.
- Dijkstra's algorithm is executed from the source node.
- The path with the minimum cumulative weight is selected.
- The final success probability is obtained by applying the exponential function to the negative of the total path weight.

This approach enables efficient computation while preserving the probabilistic nature of the routing problem.

5. Implementation Environment

The methodology was implemented using Python in a Google Colab environment.
The following libraries were used:

- **NetworkX** for graph construction and routing

- **NumPy** for numerical operations

- **Matplotlib** for visualizing network graphs and experimental results

Google Colab was chosen to ensure reproducibility and ease of experimentation without requiring local setup.

6. Simulation of Link Failures

To evaluate the robustness of the proposed routing strategy, random link failure simulations were conducted.
During each simulation run, a subset of links was randomly removed based on a predefined failure probability.

After removing failed links, the routing algorithm was re-executed to determine whether a valid path still existed and to compute the new end-to-end success probability.

This process was repeated multiple times to observe how routing performance changes under increasing levels of unreliability.

7. Performance Evaluation

The performance of the routing strategy was evaluated using the following metrics:

- End-to-end success probability
- Availability of a valid routing path after failures
- Stability of routing decisions across multiple simulations

The experimental results provide insight into how reliability-aware routing improves network performance compared to deterministic routing approaches.

8. Summary

In summary, the methodology combines probabilistic network modeling with classical shortest-path algorithms through mathematical transformation.This allows reliable routing decisions to be made efficiently while accounting for real-world link unreliability.