Routing Algorithms and Their Types - Complete Study Guide

Based on your Unit 3 materials, here's a comprehensive guide covering all routing algorithms and their classifications:

What is Routing?

Routing is the process of establishing routes that data packets must follow to reach their destination. It involves: [1]

- Creating routing tables with route information
- Using algorithms to decide the best path for packet transmission
- Considering factors like network topology, traffic load, and link costs

Key Difference:

- **Routing** = Planning (deciding where to go)
- **Forwarding** = Action (actually sending the packet)

Classification of Routing Algorithms

1. Adaptive Algorithms (Dynamic Routing)

These algorithms **change routing decisions** when network topology or traffic load changes. [1]

Types of Adaptive Algorithms:

A) Isolated Routing:

- Each node makes decisions using only **local information**
- **Disadvantage**: May send packets through congested networks
- **Examples**: Hot potato routing, backward learning

B) Centralized Routing:

- **One central node** has complete network information
- **Advantage**: Only one node maintains full network data
- **Disadvantage**: Single point of failure

- **Example**: Link State Algorithm

C) Distributed Routing:

- Nodes receive information from **neighbors** to make decisions
- **Also known as**: Decentralized algorithm
- **Advantage**: Computes least-cost paths distributedly
- **Disadvantage**: Packet delays during information updates

2. Non-Adaptive Algorithms (Static Routing)

These algorithms **do not change** routing decisions once selected. [1]

Types of Non-Adaptive Algorithms:

A) Flooding:

- Sends incoming packets to **all outgoing links** except arrival link
- **Problem**: Can create packet loops and duplicates
- **Types**: Uncontrolled, Controlled (SNCF, RPF), Selective

B) Random Walk:

- Packets sent **randomly** from node to node

3. Hybrid Algorithms

Combination of adaptive and non-adaptive approaches. [1]

- Network divided into regions using different algorithms
- **Examples**: Link-state, Distance vector

Specific Routing Algorithms

1. Shortest Path Algorithm (Dijkstra's)

Purpose: Find minimum distance between source and all other nodes [1]

Algorithm Steps:

1. Mark source distance = 0, others = infinity
2. Select unvisited node with smallest distance as current
3. For each neighbor: Calculate new_distance = current_distance + edge_weight
4. Update if new distance is smaller
5. Mark current node as visited
6. Repeat until all nodes visited

Key Formula: `new_distance = current_distance + edge_weight`

2. Distance Vector Routing (DVR)

Also Known As: Bellman-Ford Algorithm, Old ARPANET routing [1]

How It Works:

- Each router maintains **distance vector table**
- Shows shortest distance to all destinations
- Routers **periodically share** distance vectors with neighbors

Key Formula: `Dx(y) = min{Cost(x,v) + Dv(y)}`

- x = source, y = destination, v = intermediate node

Initialization:

- Distance to self = 0
- Distance to all others = infinity

3. Count-to-Infinity Problem

Definition: DVR issue where routers take excessive time to converge after link failure [1]

How It Happens:

1. Link fails between routers
2. Routers update tables (set failed link to infinity)
3. Neighboring routers feed each other **incorrect information**
4. Costs incrementally increase toward infinity

Solutions: [1]

- **Route Poisoning**: Advertise failed routes with infinity metric
- **Split Horizon**: Don't send route info back in same direction it was learned

4. Link State Routing (LSR)

Concept: Each router shares neighborhood knowledge with all other routers [1]

Three Key Features:

1. **Neighborhood Knowledge**: Broadcast directly attached link costs
2. **Flooding**: Send information to all routers except neighbors
3. **Information Sharing**: Updates only when changes occur

Two Phases:

1. **Reliable Flooding**: All nodes learn complete topology

2. **Route Calculation**: Use Dijkstra's algorithm for optimal routes

5. Path Vector Routing

Key Feature: Routers share **full path information** to destinations[1]

Purpose: Prevent loops and enable policy-based routing

Best Example: **Border Gateway Protocol (BGP)**

- Used for **inter-domain routing** between Autonomous Systems
- Maintains complete AS path to avoid loops
- Decides which networks data passes through globally

Three Phases:[1]

1. **Initiation**: Setup routing tables and path info
2. **Sharing**: Exchange path-vector messages
3. **Updating**: Modify tables based on received info

6. Hierarchical Routing

Purpose: Solve routing table size problems in large networks[1]

How It Works:

- Divide routers into **regions**
- Complete details within own region
- Basic info about other regions
- For huge networks: regions → clusters → zones → groups

Trade-off: Reduced table space but increased path length

7. Flooding Algorithm

Method: Send packet to **ALL outgoing links** except arrival link[1]

Advantages:

- Simple to implement
- Extremely robust - works even with router failures
- Always finds shortest path
- Ensures all nodes receive packet

Limitations:

- Creates duplicate packets without control
- Wasteful for single destinations

- Can clog network with unwanted traffic

Algorithm Comparison Table

Algorithm	Type	Convergence	Memory	Best Use Case
Dijkstra's	Centralized	Fast	High	Small networks
Distance Vector	Distributed	Slow	Medium	Simple networks
Link State	Centralized	Fast	High	Large networks
Path Vector	Distributed	Medium	High	Inter-domain
Flooding	Non-adaptive	Immediate	Low	Broadcasting

Key Principles

Optimality Principle[1]

If router J lies on optimal path from I to K, then path from J to K must also be optimal.

Proof: If better route existed from J to K, the I→K path would be updated, making original path sub-optimal.

Exam Preparation Tips

High-Priority Topics:

1. **Dijkstra's step-by-step calculation** - Most frequently tested
2. **DVR Bellman-Ford formula** and Count-to-Infinity problem
3. **Comparison tables** between different algorithms
4. **Path Vector vs Distance Vector** differences

Practice Problems:

1. Work through Dijkstra's algorithm on sample networks
2. Trace DVR table updates with 3-4 node networks
3. Explain Count-to-Infinity with solutions
4. Draw hierarchical routing structure diagrams

Key Formulas to Memorize:

- **Dijkstra's**: `new_distance = current_distance + edge_weight`
- **Distance Vector**: `Dx(y) = min{Cost(x,v) + Dv(y)}`
- **Optimality**: Subpaths of optimal paths are optimal

This comprehensive guide covers all routing algorithm types from your Unit 3 syllabus. Focus especially on Dijkstra's calculations and DVR concepts as these appear frequently in exams.

⁂

1. M23DE0203_3007_Unit-3.pdf
2. M23DE0203_3007_Unit-4.pdf
3. image.jpg
4. WhatsApp-Image-2025-09-15-at-18.44.37_b5e4b3c3.jpg
5. Computer-Networks_Assignment-2.xlsx