REVA UNIVERSITY
Bengaluru, India

COMPUTER NETWORKS
M23DE0203
UNIT-III

School of CSA/ MCA- 2nd Semester
Nagaraj C

www.reva.edu.in

UNIT III

Introduction to Network Layer

Layer-3

LECTURE -1

Introduction to Network Layer

- **Network layer**
- **Network Layer Function**
- **Routing**
- **Forwarding**

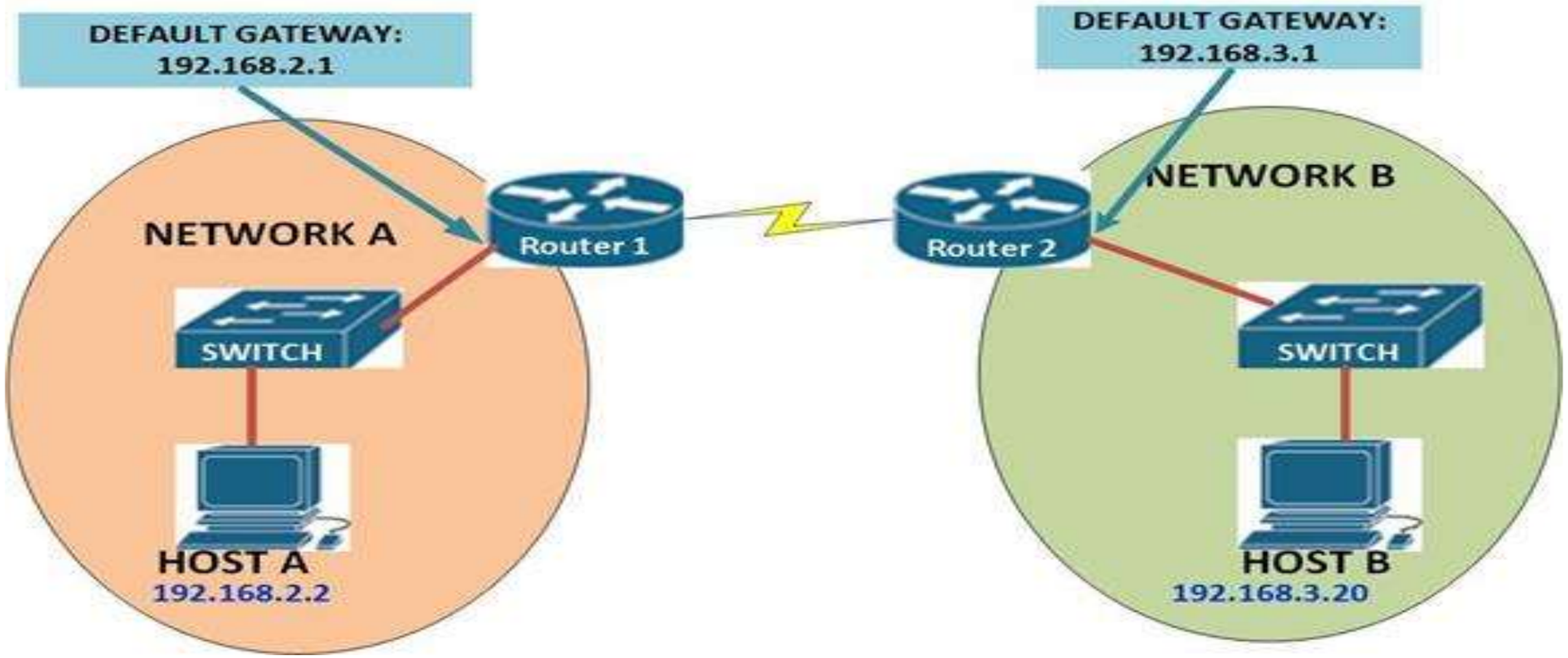
WHAT IS NETWORK LAYER?

Definition

The **network layer** is the **third layer** in the OSI (Open Systems Interconnection) model, responsible for **delivering data across different networks**. Its main job is to move packets from the **source** to the **destination**, even if they are in **different networks**.

NETWORK LAYER

Example



NETWORK LAYER

Key features:

Uses **IP addresses**

These addresses are used to **identify the source and destination** of data packets so that they can be delivered correctly across networks.

Handles **packet switching**

Packet switching means breaking data into small units called **packets**, which are sent individually through the network. These packets may take **different paths** to reach the destination, where they are **reassembled** into the original message.

Provides **connectionless communication** (like with IP)

This means the network layer can **send data without setting up a dedicated path or session** between sender and receiver. Each packet is treated independently and may travel a different route. **IP is connectionless**, meaning it just sends packets without checking if the destination is ready or available. That's why higher layers like TCP handle things like reliability and ordering.

NETWORK LAYER

The main **functions** of the network layer include:

- **Logical addressing**: Assigns IP addresses to devices so data can reach the correct destination.
- **Routing**: Finds the best path for data to travel across multiple networks.
- **Packet forwarding**: Sends the data packets toward the destination.
- **Fragmentation and reassembly**: Breaks down large packets if needed and reassembles them at the destination.

FORWARDING

Definition

Forwarding is the act of **moving packets from an incoming interface to the appropriate outgoing interface** on a router or switch based on the routing table.

Key difference from routing:

- **Routing** is the planning (deciding where to go)
- **Forwarding** is the action (actually sending the packet)

ROUTING

Definition

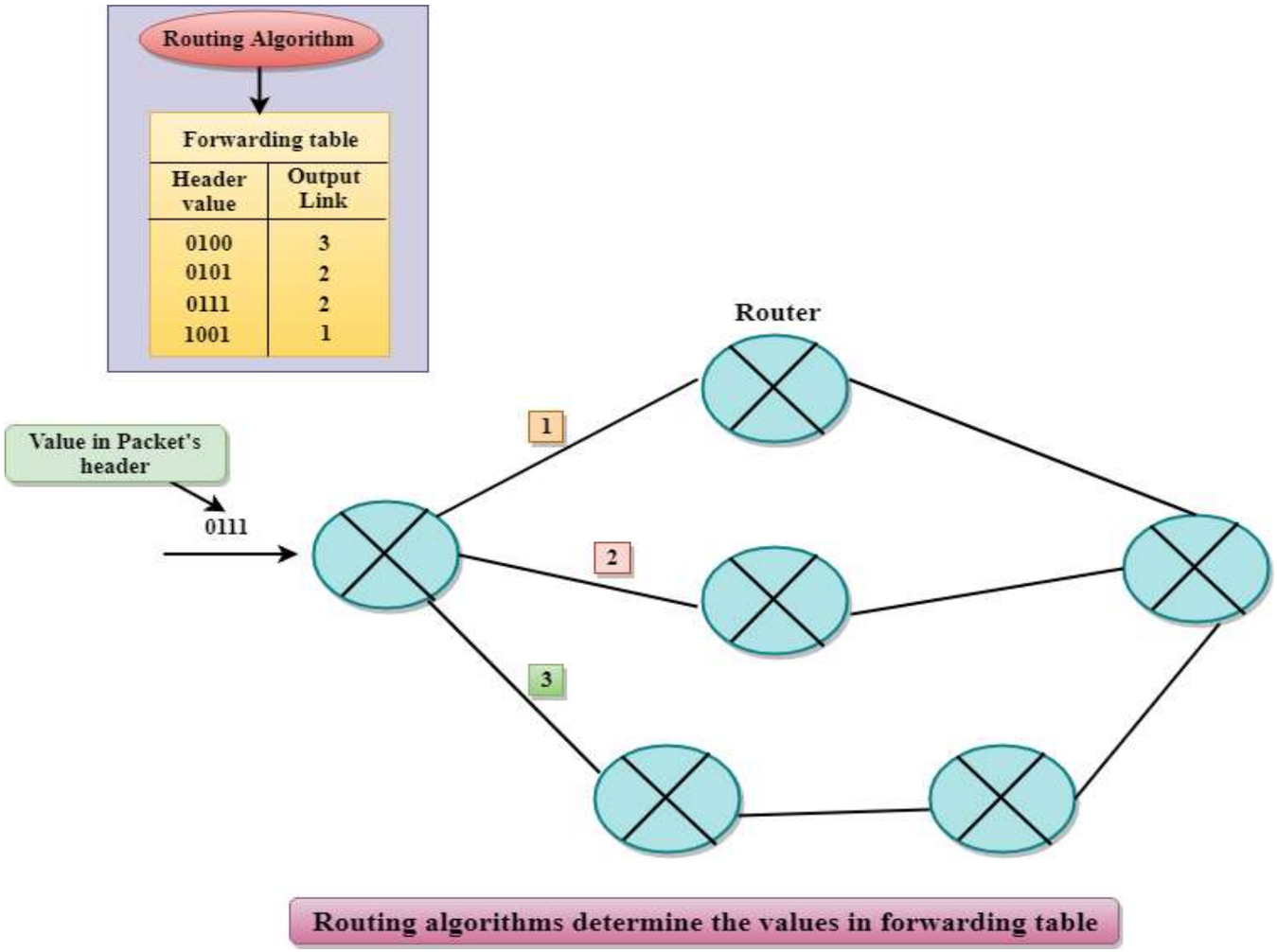
Routing is the process of determining the best path for data to travel from the source to the destination. This is done using **routing algorithms and protocols** (like OSPF, BGP, RIP).

Characteristics:

- Involves building and maintaining **routing tables**
- Takes into account factors like **network topology**, **traffic load**, and **link cost**
- Performed by **routers**

ROUTING & FORWARDING

Example



ROUTING & FORWARDING

Routing is the process of determining the best path for data to travel from the source to the destination across interconnected networks. It involves building and maintaining routing tables using various algorithms and protocols like OSPF or BGP. This is more of a decision-making process that happens periodically.

forwarding is the actual movement of data packets based on the routing table's decisions. It involves looking up the destination address in the forwarding table and sending the packet out through the appropriate interface. While routing is concerned with path selection, forwarding is focused on the fast and efficient delivery of each packet along that path.

SUMMARY

Lecture-1 : Introduction to Network Layer

Q.1 What is Network layer?

Q.2 What are the features of network layer?

Q.3 What are the function of network layer?

Q.4 What is forwarding?

Q.5 What is routing?

Q.6 What is/are the differences between forwarding and routing?

ACTIVITY

Lecture-1 : Introduction to Network Layer

Reading :

Forwarding Techniques:

http://www.brainkart.com/article/Forwarding-Techniques_13480/

NETWORK LAYER DESIGN ISSUES

1. Services to the Transport Layer

- The network layer provides services to the transport layer, which can be either **connectionless** or **connection-oriented**.
- **Connectionless** (like IP): Sends packets without setting up a path. There's no guarantee of delivery.
- **Connection-oriented** (like virtual circuits): A path is established first, and all packets follow the same route.

Example:

- **Connectionless**: Sending multiple emails to someone; each one goes independently.
- **Connection-oriented**: Making a phone call; a connection is established before talking.

NETWORK LAYER DESIGN ISSUES

2. Packetizing / Packet Format

- Data must be broken into **packets**, each with a **header** containing information like source/destination address, sequence number, etc.
- **Design Challenge**: How should packets be formatted for efficiency and ease of routing?

Example:

- To make a packet efficient and easy to process, the **structure** (or format) must be **simple, consistent, and optimized** for both routers and receivers. Here's how it should be designed:
- The packet should be divided into two parts:
  - **Header**: Contains control information (source/destination IP, packet length, etc.)
  - **Payload**: Contains the actual data being sent.

NETWORK LAYER DESIGN ISSUES

3. Addressing

Every device on a network must have a unique **IP address** so data knows where to go.

- IPv4: 192.168.1.1
- IPv6: 2001:0db8:85a3::8a2e:0370:7334

Example:

- Like **sending a letter** to a friend's house. You need their **home address** (IP address) to deliver it.

NETWORK LAYER DESIGN ISSUES

4. Routing

The network layer chooses the **best path** for a packet to travel through routers from source to destination.

- Uses routing algorithms and protocols like OSPF(Open Shortest Path First) or BGP(Border Gateway Protocol).

Example:

- Like **choosing the fastest or safest route** on Google Maps. If one road is blocked, GPS suggests another route.

NETWORK LAYER DESIGN ISSUES

5. Error Handling

- Packets might get **lost, duplicated, or corrupted**. The network layer may detect errors and notify the sender or drop the packet.
- Some protocols do basic error detection (e.g., checksum in IP header).

Example:

- Like a courier realizing that a package label is unreadable and either returns it to the sender or discards it.

NETWORK LAYER DESIGN ISSUES

6. Congestion Control

- Too many packets in the network can **overload routers and links**, causing delays or loss.
- The network layer should manage traffic flow to prevent this.

Example:

- Like **traffic congestion** during rush hour — too many cars (packets) on the road (network).

NETWORK LAYER DESIGN ISSUES

7. Internetworking

- The network layer must support communication between **different types of networks** (e.g., Wi-Fi, Ethernet, 4G, etc.).
- It hides the differences between networks and ensures **universal communication**.

Example:

- Like sending a parcel that passes through truck, plane, and bike — different transport modes, but the parcel still arrives.

NETWORK LAYER DESIGN ISSUES

8. Fragmentation and Reassembly

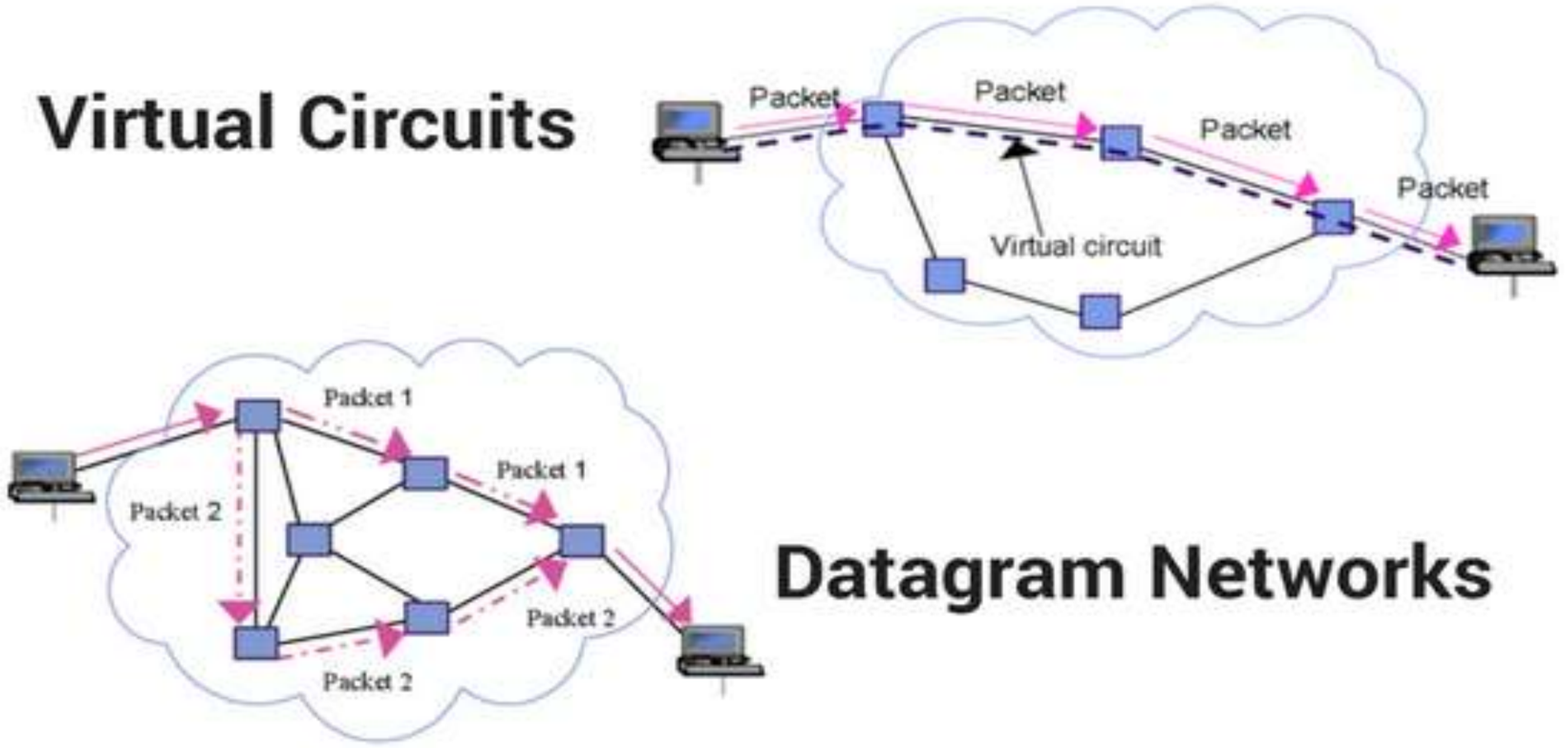
- Some networks have a **Maximum Transmission Unit (MTU)**. If a packet is too large, it must be **fragmented** into smaller pieces and later **reassembled**.
- IP handles this using fragmentation flags and offsets in headers.

Example:

- Like trying to move a big piece of furniture through a small door. You must **disassemble it**, move the parts, and **reassemble** it inside.

TYPES OF PACKET SWITCHING NETWORK

Example



NETWORK LAYER

Virtual Circuit - Services

1. Guaranteed delivery
2. Guaranteed delivery with bounded delay
3. In-order packet delivery
4. Guaranteed minimal bandwidth
5. Guaranteed maximum jitter
6. Security service

NETWORK LAYER

Datagram- Services

Provides **Best effort services (Service model)**

1. No Guaranteed delivery
2. No Guaranteed delivery with bounded delay
3. No In-order packet delivery
4. No Guaranteed minimal bandwidth
5. No Guaranteed maximum jitter

STORE – AND – FORWARD PACKET SWITCHING

In networks, store – and – forward packet switching is a technique where the data packets are stored in each intermediate node, before they are forwarded to the next node.

The intermediate node checks whether the packet is error–free before transmitting, thus ensuring integrity of the data packets.

In general, the network layer operates in an environment that uses store and forward packet switching.

STORE – AND – FORWARD PACKET SWITCHING-

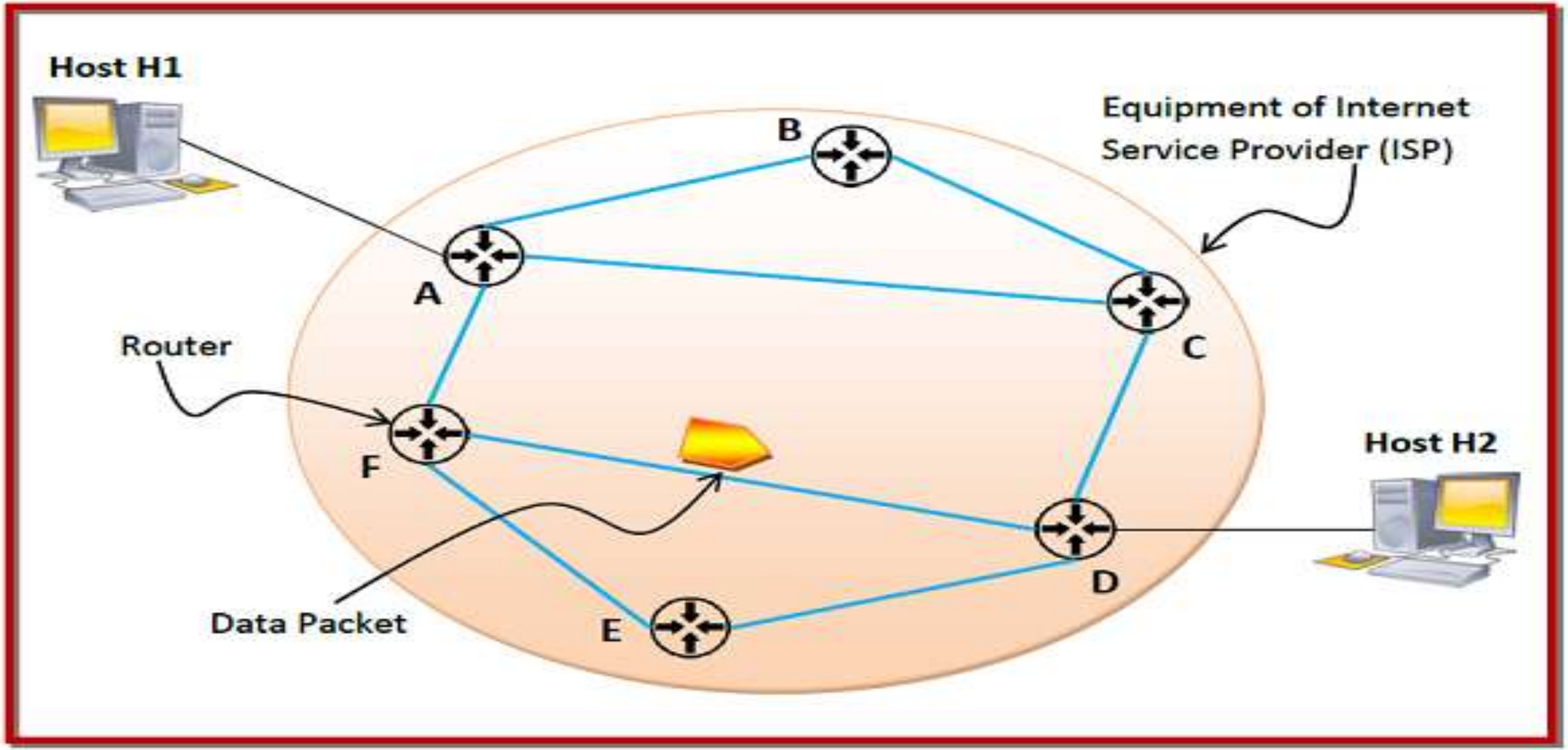
WORKING PRINCIPLE

The node which has a packet to send, delivers it to the nearest node, i.e. router. The packet is stored in the router until it has fully arrived and its checksum is verified for error detection.

Once, this is done, the packet is transmitted to the next router.

The same process is continued in each router until the packet reaches its destination.

STORE – AND – FORWARD PACKET SWITCHING-
MECHANISM



STORE – AND – FORWARD PACKET SWITCHING- MECHANISM

In the above diagram, we can see that the Internet Service Provider (ISP) has six routers (A to F) connected by transmission lines shown in blue lines.

There are two hosts, host H1 is connected to router A, while host H2 is connected to router D.

Suppose that H1 wants to send a data packet to H2. H1 sends the packet to router A. The packet is stored in router A until it has arrived fully. Router A verifies the checksum using CRC (cyclic redundancy check) code.

If there is a CRC error, the packet is discarded, otherwise it is transmitted to the next hop, here router F.

The same process is followed by router F which then transmits the packet to router D. Finally router D delivers the packet to host H2.

CONNECTION ORIENTED AND CONNECTIONLESS SERVICES

In computer networks, communication between devices occurs using two types of services: **connection-oriented** and **connectionless**.

These services define how data is transferred between a source and a destination.

Connection-oriented services establish a dedicated connection before data transfer, ensuring reliability.

In contrast, connectionless services do not establish a connection, sending data without acknowledgment or error correction.

WHAT IS A CONNECTION-ORIENTED SERVICE?

Connection-oriented services involve setting up a dedicated path between the source and destination before data transfer begins.

These services ensure that data is delivered in the correct sequence and without errors. In a connection-oriented service, the Handshake method is used to establish the connection between sender and receiver.

Before data transmission starts, connection-oriented services create a dedicated communication channel between the sender and the recipient. A

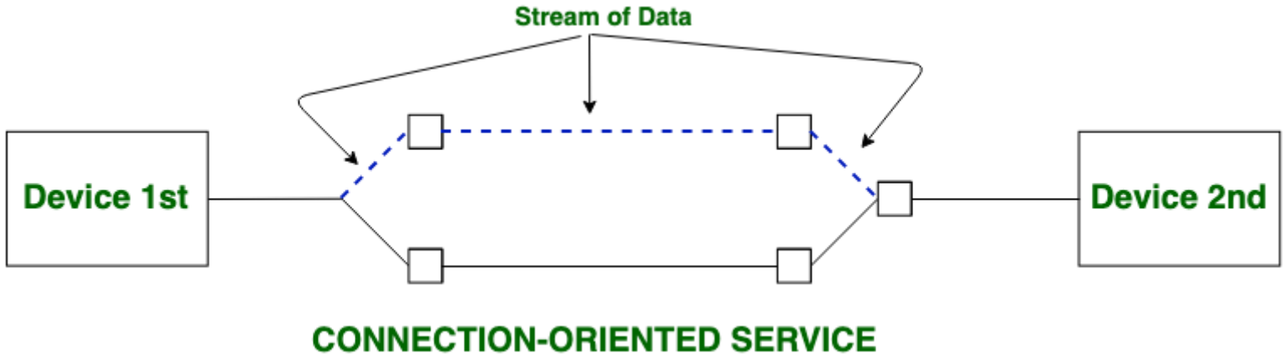
s the connection is kept open until all data is successfully transferred, this guarantees dependable data delivery.

One example is TCP (Transmission Control Protocol), which ensures error-free and accurate data packet delivery.

CONNECTION-ORIENTED SERVICE

Examples of Connection-Oriented Services

- TCP (Transmission Control Protocol) in the TCP/IP suite.
- Telephone calls in traditional telecommunication systems.



CONNECTION-ORIENTED SERVICE

Key Features of Connection-Oriented Services

- **Dedicated Connection** : A logical or physical connection is established before data transfer.
- **Reliable Transmission** : Data is transmitted with error checking, acknowledgments, and retransmissions in case of errors.
- **Sequencing** : Data packets arrive at the destination in the correct order.

Advantages of Connection-Oriented Services

- **Reliable Data Transfer** : Ensures that all data reaches its destination without errors.
- **Data Sequencing** : Packets are delivered in the correct order.
- **Error Correction** : Mechanisms are in place to detect and correct errors during transmission.
- **Guaranteed Delivery** : Retransmissions occur if data is lost.

CONNECTION-ORIENTED SERVICE

Disadvantages of Connection-Oriented Services

- **Higher Latency** : Establishing a connection adds latency before data transfer begins.
- **More Overhead** : Requires more resources for maintaining the connection, acknowledgments, and retransmissions.
- **Less Efficient for Small Transfers** : For short messages, the overhead of connection setup can outweigh the benefits.

WHAT IS CONNECTION-LESS SERVICE?

Connectionless services send data without establishing a dedicated connection between the source and destination.

Each data packet is treated independently, and there is no guarantee of delivery or sequencing.

Connection-less Service does not give a guarantee of reliability. In this, Packets do not follow the same path to reach their destination.

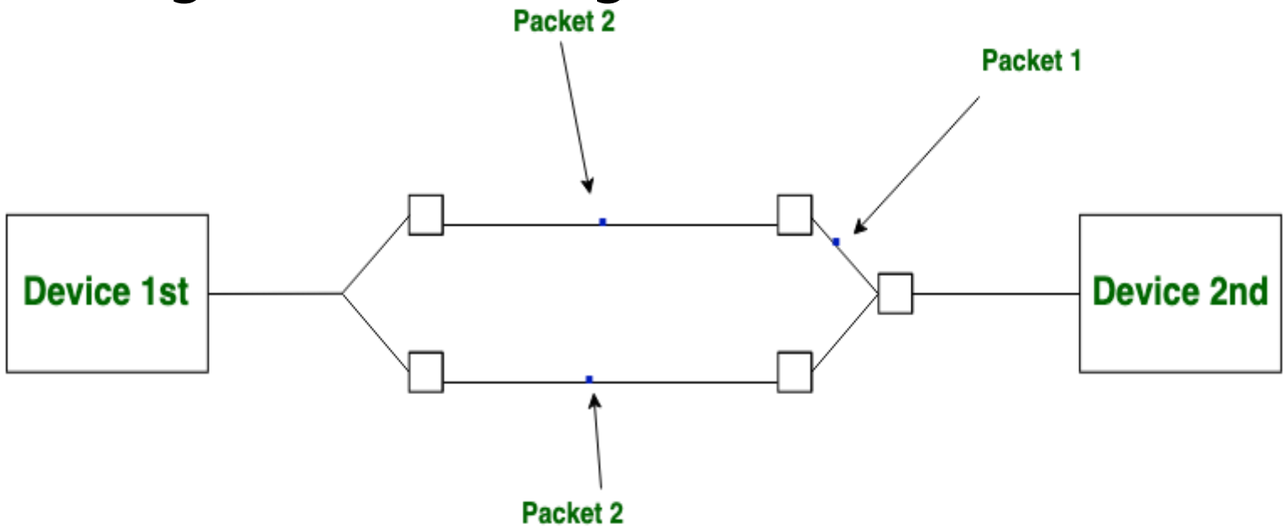
Connectionless Services deliver individual data packets without first making a connection. Since each packet is sent separately, delivery, order, and mistake correction cannot be guaranteed.

As a result, the service is quicker but less dependable. UDP (User Datagram Protocol) is one example, which is frequently used for streaming where dependability is not as important as speed.

WHAT IS CONNECTION-LESS SERVICE?

Examples of Connectionless Services

- UDP (User Datagram Protocol) in the TCP/IP suite.
- Postal services (analogous to sending letters without confirmation of receipt).



CONNECTIONIESS SERVICE

CONNECTION-LESS SERVICE

Key Features of Connectionless Services

- **No Connection Setup** : Data is sent directly without establishing a prior connection.
- **Independent Packets** : Each packet is treated individually and may take different routes to the destination.
- **Faster Transmission** : No time is spent establishing or tearing down a connection.
- **Unreliable** : No acknowledgment, retransmission, or error correction is performed.

Advantages of Connectionless Services

- **Low Latency** : Data is transmitted immediately without waiting for a connection to be established.
- **Efficient for Small Transfers** : Ideal for small, time-sensitive messages like DNS lookups or VoIP.
- **Scalable** : Suitable for systems with many simultaneous users, as no connection needs to be maintained.

CONNECTION-LESS SERVICE

Disadvantages of Connectionless Services

- **Unreliable** : Data packets may be lost, duplicated, or arrive out of order.
- **No Error Handling** : No built-in mechanisms for retransmissions or error correction.
- **Unsuitable for Large Transfers** : Not ideal for applications requiring reliable and ordered delivery.

CONNECTION-ORIENTED AND CONNECTIONLESS SERVICES

Connection-oriented Service	Connection-less Service
Connection-oriented service is related to the telephone system.	Connection-less service is related to the postal system.
Connection-oriented service is preferred by long and steady communication.	Connection-less Service is preferred by bursty communication.
Connection-oriented Service is necessary.	Connection-less Service is not compulsory.
Connection-oriented Service is feasible.	Connection-less Service is not feasible.
In connection-oriented Service, Congestion is not possible.	In connection-less Service, Congestion is possible.
Connection-oriented Service gives the guarantee of reliability.	Connection-less Service does not give a guarantee of reliability.
Includes error detection, correction, and retransmission.	No error handling; errors are not corrected.
In connection-oriented Service, Packets follow the same route.	In connection-less Service, Packets do not follow the same route.
Ensures data is delivered in the correct order.	Data may arrive out of order or not at all.
Less scalable due to the need for maintaining connections.	Highly scalable for large networks with many users.
Higher overhead due to connection setup and maintenance.	Lower overhead as no connection is required.
Connection-oriented services require a bandwidth of a high range.	Connection-less Service requires a bandwidth of low range.
Ex: TCP (Transmission Control Protocol)	Ex: UDP (User Datagram Protocol)
Connection-oriented requires authentication.	Connection-less Service does not require authentication.

ROUTING ALGORITHMS

Routing is the process of establishing the routes that data packets must follow to reach the destination.

In this process, a routing table is created which contains information regarding routes that data packets follow.

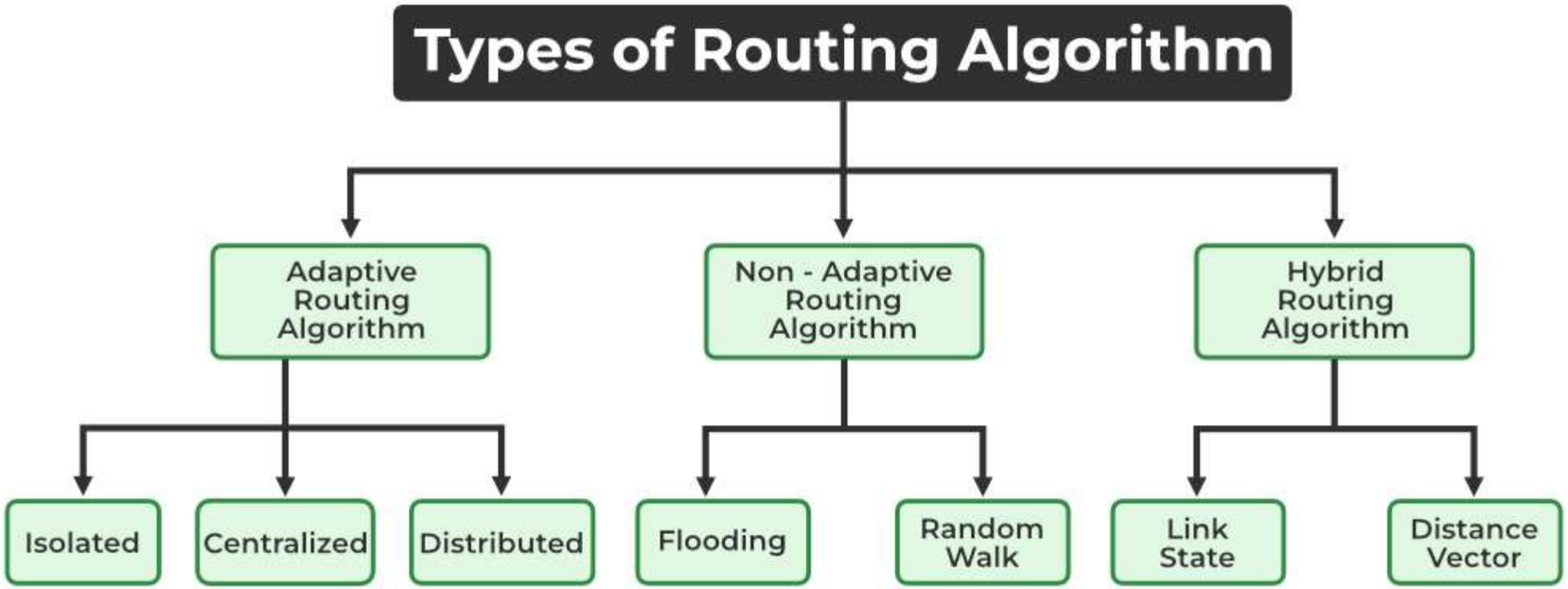
Various routing algorithms are used for the purpose of deciding which route an incoming data packet needs to be transmitted on to reach the destination efficiently.

Classification of Routing Algorithms

The routing algorithms can be classified as follows:

- Adaptive Algorithms
- Non-Adaptive Algorithms
- Hybrid Algorithms

ROUTING ALGORITHMS



ROUTING ALGORITHMS

1. Adaptive Algorithms

- These are the algorithms that change their routing decisions whenever network topology or traffic load changes. The changes in routing decisions are reflected in the topology as well as the traffic of the network. Also known as dynamic routing.

Further, these are classified as follows:

- **Isolated:** In this method each, node makes its routing decisions using the information it has without seeking information from other nodes. The sending nodes don't have information about the status of a particular link. The disadvantage is that packets may be sent through a congested network which may result in delay. Examples: Hot potato routing, and backward learning.

ROUTING ALGORITHMS

- **Centralized:** In this method, a centralized node has entire information about the network and makes all the routing decisions. The advantage of this is only one node is required to keep the information of the entire network and the disadvantage is that if the central node goes down the entire network is done. The link state algorithm is referred to as a centralized algorithm since it is aware of the cost of each link in the network.
- **Distributed:** In this method, the node receives information from its neighbors and then takes the decision about routing the packets. A disadvantage is that the packet may be delayed if there is a change in between intervals in which it receives information and sends packets. It is also known as a decentralized algorithm as it computes the least-cost path between source and destination.

ROUTING ALGORITHMS

2. Non-Adaptive Algorithms

- These are the algorithms that do not change their routing decisions once they have been selected. This is also known as static routing.

Further, these are classified as follows:

- **Flooding:** This adapts the technique in which every incoming packet is sent on every outgoing line except from which it arrived. One problem with this is that packets may go in a loop and as a result of which a node may receive duplicate packets.
- **Random walk:** In this method, packets are sent host by host or node by node to one of its neighbors randomly.

ROUTING ALGORITHMS

3. Hybrid Algorithms

- As the name suggests, these algorithms are a combination of both adaptive and non-adaptive algorithms. In this approach, the network is divided into several regions, and each region uses a different algorithm.
  Further, these are classified as follows:
- Link-state: In this method, each router creates a detailed and complete map of the network which is then shared with all other routers. This allows for more accurate and efficient routing decisions to be made.
- Distance vector: In this method, each router maintains a table that contains information about the distance and direction to every other node in the network. This table is then shared with other routers in the network. The disadvantage of this method is that it may lead to routing loops.

ROUTING ALGORITHMS-OPTIMALITY PRINCIPLE

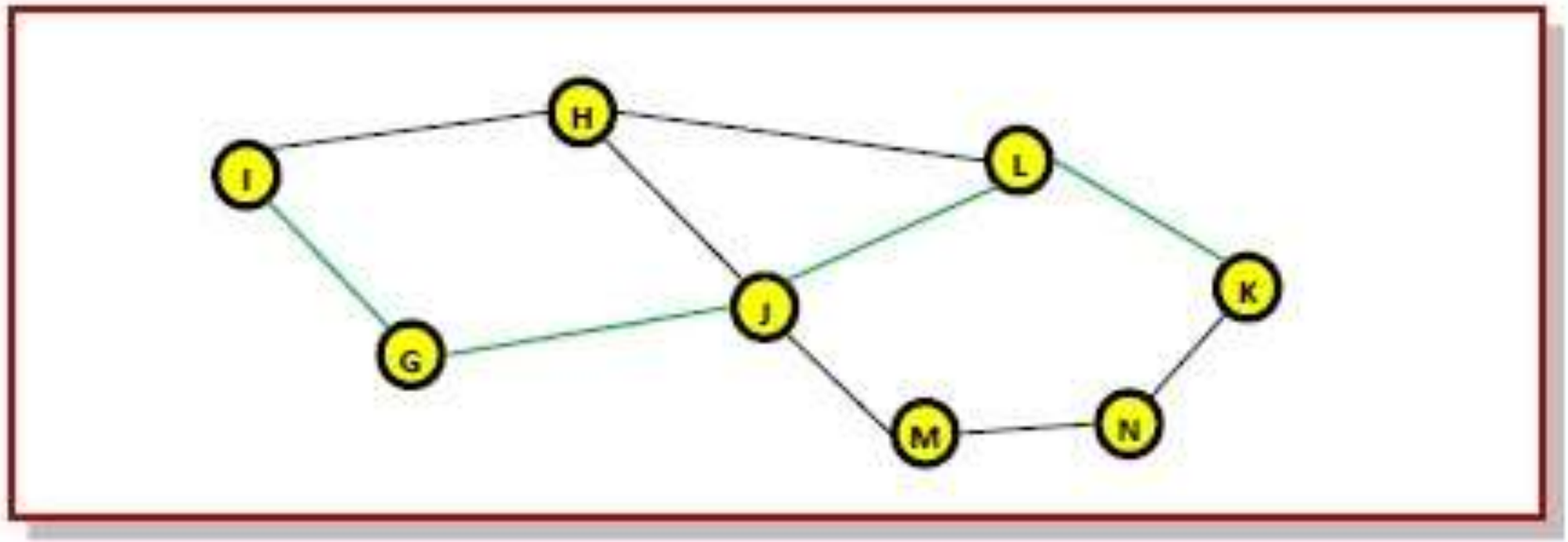
Optimality Principle

- The purpose of a routing algorithm at a router is to decide which output line an incoming packet should go. The optimal path from a particular router to another may be the least cost path, the least distance path, the least time path, the least hops path or a combination of any of the above.
- The optimality principle can be logically proved as follows
- If a better route could be found between router J and router K, the path from router I to router K via J would be updated via this route. Thus, the optimal path from J to K will again lie on the optimal path from I to K.

ROUTING ALGORITHMS-OPTIMALITY PRINCIPLE

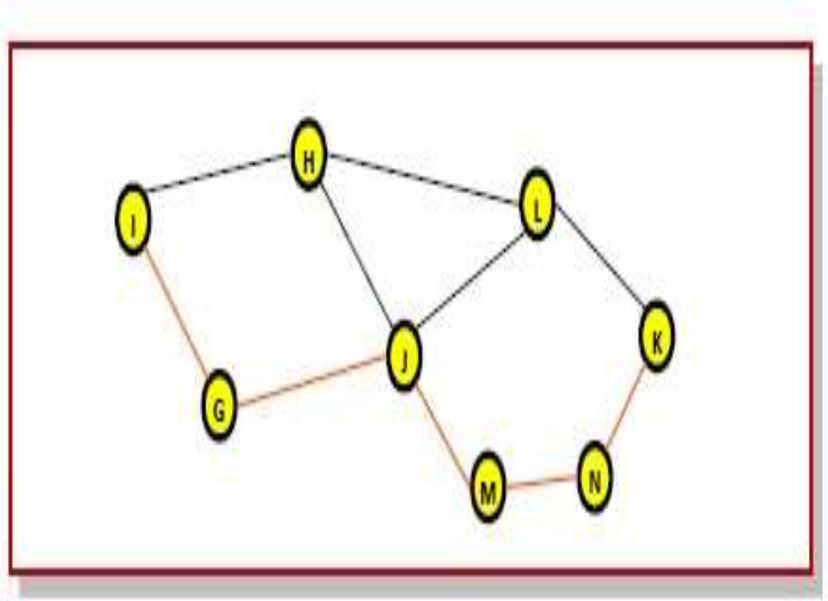
Example

- Consider a network of routers, {G, H, I, J, K, L, M, N} as shown in the figure. Let the optimal route from I to K be as shown via the green path, i.e. via the route I-G-J-L-K. According to the optimality principle, the optimal path from J to K with be along the same route, i.e. J-L-K.



ROUTING ALGORITHMS-OPTIMALITY PRINCIPLE

- Now, suppose we find a better route from J to K is found, say along J-M-N-K. Consequently, we will also need to update the optimal route from I to K as I-GJ- M-N-K, since the previous route ceases to be optimal in this situation. This new optimal path is shown line orange lines in the following figure



ROUTING ALGORITHMS-SHORTEST PATH ALGORITHM

In between sending and receiving data packets from the sender to the receiver, it will go through many routers and subnets. So as a part of increasing the efficiency in routing the data packets and decreasing the traffic, we must find the shortest path.

What is Shortest Path Routing?

- It refers to the algorithms that help to find the shortest path between a sender and receiver for routing the data packets through the network in terms of shortest distance, minimum cost, and minimum time.
- It is mainly for building a graph or subnet containing routers as nodes and edges as communication lines connecting the nodes.
- Hop count is one of the parameters that is used to measure the distance.
- **Hop count**: It is the number that indicates how many routers are covered. If the hop count is 6, there are 6 routers/nodes and the edges connecting them.

SHORTEST PATH-DIJKSTRA'S ALGORITHM

- The Dijkstra's Algorithm is used to find the minimum distance between a node and all other nodes in a given graph. Here we can consider node as a router and graph as a network. It uses weight of edge .ie, distance between the nodes to find a minimum distance route.

Algorithm:

1: Mark the source node current distance as 0 and all others as infinity.

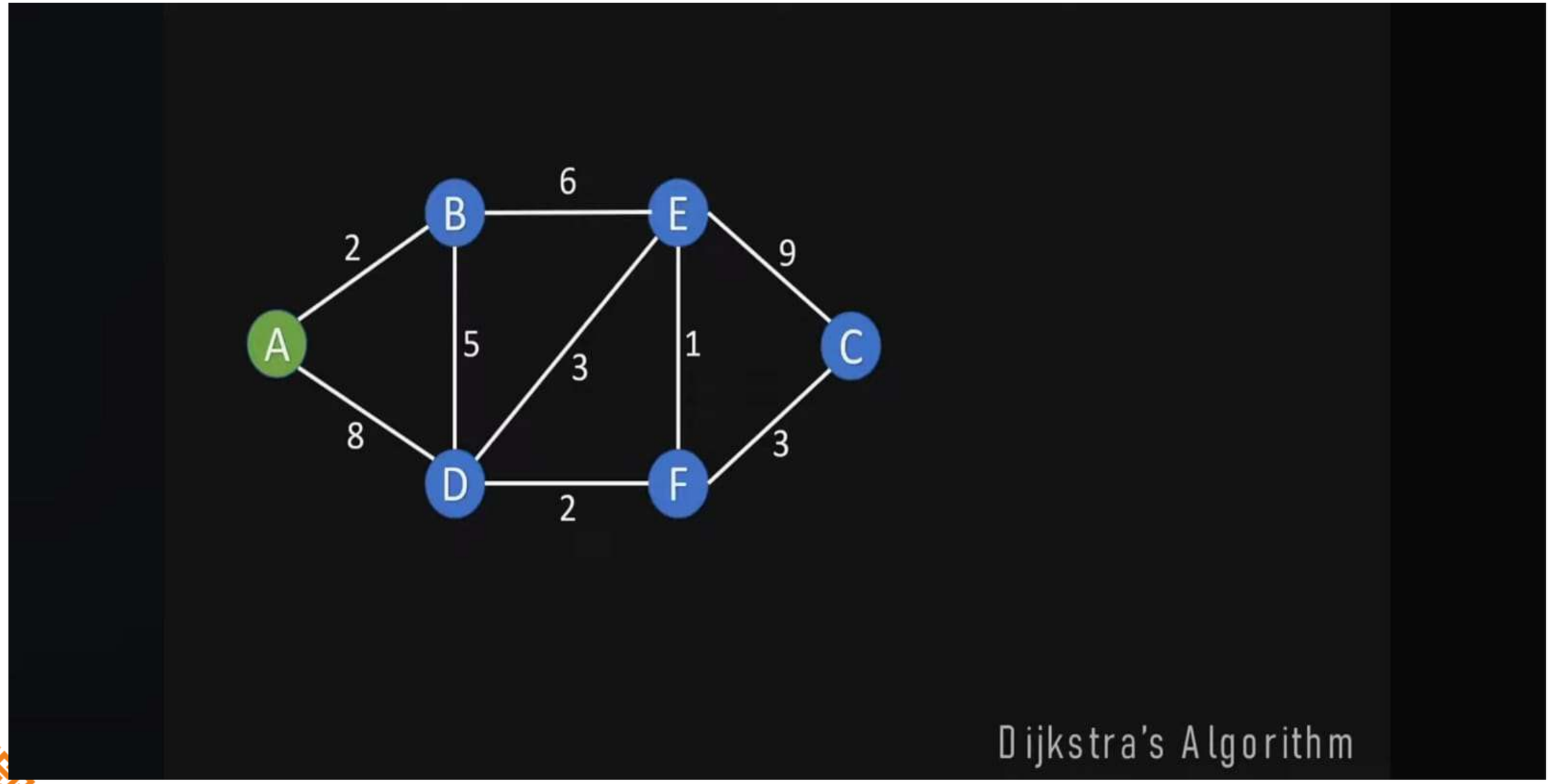
2: Set the node with the smallest current distance among the non-visited nodes as the current node.

3: For each neighbor, N, of the current node:Calculate the potential new distance by adding the current distance of the current node with the weight of the edge connecting the current node to N.If the potential new distance is smaller than the current distance of node N, update N's current distance with the new distance.

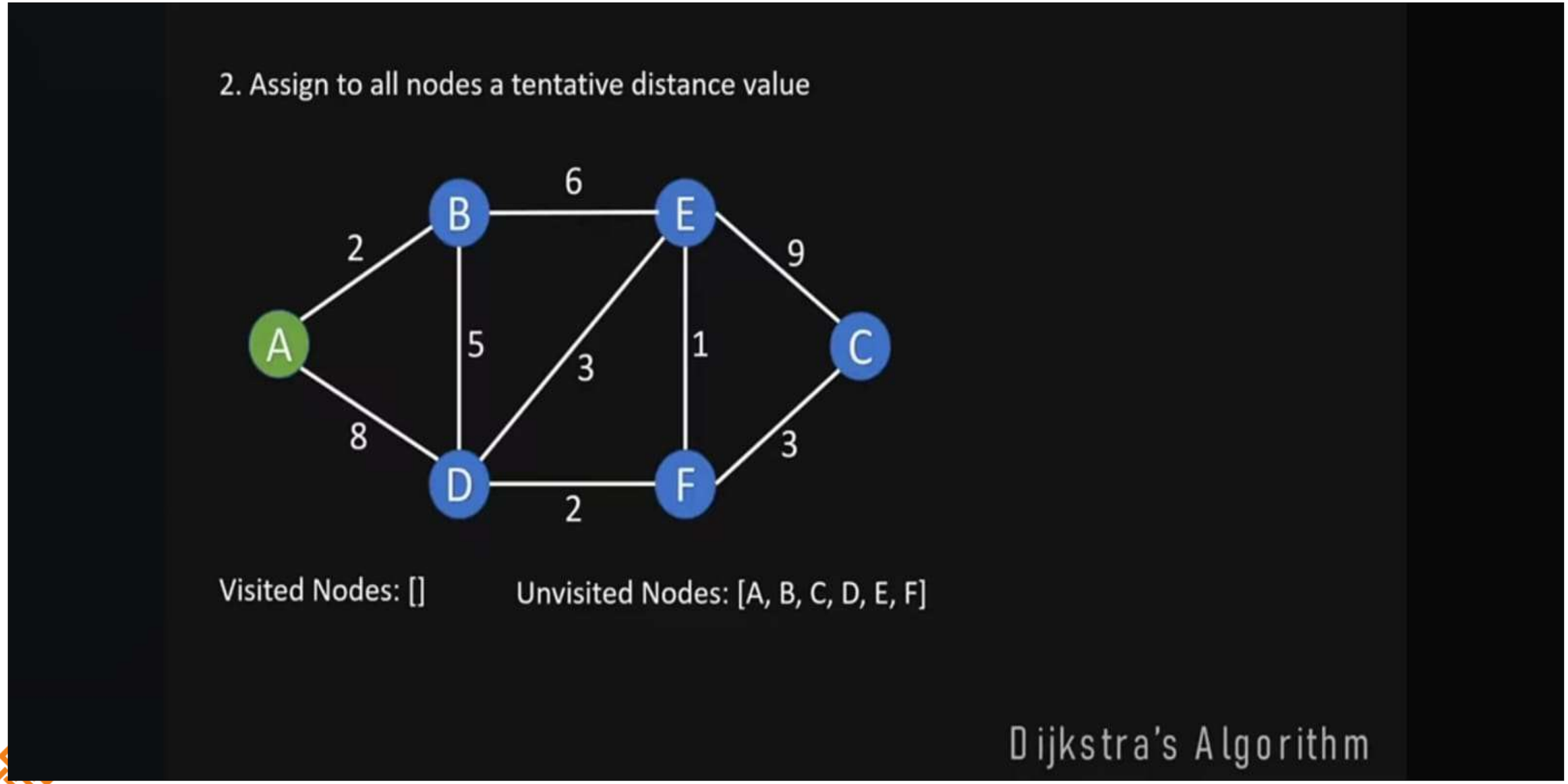
4: Make the current node as visited node.

5: If we find any unvisited node, go to step 2 to find the next node which has the smallest current distance and continue this process.

SHORTEST PATH-DIJKSTRA'S ALGORITHM

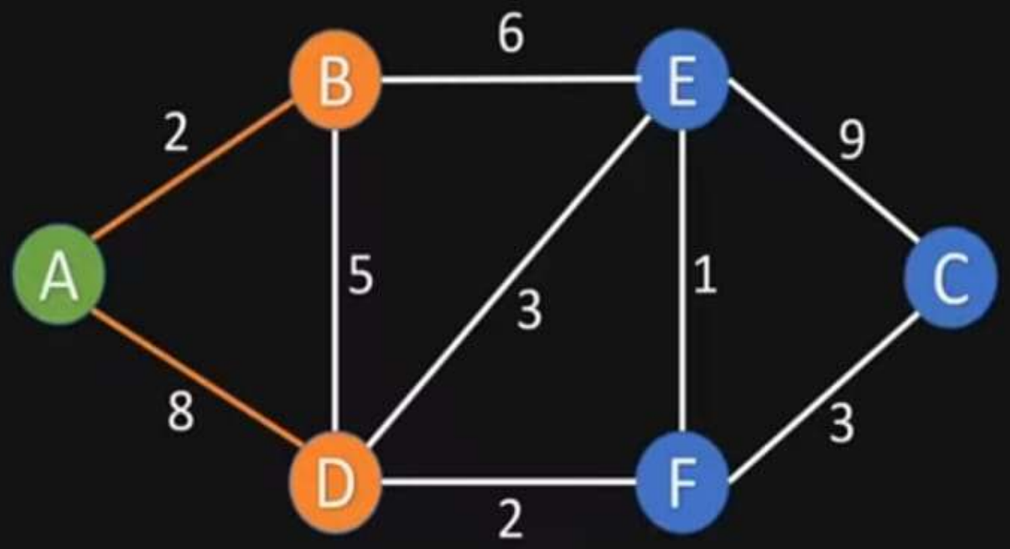


SHORTEST PATH-DIJKSTRA'S ALGORITHM



SHORTEST PATH-DIJKSTRA'S ALGORITHM

3. For the current node calculate the distance to all unvisited neighbours
3.1. Update shortest distance, if new distance is shorter than old distance



Node	Shortest Distance	Previous Node
A	0	
B	2	A
C	∞	
D	8	A
E	∞	
F	∞	

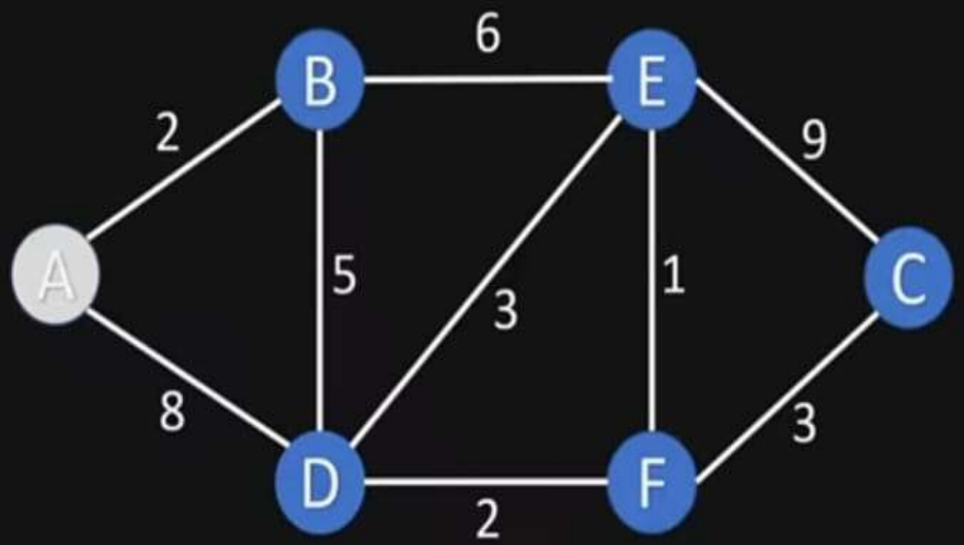
Visited Nodes: []

Unvisited Nodes: [A, B, C, D, E, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

4. Mark current node as visited



Node	Shortest Distance	Previous Node
A	0	
B	2	A
C	∞	
D	8	A
E	∞	
F	∞	

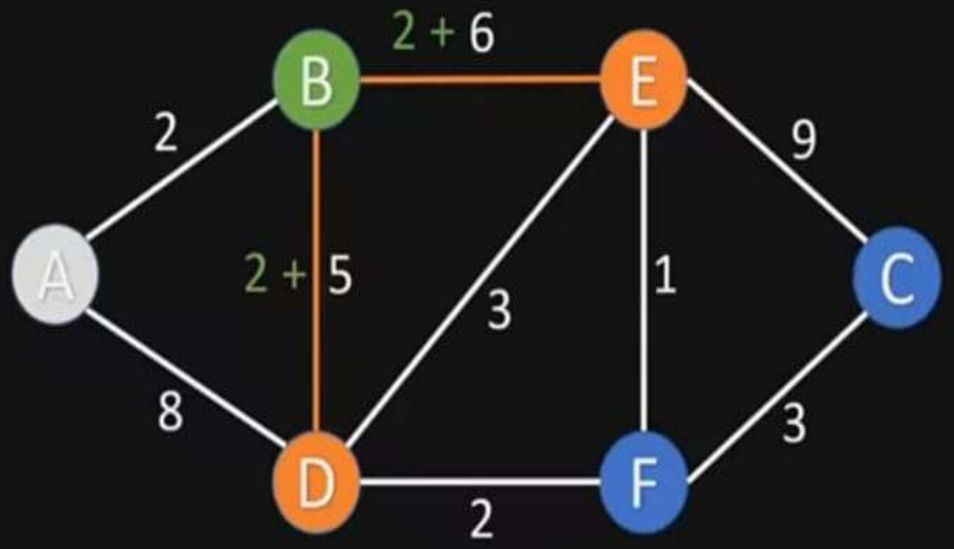
Visited Nodes: [A]

Unvisited Nodes: [B, C, D, E, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

3. For the current node calculate the distance to all unvisited neighbours
3.1. Update shortest distance, if new distance is shorter than old distance



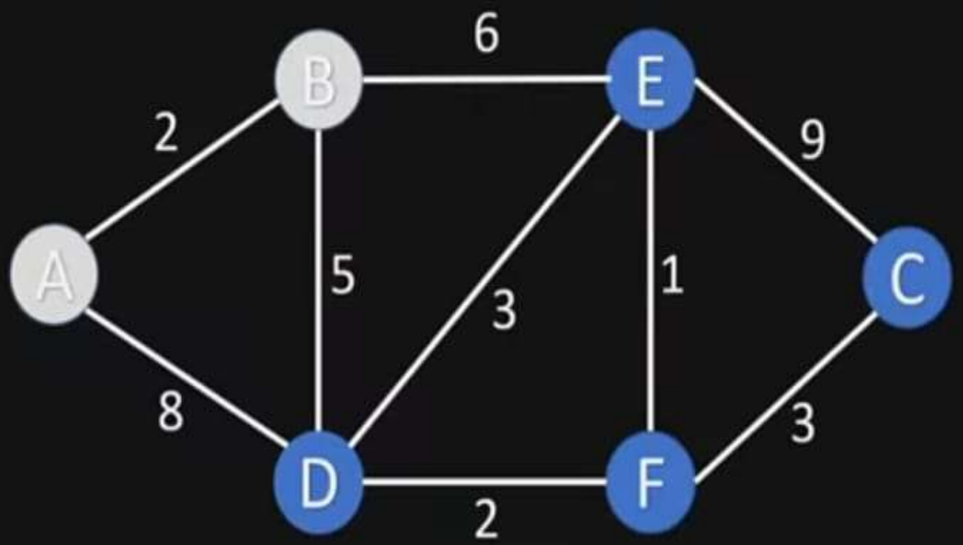
Node	Shortest Distance	Previous Node
A	0	
B	2	A
C	∞	
D	7	B
E	8	B
F	∞	

Visited Nodes: [A] Unvisited Nodes: [B, C, D, E, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

4. Mark current node as visited



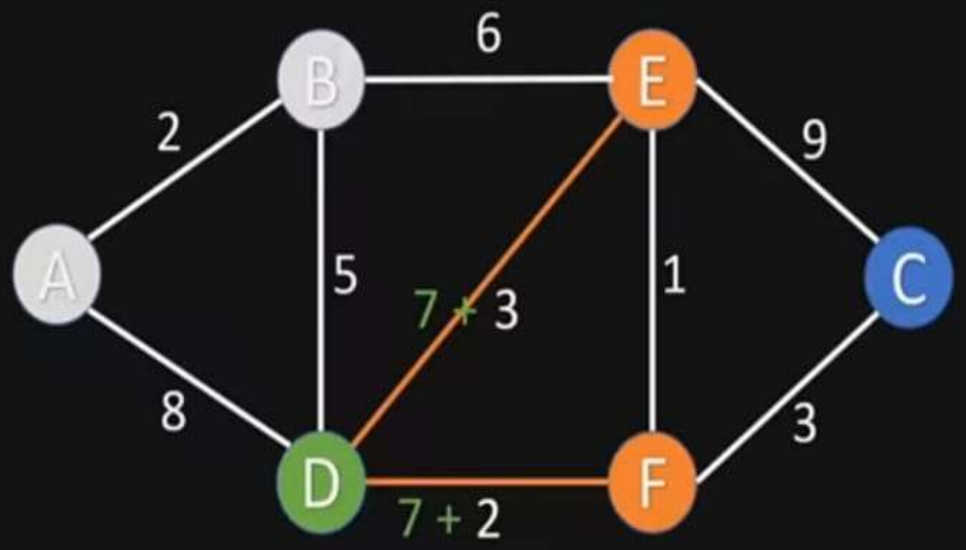
Node	Shortest Distance	Previous Node
A	0	
B	2	A
C	∞	
D	7	B
E	8	B
F	∞	

Visited Nodes: [A, B] Unvisited Nodes: [C, D, E, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

3. For the current node calculate the distance to all unvisited neighbours
3.1. Update shortest distance, if new distance is shorter than old distance



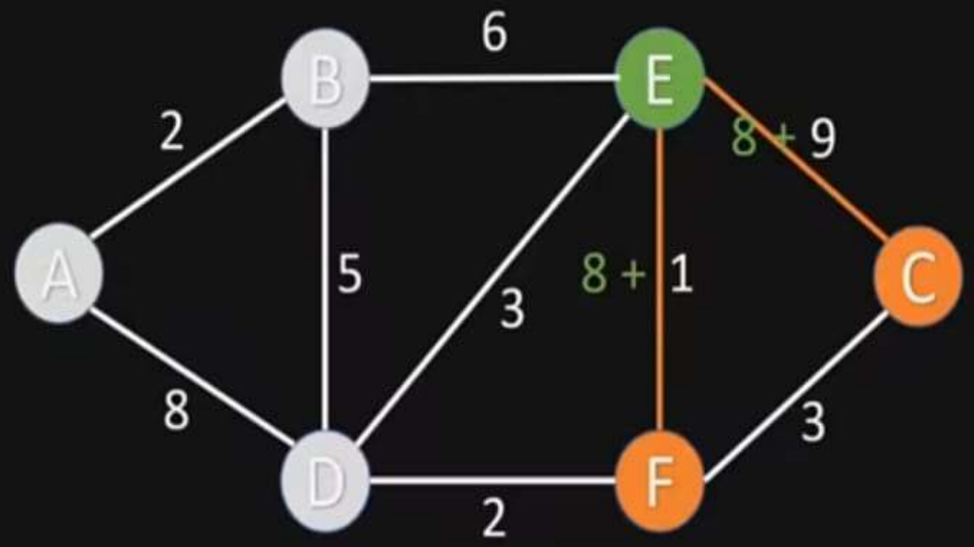
Node	Shortest Distance	Previous Node
A	0	
B	2	A
C	∞	
D	7	B
E	8	B
F	9	D

Visited Nodes: [A, B] Unvisited Nodes: [C, D, E, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

3. For the current node calculate the distance to all unvisited neighbours
3.1. Update shortest distance, if new distance is shorter than old distance



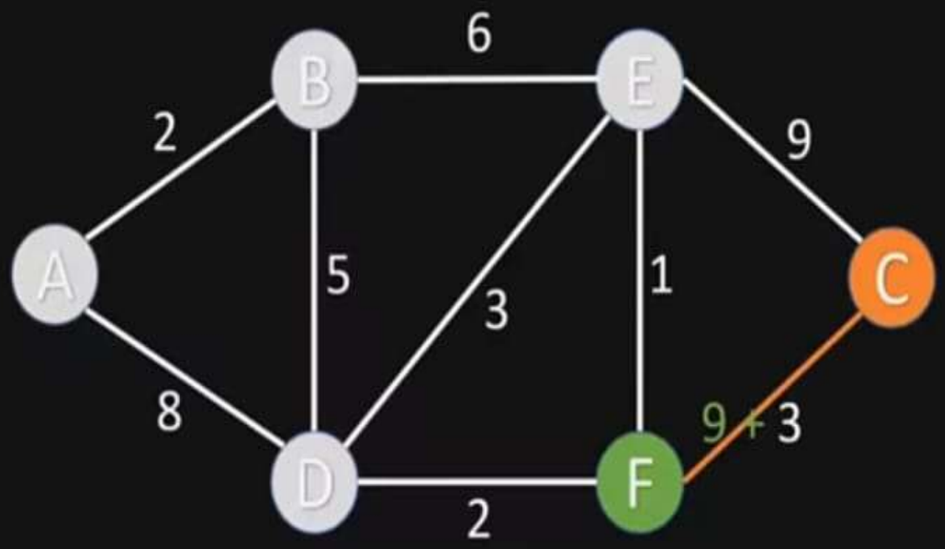
Node	Shortest Distance	Previous Node
A	0	
B	**2**	A
C	**17**	E
D	**7**	B
E	**8**	B
F	**9**	D

Visited Nodes: [A, B, D] Unvisited Nodes: [C, E, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

3. For the current node calculate the distance to all unvisited neighbours



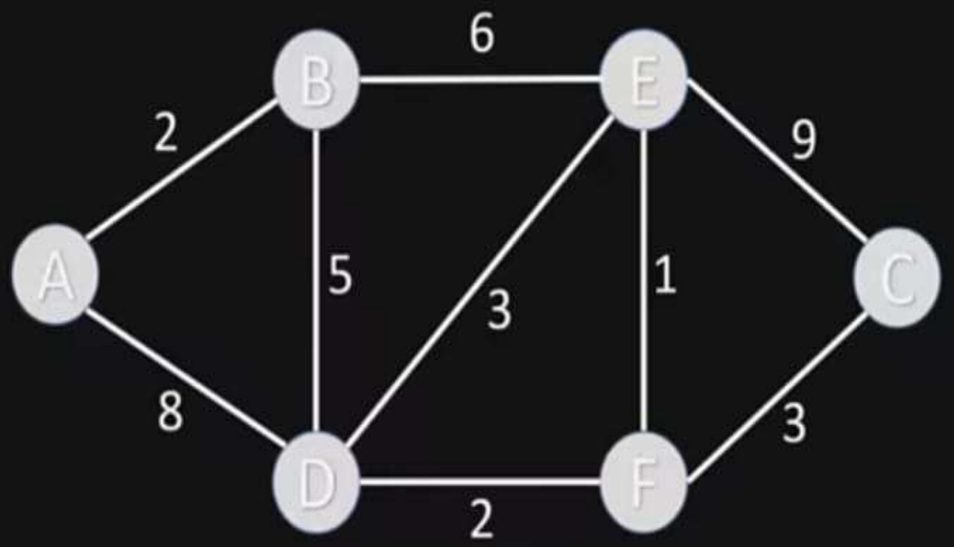
Node	Shortest Distance	Previous Node
A	0	
B	2	A
C	17	E
D	7	B
E	8	B
F	9	D

Visited Nodes: [A, B, D, E] Unvisited Nodes: [C, F]

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

4. Mark current node as visited



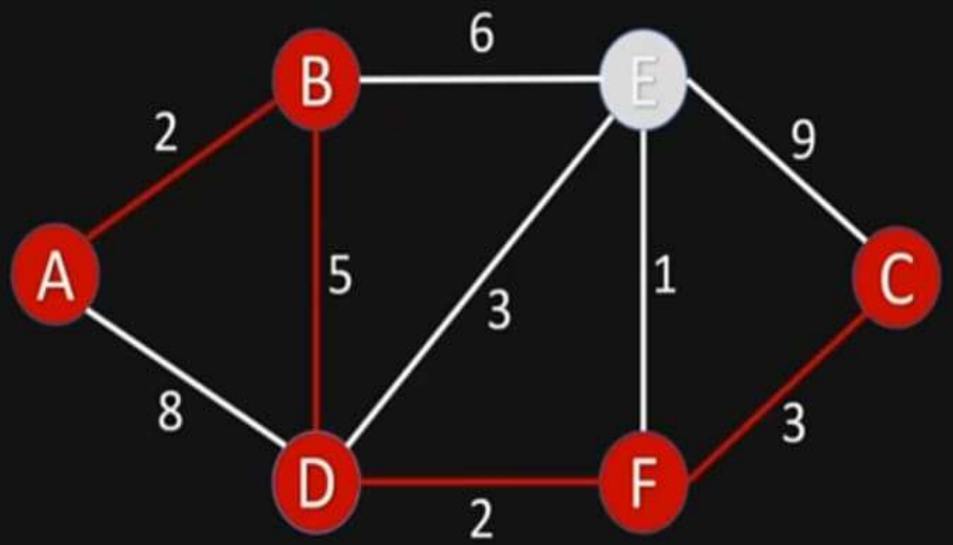
Node	Shortest Distance	Previous Node
A	0	
B	**2**	A
C	**12**	F
D	**7**	B
E	**8**	B
F	**9**	D

Visited Nodes: [A, B, D, E, F, C] Unvisited Nodes: []

Dijkstra's Algorithm

SHORTEST PATH-DIJKSTRA'S ALGORITHM

Get shortest path from A to C

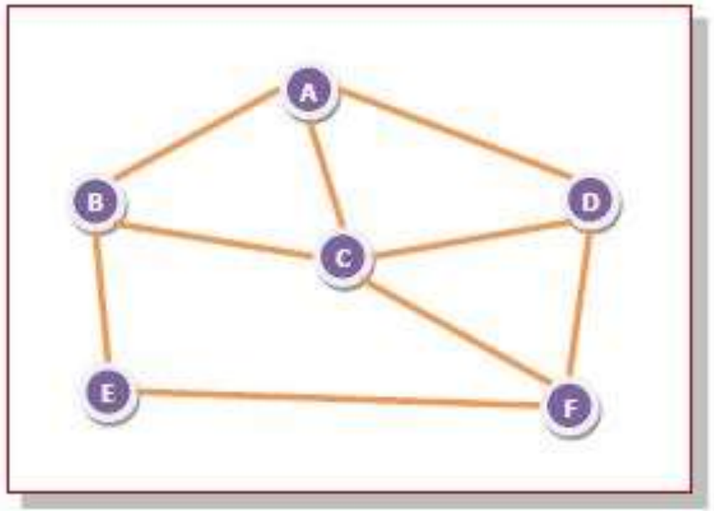


Node	Shortest Distance	Previous Node
A	0	
B	**2**	A
C	**12**	F
D	**7**	B
E	**8**	B
F	**9**	D

Dijkstra's Algorithm

ROUTING ALGORITHMS- FLOODING

- Flooding is a non-adaptive routing technique following this simple method: when a data packet arrives at a router, it is sent to all the outgoing links except the one it has arrived on.
- For example, let us consider the network in the figure, having six routers that are connected through transmission lines.



ROUTING ALGORITHMS- FLOODING

Using flooding technique

- An incoming packet to A, will be sent to B, C and D.
- B will send the packet to C and E.
- C will send the packet to B, D and F.
- D will send the packet to C and F.
- E will send the packet to F.
- F will send the packet to C and E.

ROUTING ALGORITHMS-TYPES OF FLOODING

Flooding may be of three types

- **Uncontrolled flooding** Here, each router unconditionally transmits the incoming data packets to all its neighbours.
- **Controlled flooding** They use some methods to control the transmission of packets to the neighbouring nodes. The two popular algorithms for controlled flooding are Sequence Number Controlled Flooding (SNCF) and Reverse Path Forwarding (RPF).
- **Selective flooding** Here, the routers don't transmit the incoming packets only along those paths which are heading towards approximately in the right direction, instead of every available paths.

ROUTING ALGORITHMS

Advantages of Flooding

- It is very simple to setup and implement, since a router may know only its neighbours.
- It is extremely robust. Even in case of malfunctioning of a large number routers, the packets find a way to reach the destination.
- All nodes which are directly or indirectly connected are visited. So, there are no chances for any node to be left out. This is a main criteria in case of broadcast messages.
- The shortest path is always chosen by flooding.

ROUTING ALGORITHMS

Limitations of Flooding

- Flooding tends to create an infinite number of duplicate data packets, unless some measures are adopted to damp packet generation.
- It is wasteful if a single destination needs the packet, since it delivers the data packet to all nodes irrespective of the destination.
- The network may be clogged with unwanted and duplicate data packets. This may hamper delivery of other data packets.

ROUTING ALGORITHMS-DISTANCE VECTOR ROUTING (DVR)

Distance Vector Routing (DVR) Protocol is a method used by routers to find the best path for data to travel across a network.

Each router keeps a table that shows the shortest distance to every other router, based on the number of hops (or steps) needed to reach them.

Routers share this information with their neighbors, allowing them to update their tables and find the most efficient routes.

This protocol helps ensure that data moves quickly and smoothly through the network.

What is the Distance Vector Routing Algorithm?

- The protocol requires that a router inform its neighbors of topology changes periodically. Historically known as the old ARPANET routing algorithm (or known as the Bellman-Ford algorithm).

ROUTING ALGORITHMS-DISTANCE VECTOR ROUTING (DVR)

Bellman-Ford Basics

Each router maintains a Distance Vector table containing the distance between itself and All possible destination nodes. Distances, based on a chosen metric, are computed using information from the neighbors' distance vectors.

Information kept by DV router:

- Each router has an ID
- Associated with each link connected to a router, there is a link cost (static or dynamic).
- Intermediate hops

Distance Vector Table Initialization:

- Distance to itself = 0
- Distance to ALL other routers = infinity number.

ROUTING ALGORITHMS-DISTANCE VECTOR ROUTING (DVR)

The DV calculation is based on minimizing the cost to each destination

- Formula:Dx(y) = Estimate of least cost from x to y
- Here x is source and y is destination

- Update distance based on neighbours

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

Here x is source,y destination,v is intermediate node,

DISTANCE VECTOR ROUTING- COUNT TO INFINITY PROBLEM

The **Count to Infinity Problem** is a well-known issue in **distance vector routing protocols** used in computer networks. It occurs when routers take an excessively long time to converge after a link failure, leading to inefficient routing and potential loops.

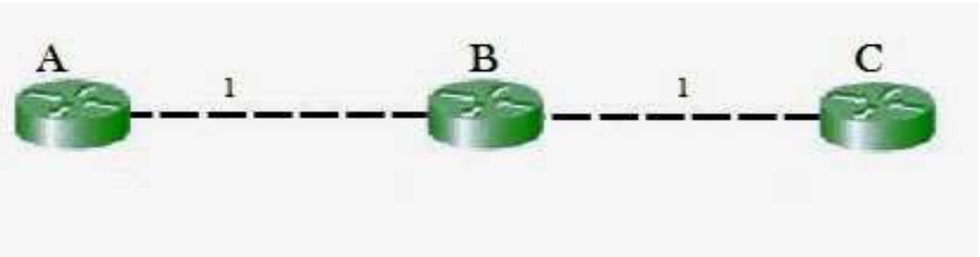
How It Happens:

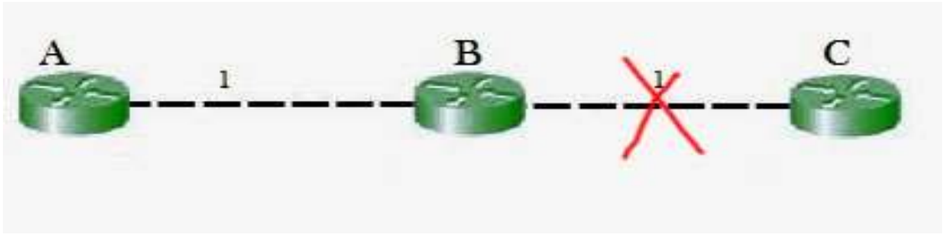
- **Initial State**: Routers exchange distance vectors (routing tables) with their neighbors, containing the cost (or distance) to reach various destinations.
- **Link Failure**: If a link between two routers fails, the affected routers update their routing tables to reflect the increased cost (often set to infinity) to reach the destination through the failed link.
- **Propagation of Incorrect Information**: Neighboring routers may not immediately recognize the failure and continue to advertise outdated or incorrect routes.
- **Counting to Infinity**: The routers incrementally increase the cost to the unreachable destination, step by step, until the cost reaches a predefined "infinity" value. This process can take a long time, especially in larger networks.

DISTANCE VECTOR ROUTING- COUNT TO INFINITY PROBLEM

Counting to infinity problem:

In this example, the Bellman-Ford algorithm will converge for each router, they will have entries for each other. B will know that it can get to C at a cost of 1, and A will know that it can get to C via B at a cost of 2.





If the link between B and C is disconnected, then B will know that it can no longer get to C via that link and will remove it from its table.

Before it can send any updates it's possible that it will receive an update from A which will be advertising that it can get to C at a cost of 2.

B can get to A at a cost of 1, so it will update a route to C via A at a cost of 3. A will then receive updates from B later and update its cost to 4.

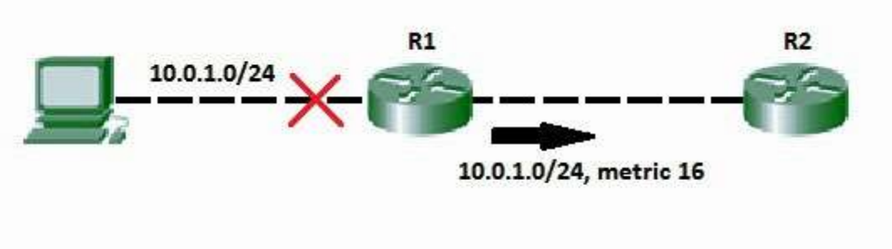
They will then go on feeding each other bad information toward infinity which is called as **Count to Infinity problem**.

DISTANCE VECTOR ROUTING- COUNT TO INFINITY PROBLEM

Solution for Count to Infinity problem:-

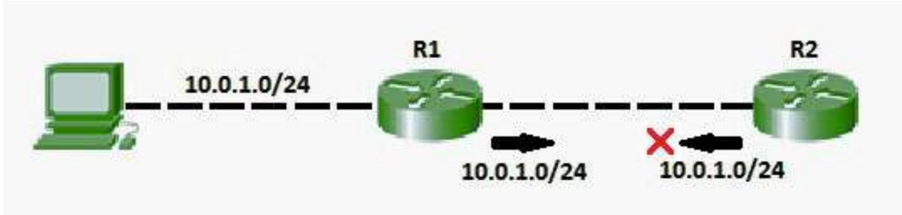
Route Poisoning:

When a route fails, distance vector protocols spread the *bad news* about a route failure by poisoning the route. Route poisoning refers to the practice of advertising a route, but with a special metric value called Infinity.



Split Horizon Rule:

A router will not send information about a route back in the same direction from which it learned it.



ROUTING ALGORITHMS-LINK STATE ROUTING (LSR)

Link state routing is a technique in which each router shares the knowledge of its neighbourhood with every other router in the internetwork.

The three keys to understand the Link State Routing algorithm:

- **Knowledge about the neighbourhood:** Instead of sending its routing table, a router sends the information about its neighbourhood only. A router broadcast its identities and cost of the directly attached links to other routers.
- **Flooding:** Each router sends the information to every other router on the internetwork except its neighbours. This process is known as Flooding. Every router that receives the packet sends the copies to all its neighbours. Finally, each and every router receives a copy of the same information.
- **Information sharing:** A router sends the information to every other router only when the change occurs in the information.

ROUTING ALGORITHMS-LINK STATE ROUTING (LSR)

Link State Routing has two phases:

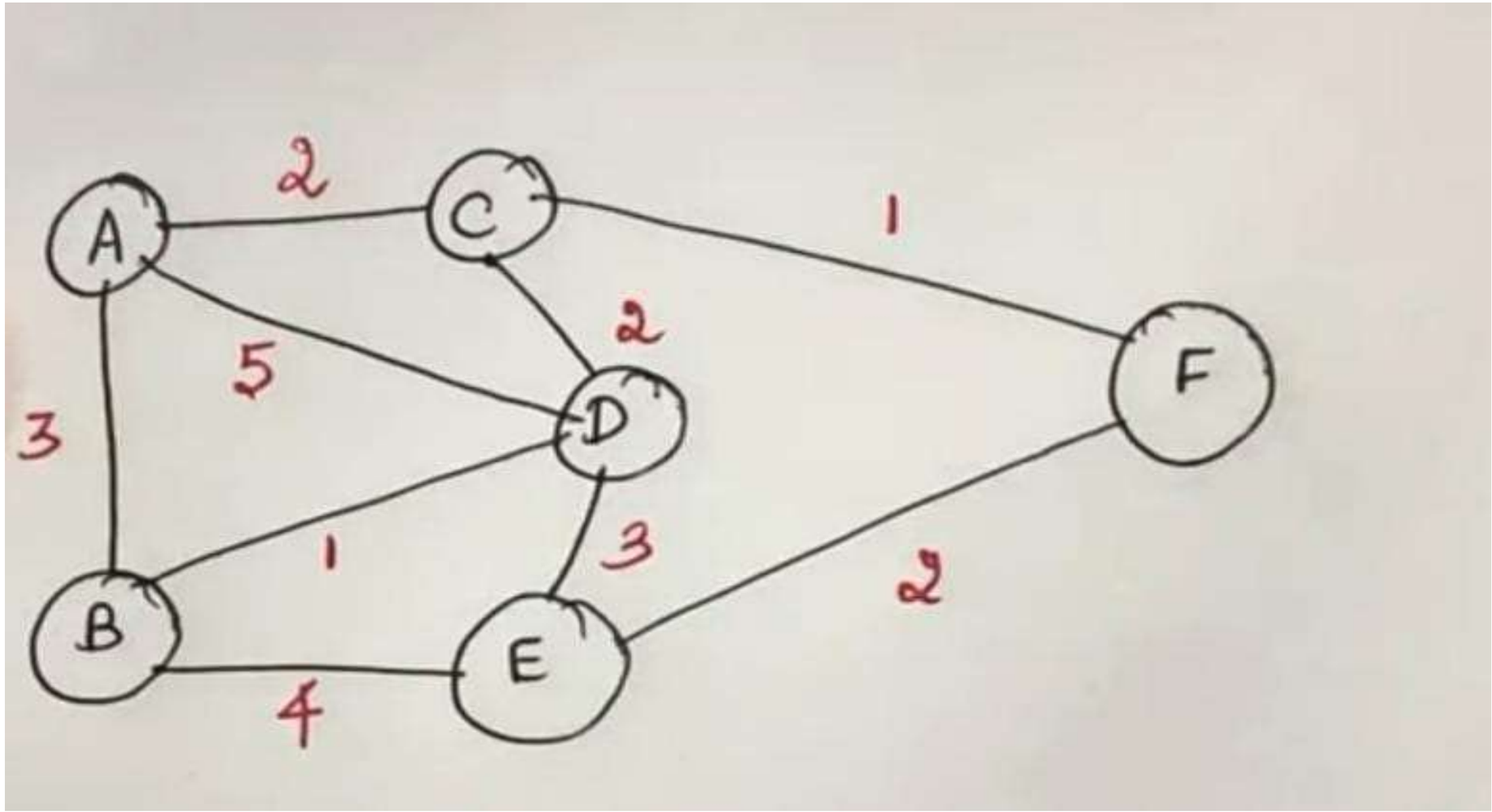
Reliable Flooding

- **Initial state:** Each node knows the cost of its neighbours.
- **Final state:** Each node knows the entire graph.

Route Calculation

- Each node uses Dijkstra's algorithm on the graph to calculate the optimal routes to all nodes.
- The Link state routing algorithm is also known as Dijkstra's algorithm which is used to find the shortest path from one node to every other node in the network.
- The Dijkstra's algorithm is an iterative, and it has the property that after k^{th} iteration of the algorithm, the least cost paths are well known for k destination nodes.

ROUTING ALGORITHMS-LINK STATE ROUTING (LSR)



ROUTING ALGORITHMS-PATH VECTOR ROUTING

- A **path-vector routing protocol** is a network routing protocol that maintains path information dynamically.
- **Path Vector Routing** is a network routing method where routers share the **full path** (list of networks or autonomous systems) to a destination. This helps prevent loops and allows routing decisions based on policies, not just distance. **BGP** is the best-known example.
- In a path-vector routing protocol, each entry in the routing table contains the destination network, the next router, and the path to reach the destination.
- This protocol is often used in conjunction with the Bellman-Ford routing algorithm to prevent the "Count to Infinity" problem.

ROUTING ALGORITHMS-PATH VECTOR ROUTING

Border Gateway Protocol (BGP)

- BGP is the **path vector routing protocol** used to exchange routing information between **Autonomous Systems (AS)** on the Internet.
  It decides the best path for data packets based on **path information and routing policies**, not just shortest distance.
- **Type:** Inter-domain routing protocol (between different networks)
- **Main Feature:** Keeps the **entire path of AS numbers** to avoid loops
- **Example:** When sending data from India to the USA, BGP decides which networks (ISPs) the data will pass through.

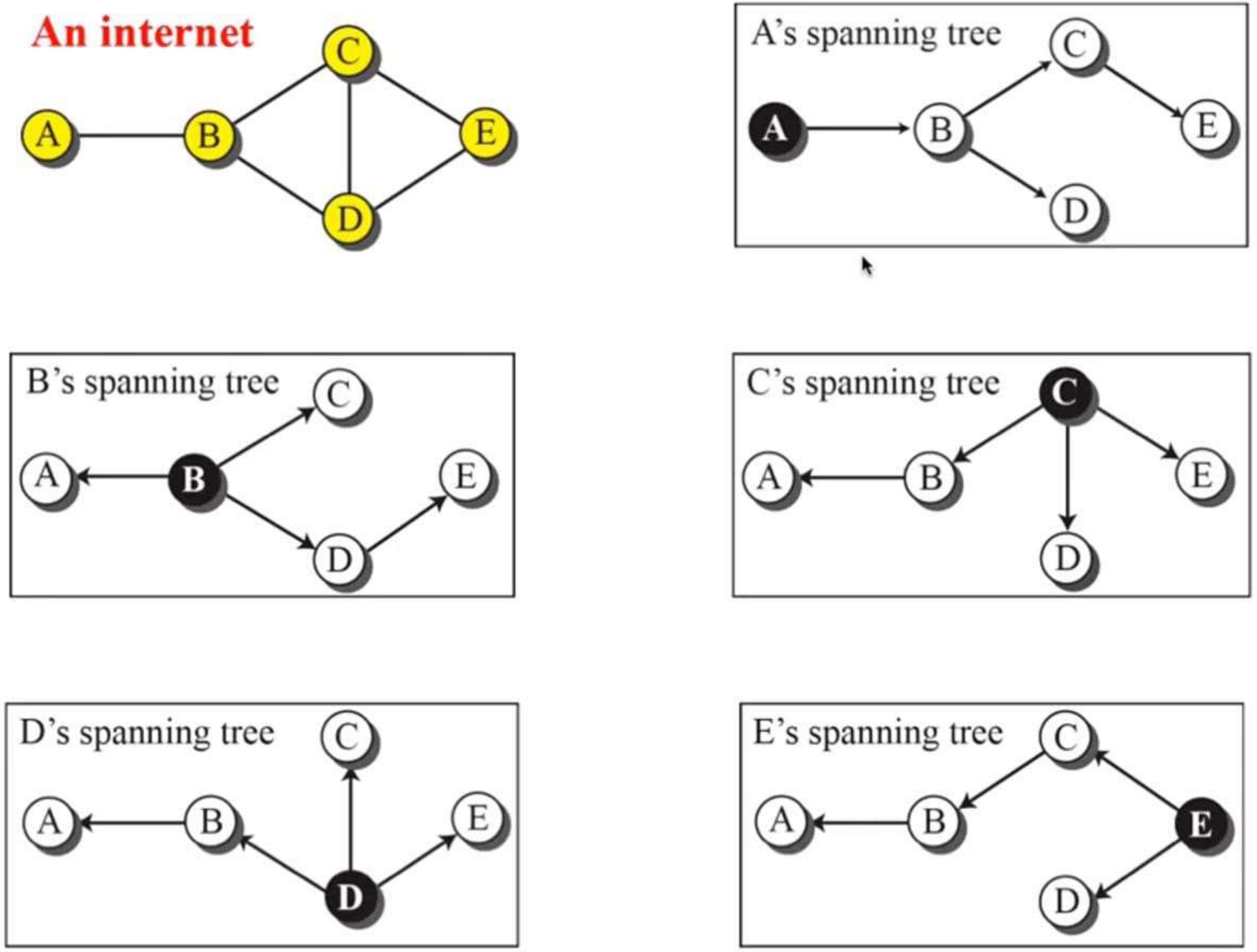
ROUTING ALGORITHMS-PATH VECTOR ROUTING

Phases of Path Vector Routing

- Path-vector routing involves three main phases:
- **Initiation**: The initial setup of the routing table and path information.
- **Sharing**: The exchange of path-vector messages between routers.
- **Updating**: The modification of routing tables and messages based on received path-vector information.

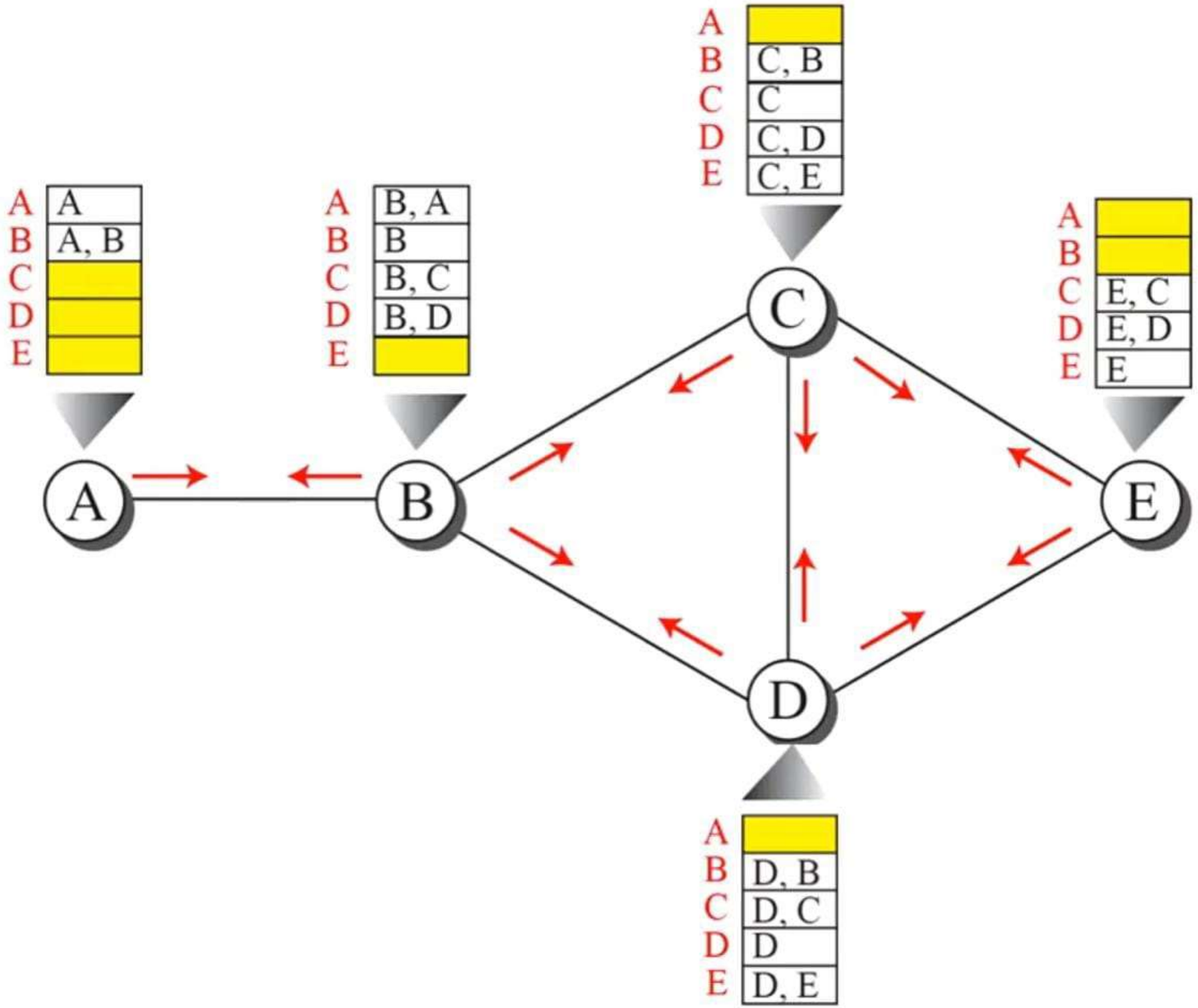
ROUTING ALGORITHMS-PATH VECTOR ROUTING

Figure 20.11: Spanning trees in path-vector routing



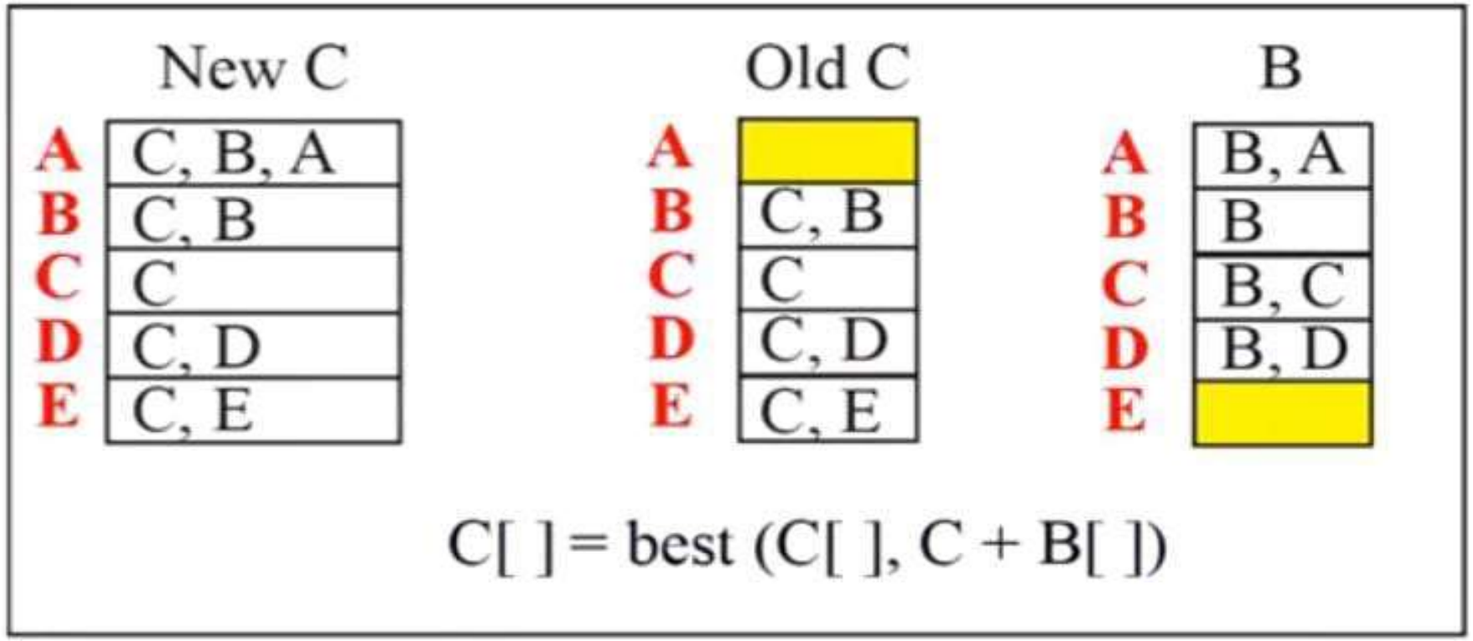
ROUTING ALGORITHMS-PATH VECTOR ROUTING

Figure 20.12: Path vectors made at booting time



ROUTING ALGORITHMS-PATH VECTOR ROUTING

Figure 20.13: ***Updating path vectors***



Event 1: C receives a copy of B's vector

ROUTING ALGORITHMS-HIERARCHICAL ROUTING

- In hierarchical routing, the **routers** are divided into regions. Each router has complete details about how to route packets to destinations within its own region.
- But it does not have any idea about the internal structure of other regions.
- As we know, in both LS and DV algorithms, every router needs to save some information about other routers. When network size is growing, the number of routers in the network will increase.
- Therefore, the size of routing table increases, then routers cannot handle network traffic as efficiently.
- To overcome this problem we are using hierarchical routing.

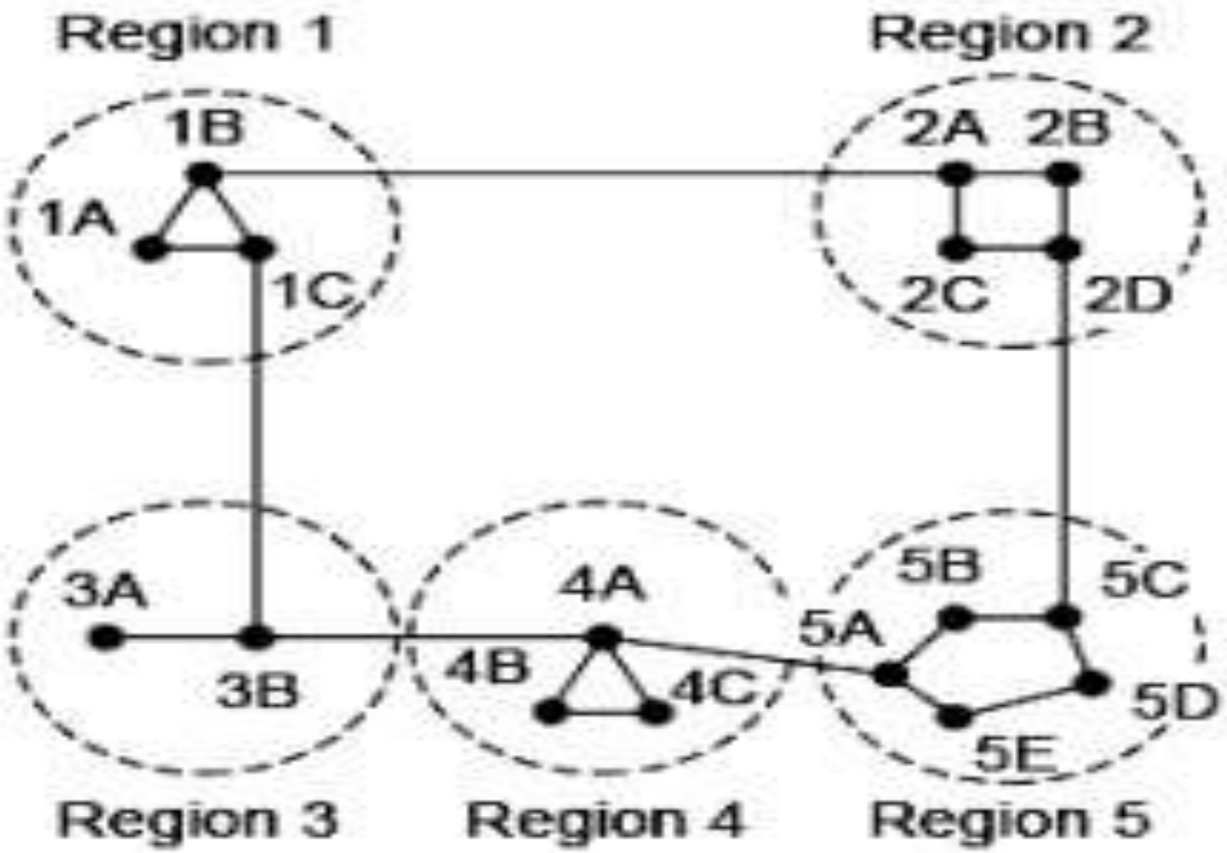
ROUTING ALGORITHMS-HIERARCHICAL ROUTING

- In hierarchical routing, routers are classified in groups called regions. Each router has information about the routers in its own region and it has no information about routers in other regions.
- So, routers save one record in their table for every other region.
- For huge networks, a two-level hierarchy may be insufficient hence, it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups and so on.

ROUTING ALGORITHMS-HIERARCHICAL ROUTING

Example

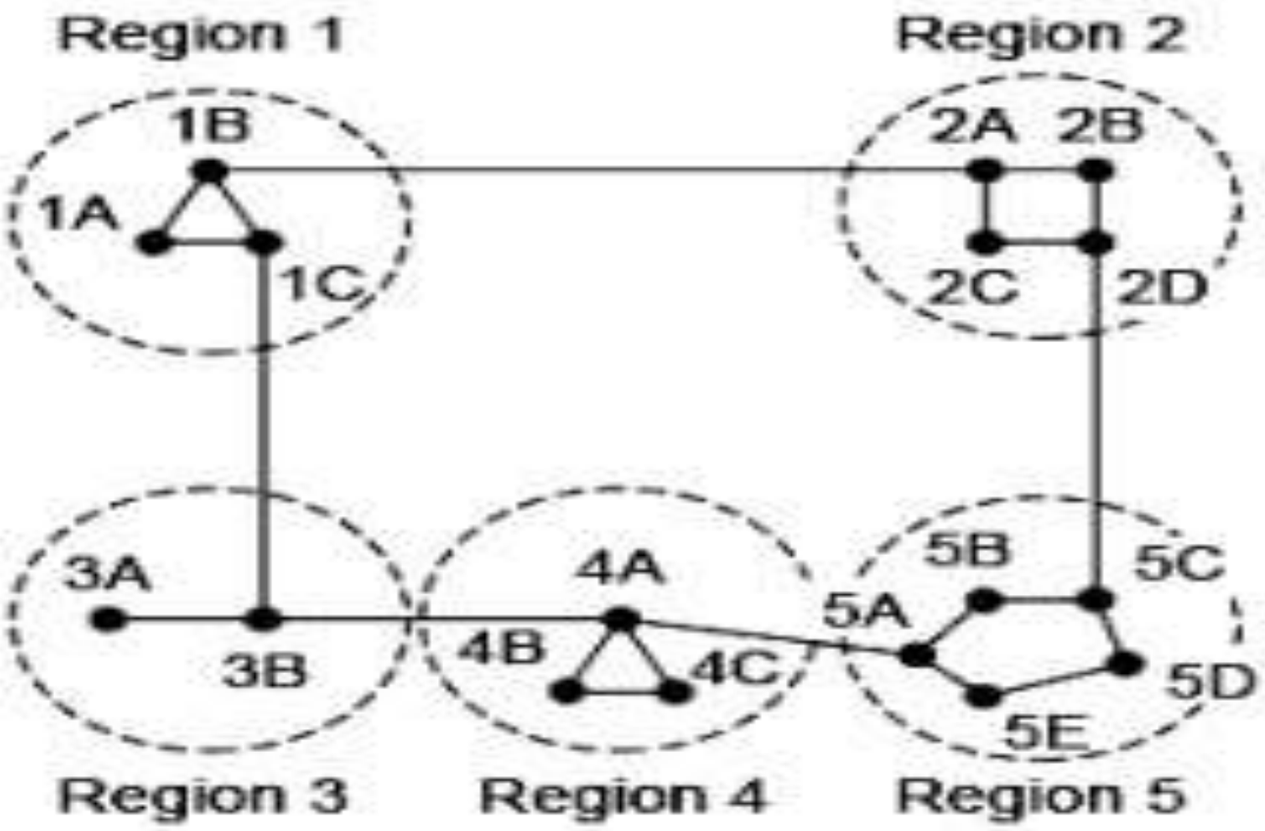
- Consider an example of two-level hierarchy with five regions as shown in figure



ROUTING ALGORITHMS-HIERARCHICAL ROUTING

Example

- Let see the full routing table for router 1A which has 17 entries, as shown below



Dest.	Line	Hops
1A	-	-
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

ROUTING ALGORITHMS-HIERARCHICAL ROUTING

Example

- When routing is done hierarchically then there will be only 7 entries as shown below
- Unfortunately, this reduction in table space comes with the increased path length.

Dest.	Line	Hops
1A	-	-
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

CONGESTION CONTROL

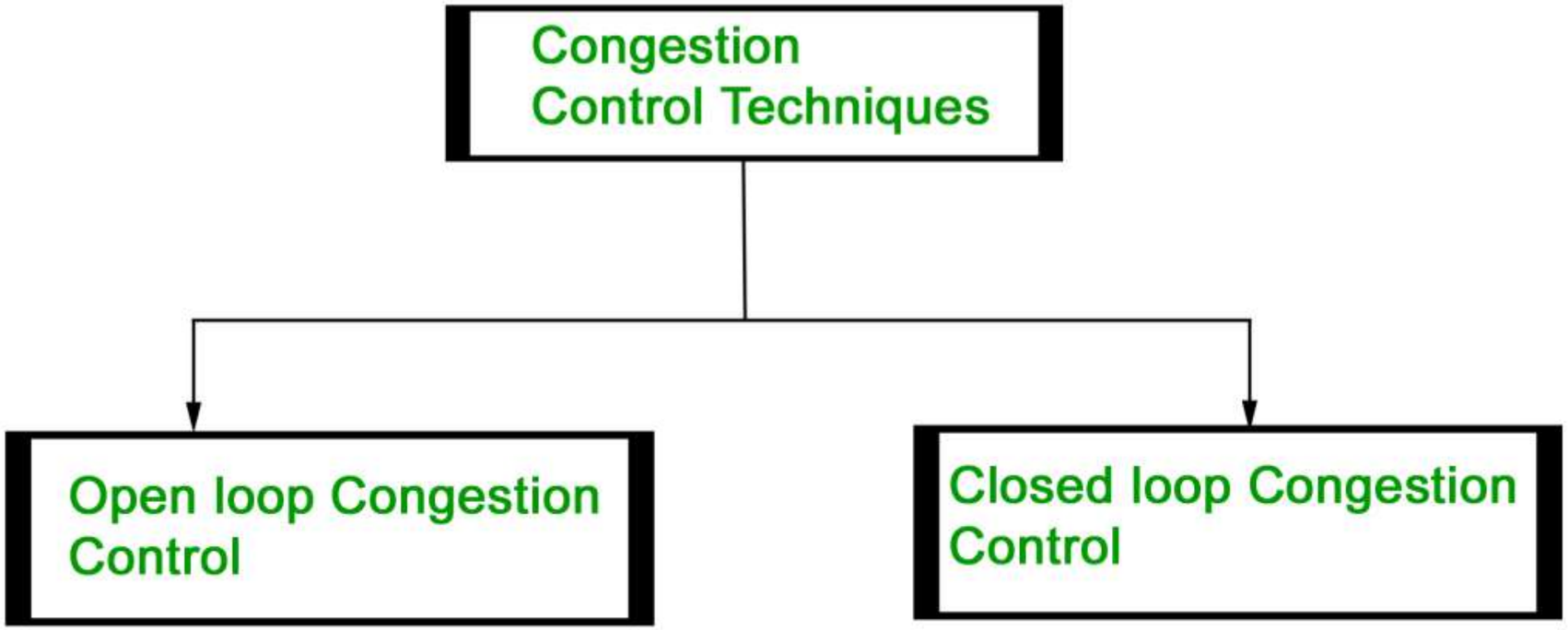
- Congestion control refers to techniques and algorithms used to manage and prevent network congestion, ensuring smooth data flow and efficient resource utilization. Congestion occurs when the volume of data exceeds the network's capacity, leading to delays, packet loss, and reduced performance.

Key Principles of Congestion Control

- Congestion control mechanisms are broadly categorized into open-loop and closed-loop solutions.
- Open-loop solutions focus on preventing congestion through proper design, such as traffic shaping and packet scheduling.
- Closed-loop solutions rely on feedback mechanisms to detect and respond to congestion dynamically.

CONGESTION CONTROL TECHNIQUES

Congestion control refers to the techniques used to control or prevent congestion. Congestion control techniques can be broadly classified into two categories:



CONGESTION CONTROL TECHNIQUES

Open Loop Congestion Control

Open loop congestion control policies are applied to prevent congestion before it happens. The congestion control is handled either by the source or the destination.

Policies adopted by open loop congestion control -

1.Retransmission Policy :

It is the policy in which retransmission of the packets are taken care of. If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. This transmission may increase the congestion in the network. To prevent congestion, retransmission timers must be designed to prevent congestion and also able to optimize efficiency.

CONGESTION CONTROL TECHNIQUES

2.Window Policy :

The type of window at the sender's side may also affect the congestion. Several packets in the Go-back-n window are re-sent, although some packets may be received successfully at the receiver side.

This duplication may increase the congestion in the network and make it worse. Therefore, Selective repeat window should be adopted as it sends the specific packet that may have been lost.

3.Discarding Policy :

A good discarding policy adopted by the routers is that the routers may prevent congestion and at the same time partially discard the corrupted or less sensitive packages and also be able to maintain the quality of a message.

CONGESTION CONTROL TECHNIQUES

4.Acknowledgment Policy :

Since acknowledgements are also the part of the load in the network, the acknowledgment policy imposed by the receiver may also affect congestion.

5.Admission Policy :

In admission policy a mechanism should be used to prevent congestion. Switches in a flow should first check the resource requirement of a network flow before transmitting it further. If there is a chance of a congestion or there is a congestion in the network, router should deny establishing a virtual network connection to prevent further congestion.

CONGESTION CONTROL TECHNIQUES

- **Closed Loop Congestion Control**

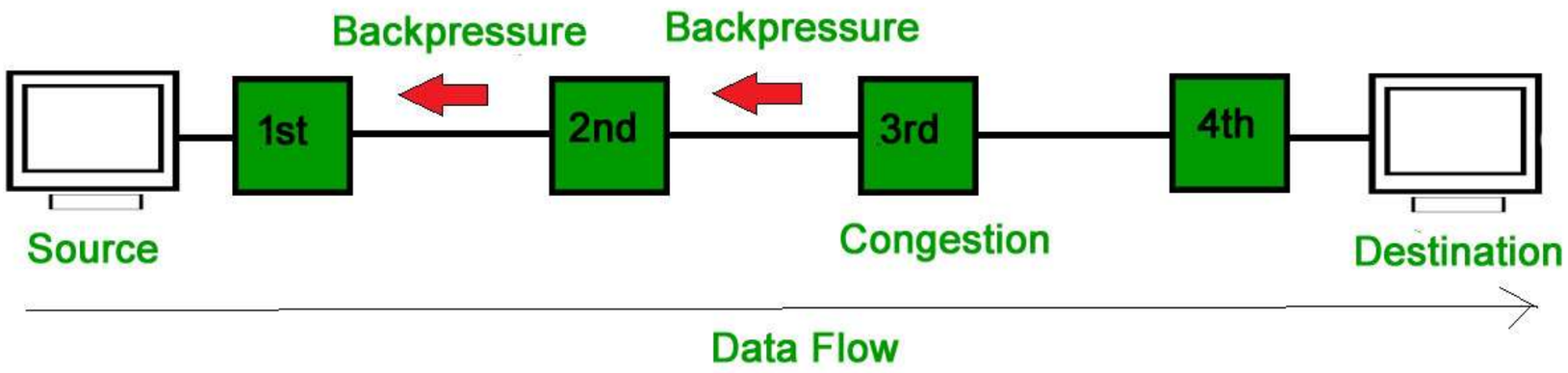
Closed loop congestion control techniques are used to treat or alleviate congestion after it happens.

1. Backpressure :

Backpressure is a technique in which a congested node stops receiving packets from upstream node. This may cause the upstream node or nodes to become congested and reject receiving data from above nodes.

Backpressure is a node-to-node congestion control technique that propagate in the opposite direction of data flow. The backpressure technique can be applied only to virtual circuit where each node has information of its above upstream node.

CONGESTION CONTROL TECHNIQUES

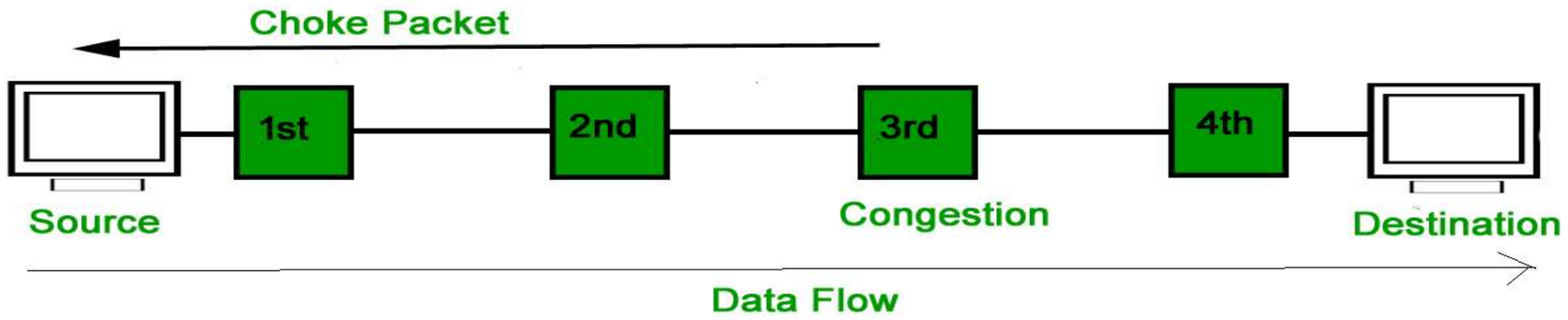


In above diagram the 3rd node is congested and stops receiving packets as a result 2nd node may be get congested due to slowing down of the output data flow. Similarly 1st node may get congested and inform the source to slow down.

CONGESTION CONTROL TECHNIQUES

2. Choke Packet Technique :

- Choke packet technique is applicable to both virtual networks as well as datagram subnets. A choke packet is a packet sent by a node to the source to inform it of congestion.
- Each router monitors its resources and the utilization at each of its output lines. Whenever the resource utilization exceeds the threshold value which is set by the administrator, the router directly sends a choke packet to the source giving it a feedback to reduce the traffic. The intermediate nodes through which the packets has traveled are not warned about congestion.



CONGESTION CONTROL TECHNIQUES

3. Implicit Signaling :

In implicit signaling, there is no communication between the congested nodes and the source. The source guesses that there is congestion in a network.

For example when sender sends several packets and there is no acknowledgment for a while, one assumption is that there is a congestion.

4. Explicit Signaling :

In explicit signaling, if a node experiences congestion it can explicitly send a packet to the source or destination to inform about congestion. The difference between choke packet and explicit signaling is that the signal is included in the packets that carry data rather than creating a different packet as in case of choke packet technique.

CONGESTION CONTROL TECHNIQUES

Explicit signaling can occur in either forward or backward direction.

- **Forward Signaling :** In forward signaling, a signal is sent in the direction of the congestion. The destination is warned about congestion. The receiver in this case adopt policies to prevent further congestion.
- **Backward Signaling :** In backward signaling, a signal is sent in the opposite direction of the congestion. The source is warned about congestion and it needs to slow down.

CONGESTION CONTROL ALGORITHM-LEAKY BUCKET ALGORITHM

- The **Leaky Bucket Algorithm** is a **traffic shaping** and **rate limiting** algorithm used in computer networks to control data transmission. It ensures that data packets are sent out at a **constant, smooth rate**, even if they arrive in bursts.
- Suppose we have a bucket in which we are pouring water at random points in time but we have to get water at a fixed rate to achieve this we will make a hole at the bottom of the bucket. This will ensure that the water coming out is at some fixed rate. If the bucket gets full, then we will stop pouring water into it.
- The input rate can vary but the output rate remains constant. Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate.

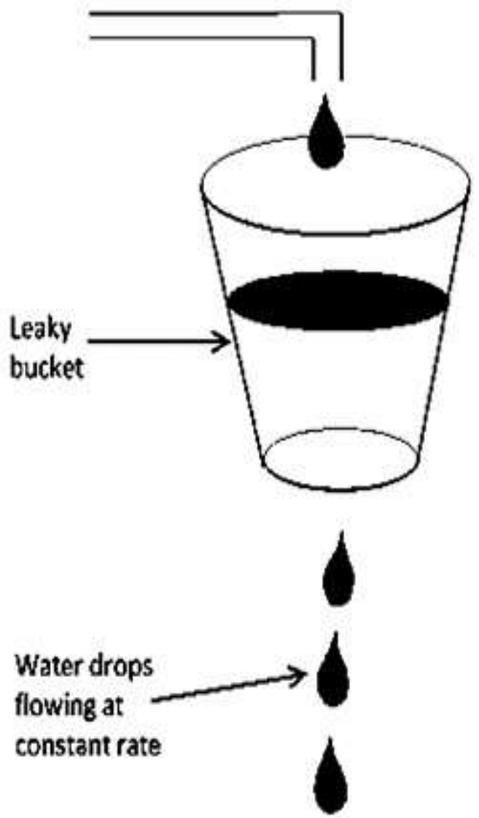
CONGESTION CONTROL ALGORITHM-LEAKY BUCKET ALGORITHM

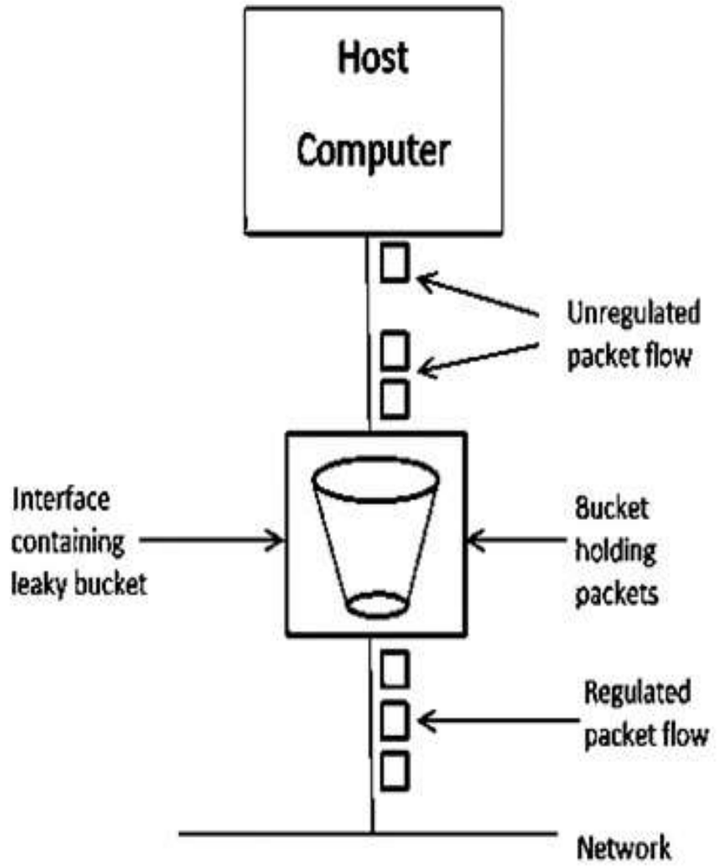
How It Works

1.Packets enter the bucket (queue).

2.Packets leave (transmit) at a fixed rate.

3.If the bucket (buffer) is full and new packets arrive → **packets are dropped** (overflow).





CONGESTION CONTROL ALGORITHM- TOKEN BUCKET ALGORITHM

The Token Bucket Algorithm is a **traffic control** and **rate-limiting** technique used in computer networks.

It ensures that:

- Packets can only be transmitted if there are enough **tokens** in the bucket.
- Each token represents permission to send a fixed number of bytes (or a single packet).

How Does It Work?

1.Tokens and Bucket:

- A "bucket" holds tokens, where each token represents permission to send a unit of data (e.g., a packet or byte).
- Tokens are added to the bucket at a constant rate, up to a maximum capacity.

2.Data Transmission:

- To send data, the system must consume tokens from the bucket.
- If enough tokens are available, the data is transmitted, and the corresponding tokens are removed.
- If insufficient tokens are available, the data is either delayed until tokens accumulate or discarded, depending on the implementation.

CONGESTION CONTROL ALGORITHM- TOKEN BUCKET ALGORITHM

3.Burst Handling:

- The bucket allows accumulation of tokens, enabling short bursts of traffic to be sent as long as the bucket has sufficient tokens.
- This makes the algorithm more flexible compared to strict rate-limiting mechanisms like the **Leaky Bucket Algorithm**.

Key Parameters

Token Generation Rate (r): The rate at which tokens are added to the bucket.

Bucket Capacity (B): The maximum number of tokens the bucket can hold.

Burst Size: The maximum amount of data that can be sent in a burst, determined by the bucket's capacity.

CONGESTION CONTROL ALGORITHM- TOKEN BUCKET ALGORITHM

Advantages

- Allows controlled bursts of traffic, making it suitable for real-world network scenarios.
- Ensures compliance with a specified average data rate.
- Simple and efficient to implement.

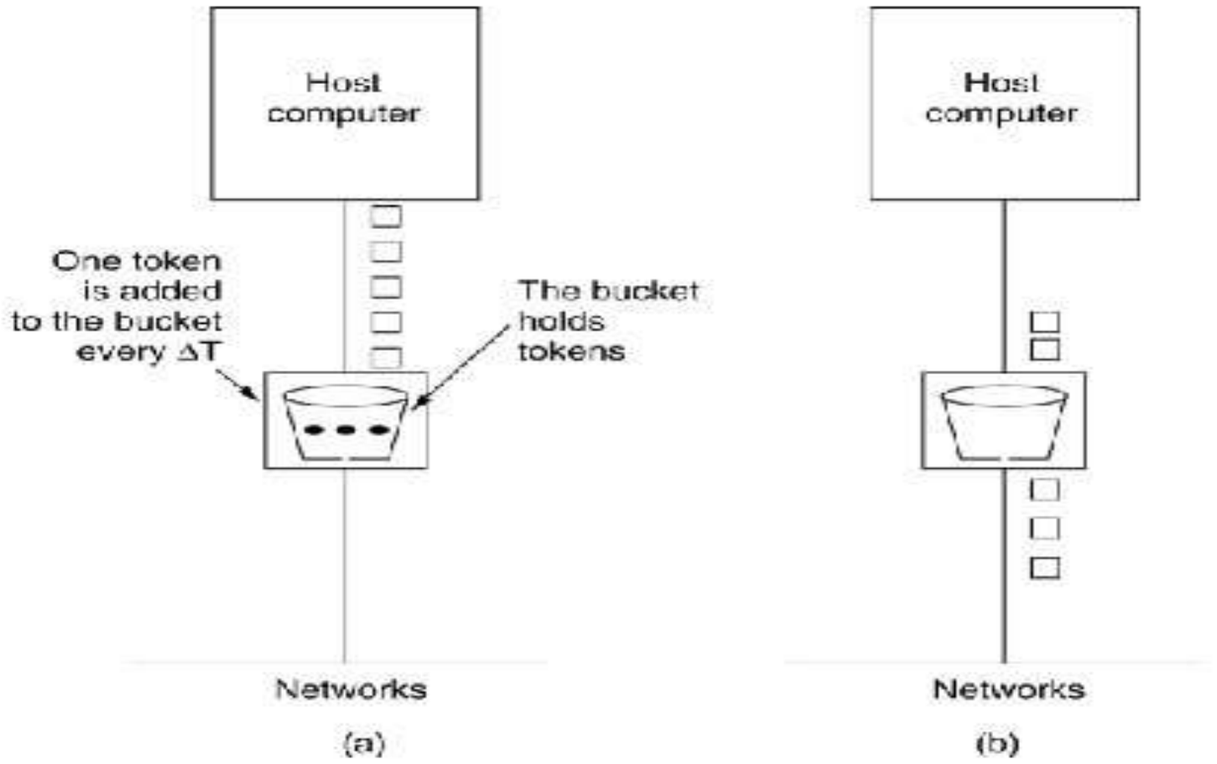
Applications

- **Traffic Shaping**: Smoothens traffic flow to prevent congestion.
- **Traffic Policing**: Ensures that traffic conforms to agreed-upon rates in Service Level Agreements (SLAs).
- **Quality of Service (QoS)**: Helps prioritize and manage network resources effectively.

The **Token Bucket Algorithm** is widely used in telecommunications and packet-switched networks to balance efficiency and flexibility in data transmission.

CONGESTION CONTROL ALGORITHM- TOKEN BUCKET ALGORITHM

The Token Bucket Algorithm



(a) Before. (b) After.

QUALITY OF SERVICE

- Quality of Service (QoS) is an important concept, particularly when working with multimedia applications.
- Multimedia applications, such as video conferencing, streaming services, and VoIP (Voice over IP), require certain bandwidth, latency, jitter, and packet loss parameters.
- QoS methods help ensure that these requirements are satisfied, allowing for seamless and re

What is Quality of Service?

- **Quality-of-service (QoS)** refers to traffic control mechanisms that seek to differentiate performance based on application or network-operator requirements or guaranteed performance to applications.
- The basic phenomenon for QoS is in terms of packet delay and losses of various kinds.

TYPES OF QUALITY OF SERVICE

- **Stateless Solutions -** Routers maintain no fine-grained state about traffic, one positive factor of it is that it is scalable and robust. But it has weak services as there is no guarantee about the kind of delay or performance in a particular application which we have to encounter.
- **Stateful Solutions -** Routers maintain a per-flow state as flow is very important in providing the Quality-of-Service i.e. providing powerful services such as guaranteed services and high resource utilization, providing protection, and is much less scalable and robust.

FLOW CHARACTERISTICS

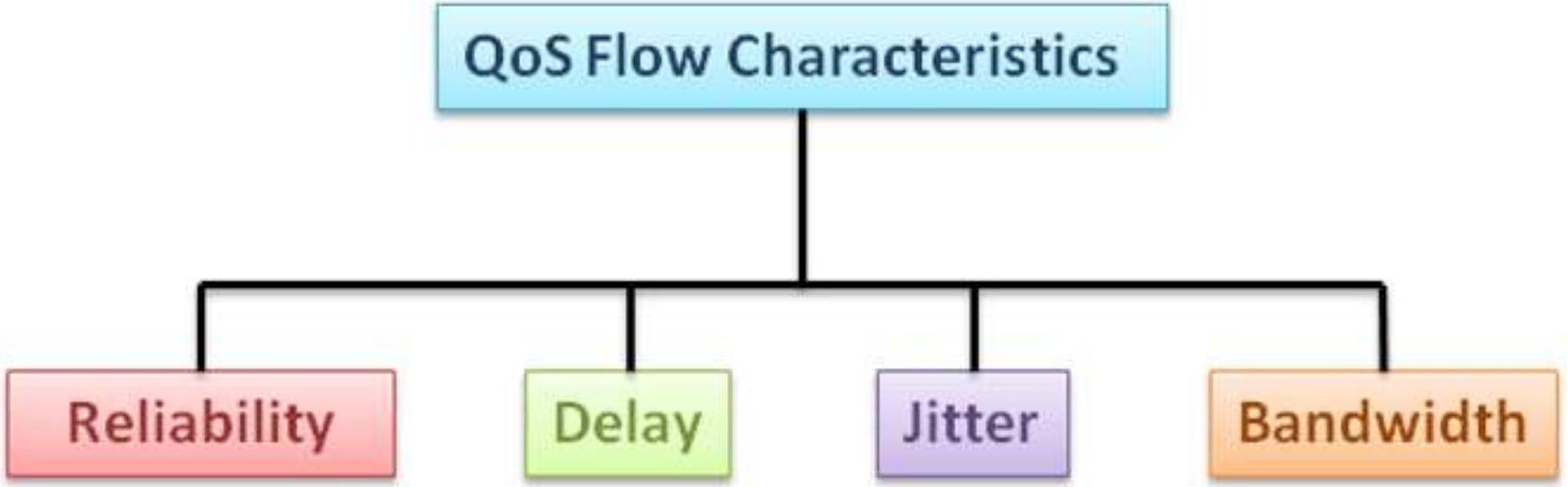


Figure 1: Quality of Service (QoS) Flow Characteristics

QOS PARAMETERS

- **Packet loss:** This occurs when network connections get congested, and routers and switches begin losing packets.
- **Latency:** This is how long it takes a packet to travel from its source to its destination. The latency should be as near to zero as possible.
- **Jitter:** This is the result of network congestion, time drift, and routing changes. Too much jitter can reduce the quality of voice and video communication.
- In reality, due to congestion, queuing, or route changes, packets might not arrive at the exact expected time. This difference between the expected arrival time and the actual arrival time is **jitter**.
- **Bandwidth:** This is a network communications link's ability to transmit the majority of data from one place to another in a specific amount of time.
- Delay: Delay can refer to specific components like **propagation delay** (distance/time), **transmission delay** (time to put bits on the wire), **processing delay** (router checks headers), and **queuing delay** (waiting in buffers).

UNDERSTANDING IPV4 AND IPV6

- IPv4 and IPv6 are versions of the Internet Protocol (IP), which assigns unique addresses to devices for communication over the internet. These protocols define how data is sent and received between devices.

IPv4 Overview

- IPv4, introduced in 1983, uses a **32-bit address** format, allowing approximately **4.3 billion unique addresses**. It is represented in **dot-decimal notation**, such as 192.168.1.1. IPv4 supports both **manual configuration** and **dynamic assignment** via DHCP. However, its limited address space has led to the development of IPv6.

UNDERSTANDING IPV4 AND IPV6

Key Characteristics of IPv4:

- **Address Length**: 32 bits.
- **Address Format**: Four decimal numbers separated by dots (e.g., 192.168.0.1).
- **Transmission**: Supports broadcasting.
- **Drawbacks**: Limited address space, lack of built-in security, and less efficient routing.

IP ADDRESS CLASSES

- IPv4 addresses are divided into **5 classes (A–E)**, based on the first octet (the first number, 0–255).
- 127.x.x.x is reserved for **loopback (localhost)**.

Examples:

- Class A: 10.0.0.0 – 10.255.255.255
- Class B: 172.16.0.0 – 172.31.255.255
- Class C: 192.168.0.0 – 192.168.255.255

Class	Range	Usage
A	1 – 126	Very large networks (e.g., ISPs, big organizations)
B	128 – 191	Medium-sized networks (e.g., universities)
C	192 – 223	Small networks (e.g., home, small business)
D	224 – 239	Special group communication
E	240 – 255	Reserved for research

TYPES OF IP ADDRESSES

Beyond classes, IPs are also categorized by **type**:

- **Private IP** → Used within local networks (e.g., home WiFi, office LAN).
- **Public IP** → Used on the internet, globally unique.
- **Static IP** → Manually assigned, does not change (used for servers).
- **Dynamic IP** → Assigned by DHCP, changes over time (used by most ISPs).
- **Unicast** → One-to-one communication.
- **Multicast** → One-to-many (Class D).
- **Broadcast** → One-to-all within a network (e.g., 192.168.1.255).

UNDERSTANDING IPV4 AND IPV6

IPv6 Overview

- IPv6, developed in 1998, uses a **128-bit address** format, providing an almost **unlimited number of unique addresses**.
- It is represented in **hexadecimal notation**, such as 2001:0db8:85a3::8a2e:0370:7334.
- IPv6 was designed to address IPv4's limitations, offering better scalability, efficiency, and security.

Unicast	One → One	A specific device
Broadcast	One → All	Everyone in subnet
Multicast	One → Many	A specific group
Anycast	One → One-of-Many	The nearest/best device

Key Characteristics of IPv6:

- **Address Length**: 128 bits.
- **Address Format**: Eight groups of hexadecimal numbers separated by colons (e.g., 2001:db8::1).
- **Transmission**: Uses multicast and anycast; no broadcasting.

REVA UNIVERSITY

THANK YOU