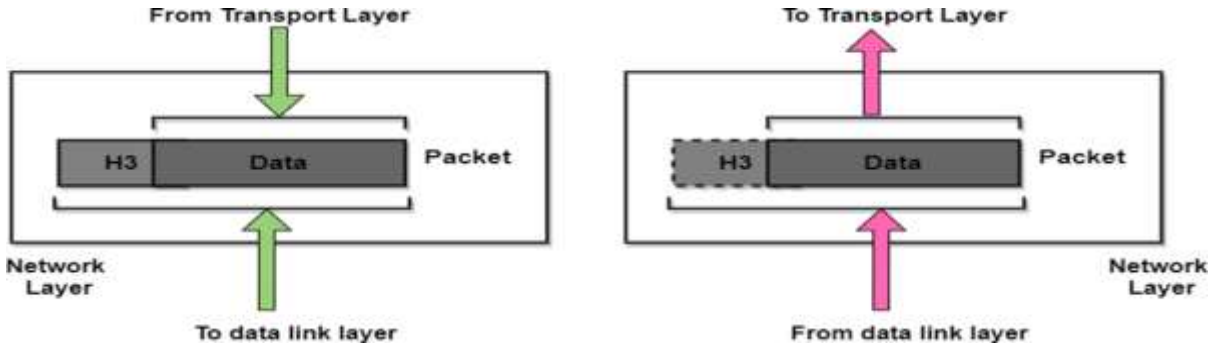
INTRODUCTION TO NETWORK LAYER

- The Network Layer is the third layer of the OSI model.
- It handles the service requests from the transport layer and further forwards the service request to the data link layer.
- The network layer translates the logical addresses into physical addresses
- It determines the route from the source to the destination and also manages the traffic problems such as switching, routing and controls the congestion of data packets.
- The main role of the network layer is to move the packets from sending host to the receiving host.



Functionalities

- Routing
- Logical Addressing
- Internetworking
- Fragmentation

➢NETWORK LAYER SERVICES

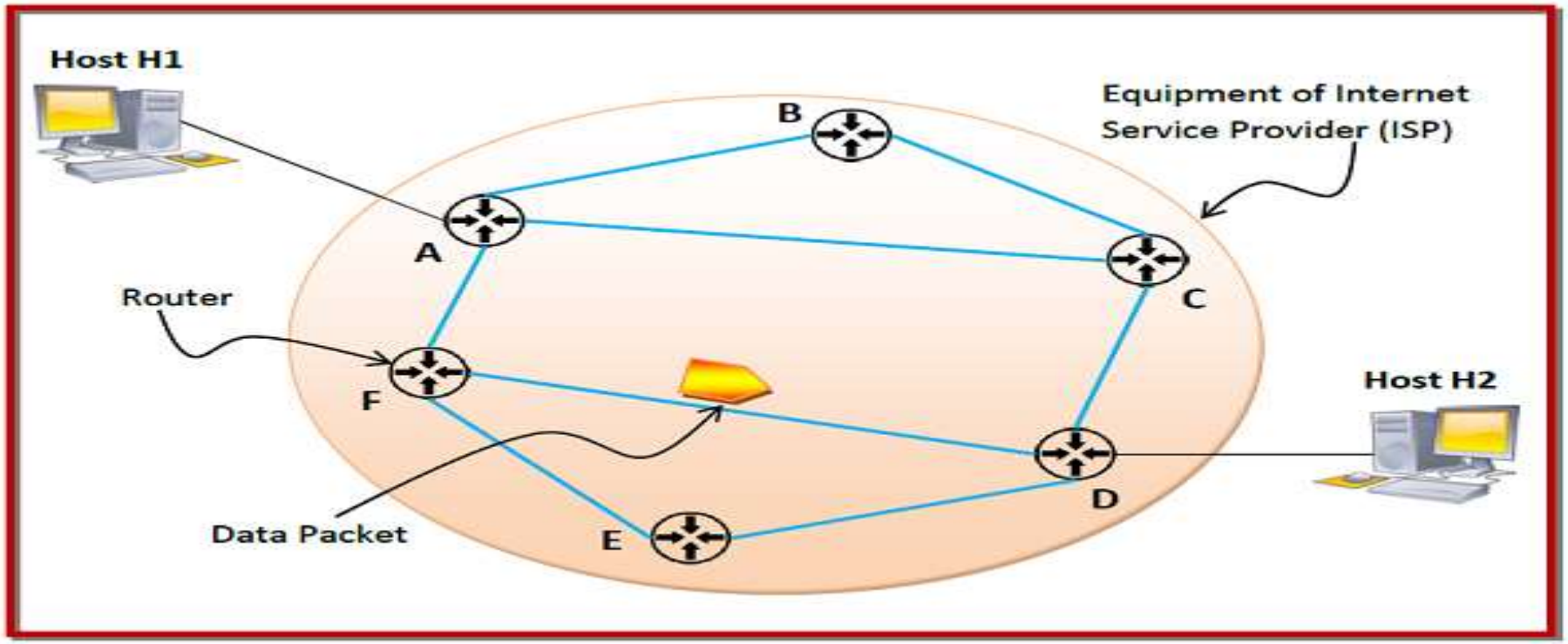
- Guaranteed delivery of Packets
- Guaranteed delivery with the bounded delay
- Transfer of packets in Order
- Security

Design Issues In Network Layer

The Network Layer Design Issues Are As Follows:

- Store & Forward Packet Switching
- Services Provided To The Transport Layer
- Implementation Of Connection – Less Service
- Implementation Of Connection Oriented Service

STORE & FORWARD PACKET



SERVICES PROVIDED TO THE TRANSPORT LAYER

- Service should be Independent of Network Topology
- Network addresses should be made available to transport with a uniform numbering plan
- Transport layer should be shielded from number, type & topology of routers present

ROUTING ALGORITHMS

- An algorithm is a procedure that lays down the route or path to transfer data packets from source to destination called Routing algorithm

- There are 2 categories in routing algorithm are as follows:
  - Adaptive routing algorithm
  - Non – adaptive routing algorithm

Adaptive Routing Algorithm:

- It is an algorithm that constructs the routing table based on network conditions called Adaptive Routing Algorithm
- It is also Called as Dynamic Routing.
- Route is based on change in topology and network traffic.

Types: There are 3 types in Adaptive Routing Algorithm are as follows

✓Centralized

✓Isolation Algorithm

✓Distributed Algorithm

Types of Adaptive Routing Alorithms:

- **Centralized Algorithm**: It also called as Global routing algorithm.

➢ It computes “least – cost path” between source & destination using complete & global knowledge about network.

➢Link State Algorithm is referred as Centralized Algorithm i.e., aware of cost of each link in network

- **Isolation Algorithm:** An algorithm that obtains routing information by using local information rather than gatherings information from other nodes
- **Distributed Algorithm:** It also known as De-centralized Algorithm.
- It computes Least – cost path between source & destination in iterative & distributed manner.
- A distance vector is a de-centralized algorithm i.e., it known direction through which packet is to be forwarded along with Least – cost path

Non Adaptive Routing Algorithms:

- It is also Called as Static Routing.
- An algorithm that constructs static table to determine which node to send packet is called Non – adaptive Routing Algorithm.
- **Types:** There are 2 types in Non – adaptive Routing algorithms are as follows:
  - Flooding
  - Random Walks

Types of Non Adaptive Routing Algorithms:

- **Flooding:** Every incoming packet is sent to all outgoing links except 1 from it has been reached.
  - **Dis – advantage:** Node may contain several copies of a particular packet

- **Random Walks:** A packet sent by node to 1 of its neighbors' randomly
  - **Advantage is** uses of alternative routes very efficiently

Shortest Path Routing Algorithm

- In shortest path Routing, path length between each node is measured as a function of distance, bandwidth, average traffic, communication cost, queue length, measured delay etc.
- By changing weighing function, algorithm then computes shortest path measured accordingly to any 1 of a no. of criteria or combination of criteria.
- **Types:** There are 2 types in Shortest Path Routing are as follows:

➢DIJKSTRA's Algorithm

➢BELLMAN – FORD Algorithm

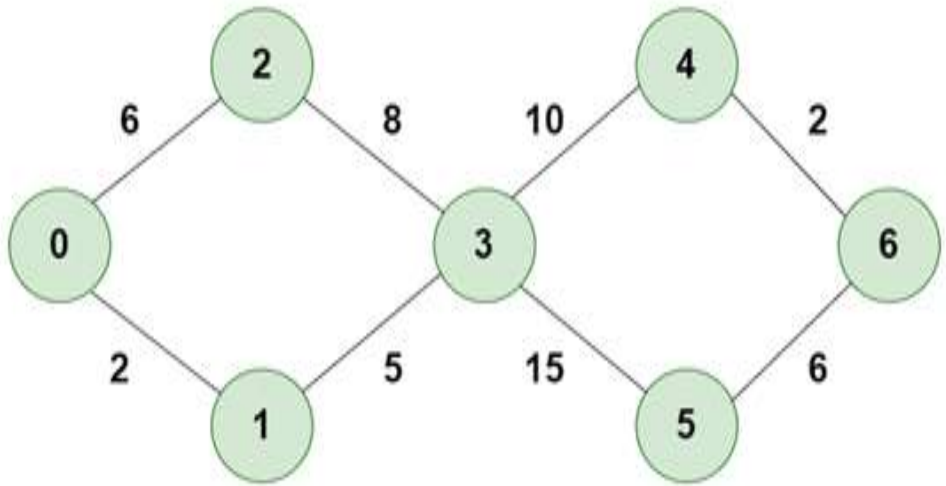
Dijkstra’s Algorithm

- The Dijkstra’s Algorithm is a greedy algorithm that 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:

- Mark the source node current distance as 0 and all others as infinity.
- Set the node with the smallest current distance among the non-visited nodes as the current node.
- 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.
- Make the current node as visited node.
- 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.

Example Graph



Dijkstra's Algorithm

BELLMAN FORD ALGORITHM

The BellMan Ford algorithm is a single source graph search algorithm which help us to find the shortest path between a source vertex and any other vertex in a give graph. We can use it in both weighted and unweighted graphs. This algorithm is slower than Dijkstra's Algorithm and it can also use negative edge weight.

Algorithm

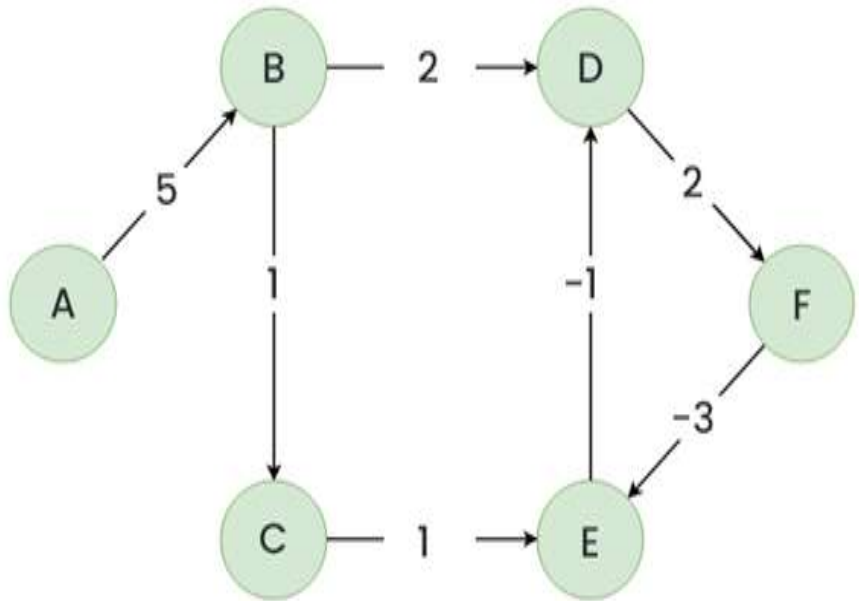
1: First we Initialize all vertices v in a distance array dist[] as INFINITY.

2: Then we pick a random vertex as vertex 0 and assign dist[0] =0.

3: Then iteratively update the minimum distance to each node (dist[v]) by comparing it with the sum of the distance from the source node (dist[u]) and the edge weight (weight) N-1 times.

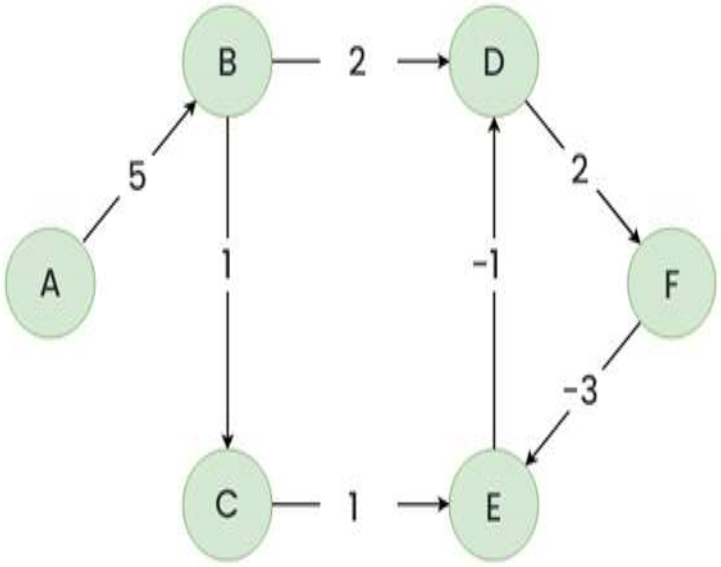
4: To identify the presence of negative edge cycles, with the help of following cases do one more round of edge relaxation.

- We can say that a negative cycle exists if for any edge uv the sum of distance from the source node (dist[u]) and the edge weight (weight) is less than the current distance to the largest node(dist[v])
- It indicates the absence of negative edge cycle if none of the edges satisfies case1.



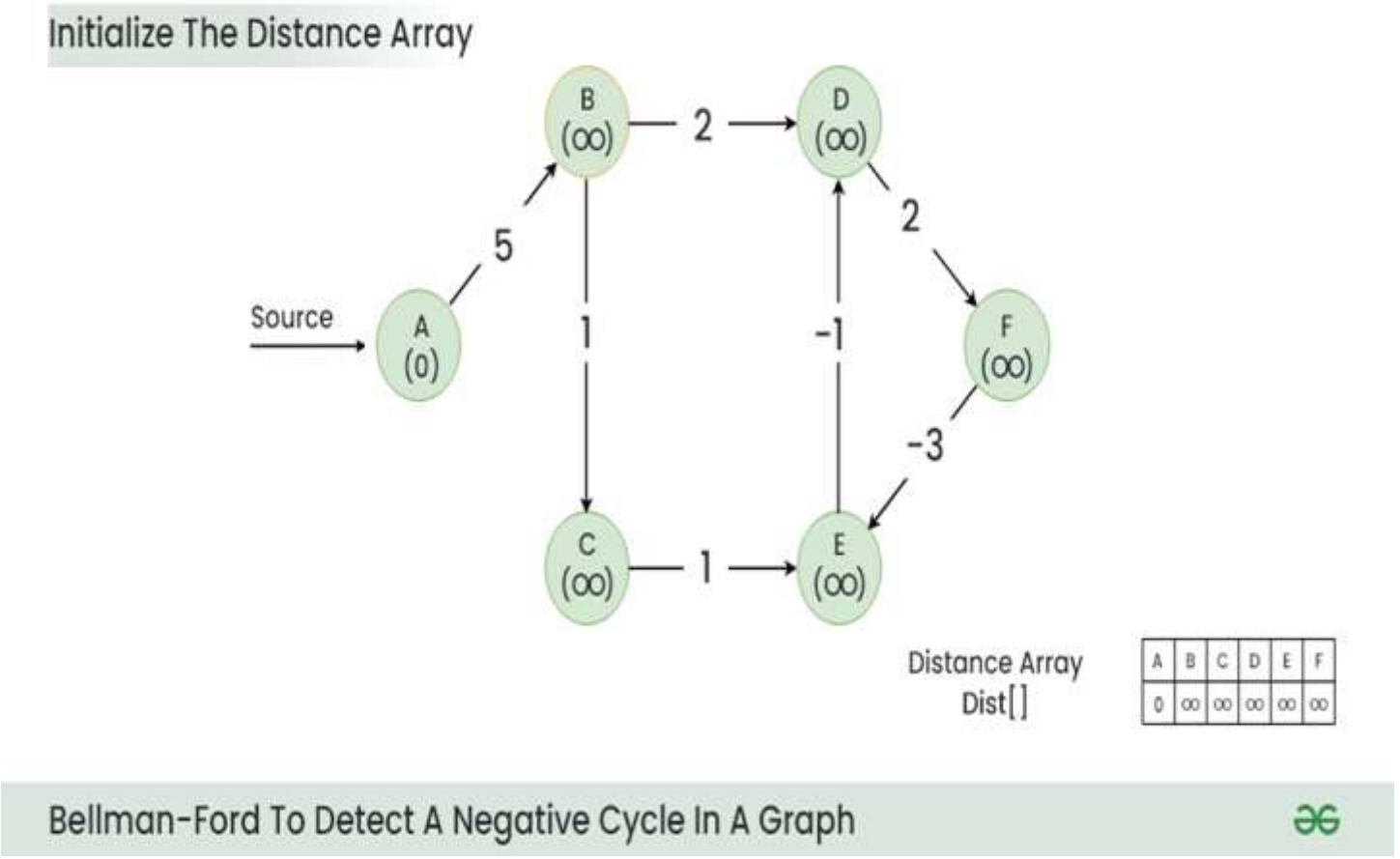
Bellman-Ford To Detect A Negative Cycle In A Graph

- Let’s suppose we have a graph which is given below and we want to find whether there exists a negative cycle or not using Bellman-Ford.

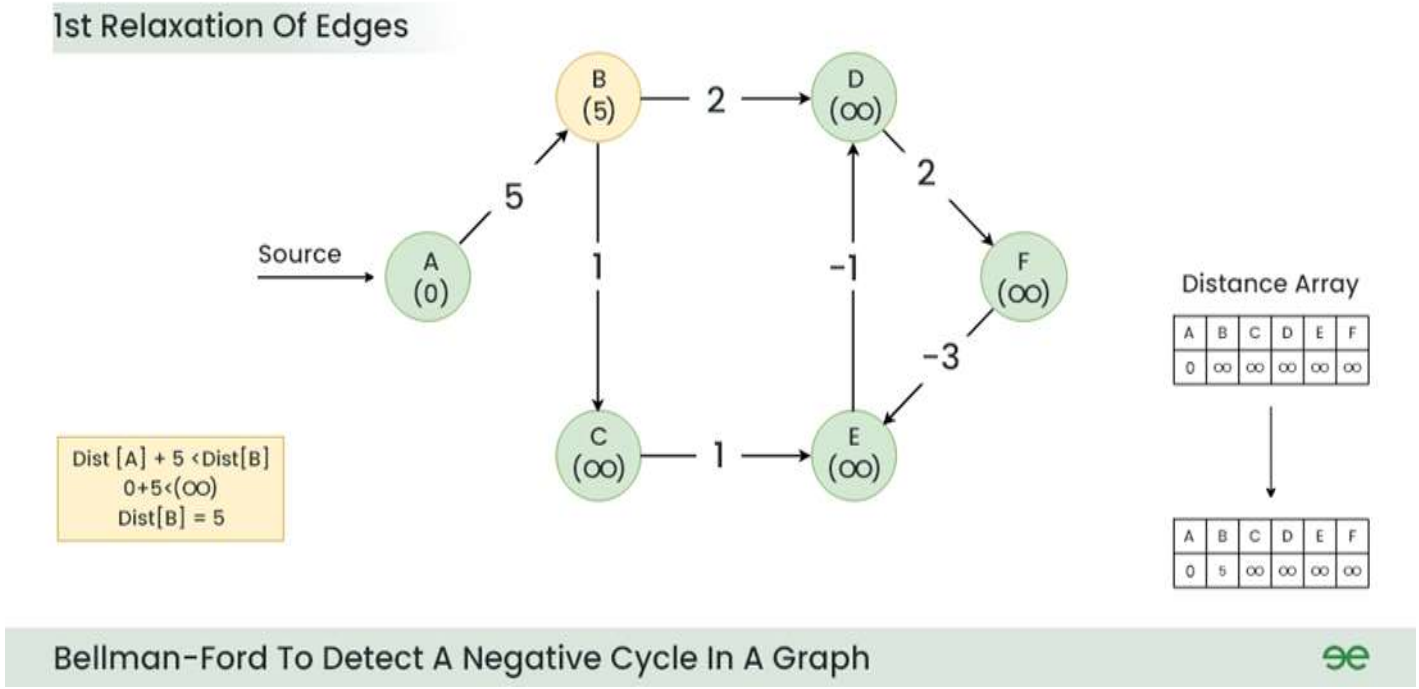


Bellman-Ford To Detect A Negative Cycle In A Graph

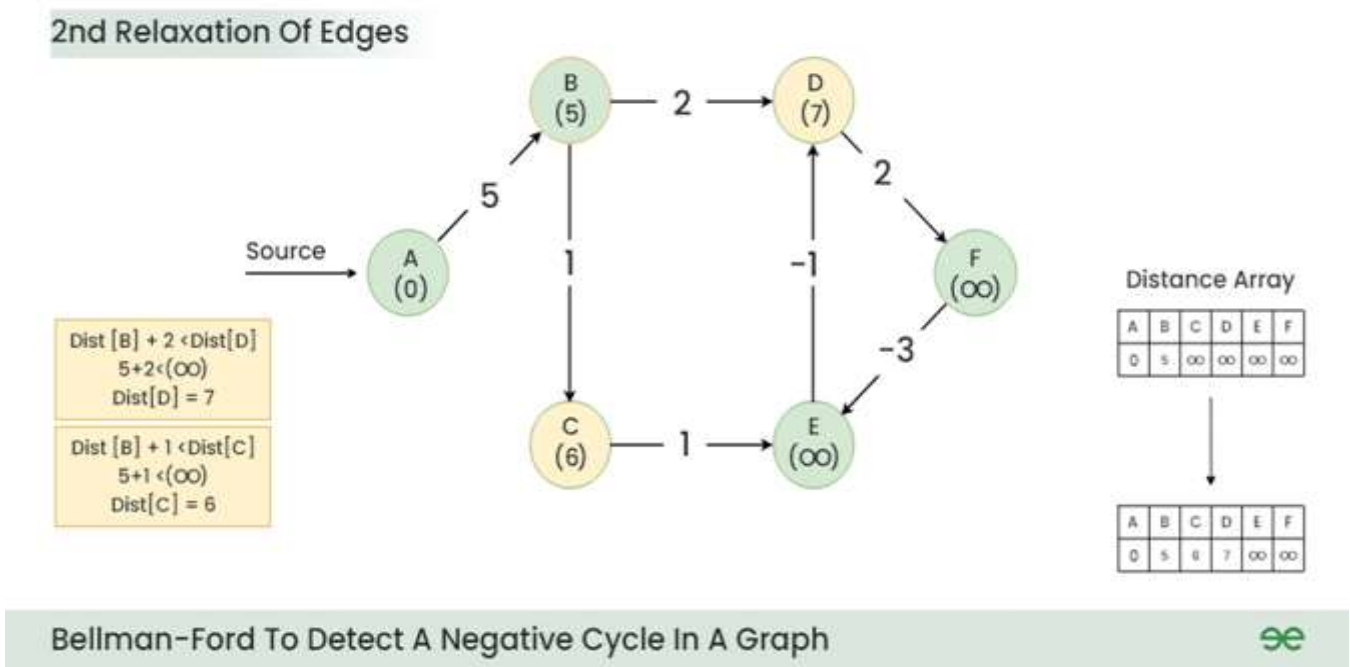
- **Step 1:** Initialize a distance array Dist[] to store the shortest distance for each vertex from the source vertex. Initially distance of source will be 0 and Distance of other vertices will be INFINITY.



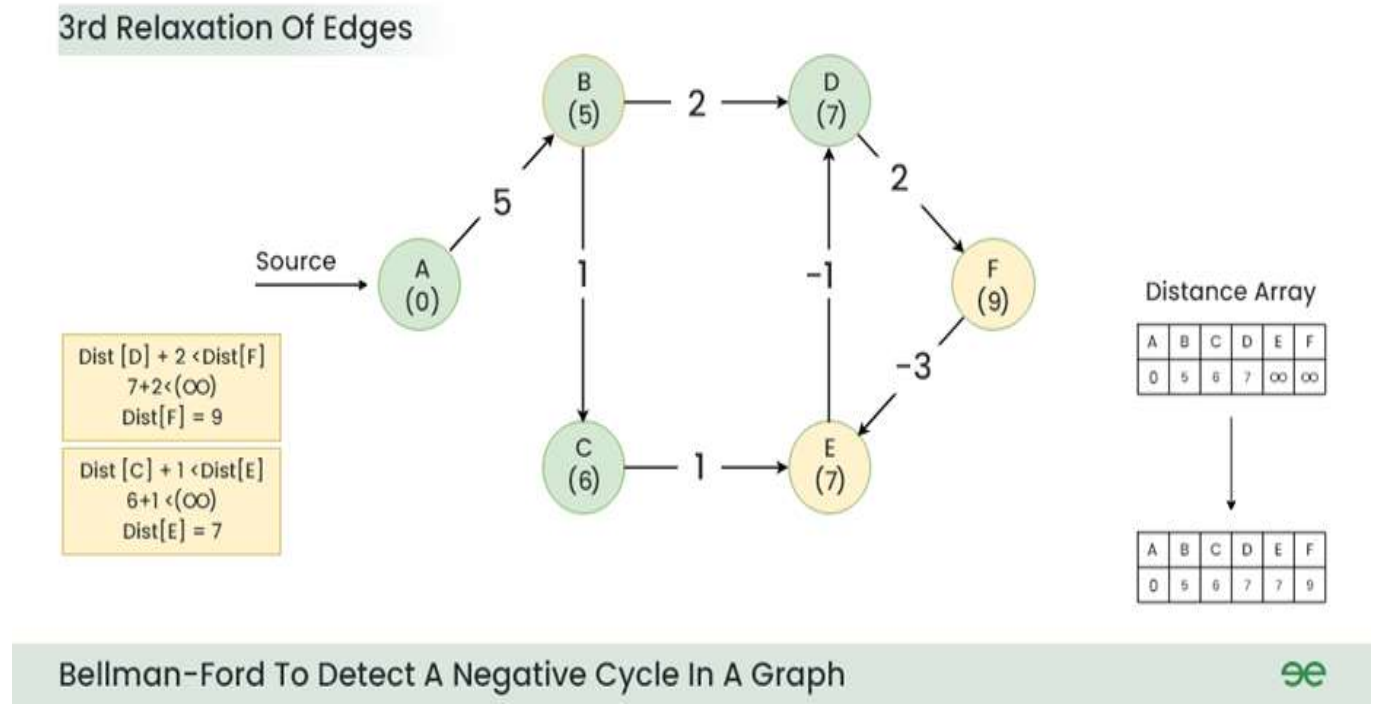
- Step 2:
- Start relaxing the edges, during 1st Relaxation:
- Current Distance of B > (Distance of A) + (Weight of A to B) i.e. Infinity > 0 + 5
- Therefore, Dist[B] = 5



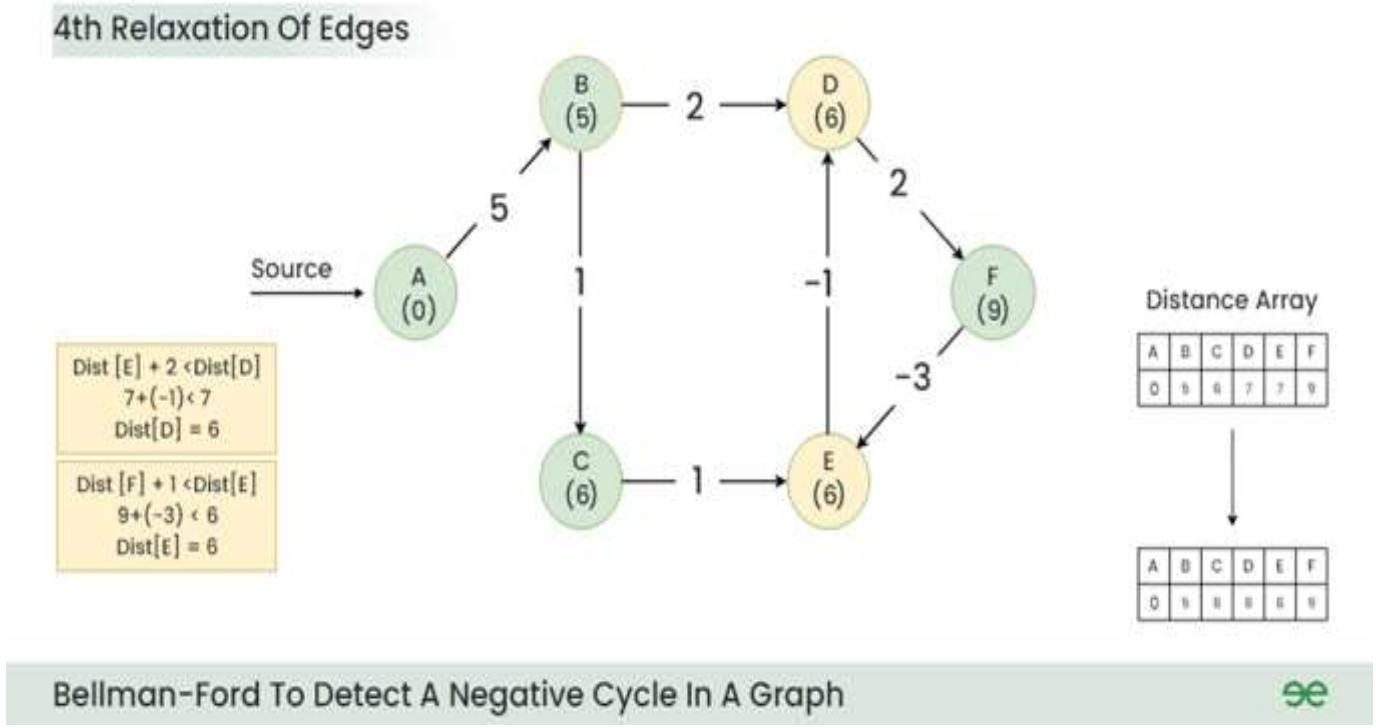
- Step 3:
- During 2nd Relaxation:
- Current Distance of D > (Distance of B) + (Weight of B to D) i.e. Infinity > 5 + 2
  - Dist[D] = 7
- Current Distance of C > (Distance of B) + (Weight of B to C) i.e. Infinity > 5 + 1
  - Dist[C] = 6



- Step 4:
- During 3rd Relaxation:
- Current Distance of F > (Distance of D) + (Weight of D to F) i.e. Infinity > 7 + 2
  - Dist[F] = 9
- Current Distance of E > (Distance of C) + (Weight of C to E) i.e. Infinity > 6 + 1
  - Dist[E] = 7

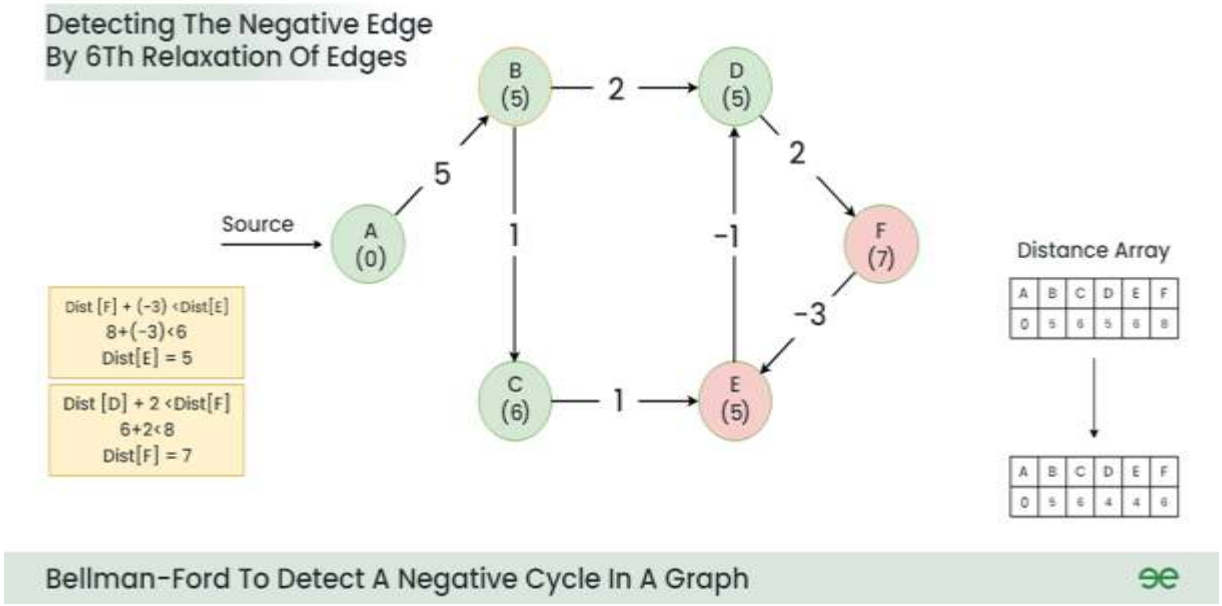


- Step 5:
- During 4th Relaxation:
- Current Distance of D > (Distance of E) + (Weight of E to D) i.e. 7 > 7 + (-1)
  - Dist[D] = 6
- Current Distance of E > (Distance of F) + (Weight of F to E) i.e. 7 > 9 + (-3)
  - Dist[E] = 6



- Step 6: During 5th Relaxation:
- Current Distance of F > (Distance of D) + (Weight of D to F) i.e. 9 > 6 + 2
  - Dist[F] = 8
- Current Distance of D > (Distance of E) + (Weight of E to D) i.e. 6 > 6 + (-1)
  - Dist[D] = 5
- Since the graph h 6 vertices, So during the 5th relaxation the shortest distance for all the vertices should have been calculated

- **Step 7:** Now the final relaxation i.e. the 6th relaxation should indicate the presence of negative cycle if there is any changes in the distance array of 5th relaxation.
- During the 6th relaxation, following changes can be seen:
- Current Distance of E > (Distance of F) + (Weight of F to E) i.e. 6 > 8 + (-3)
  - Dist[E]=5
- Current Distance of F > (Distance of D) + (Weight of D to F) i.e. 8 > 5 + 2
  - Dist[F]=7
- Since, we observer changes in the Distance array Hence ,we can conclude the presence of a negative cycle in the graph.



FLOODING

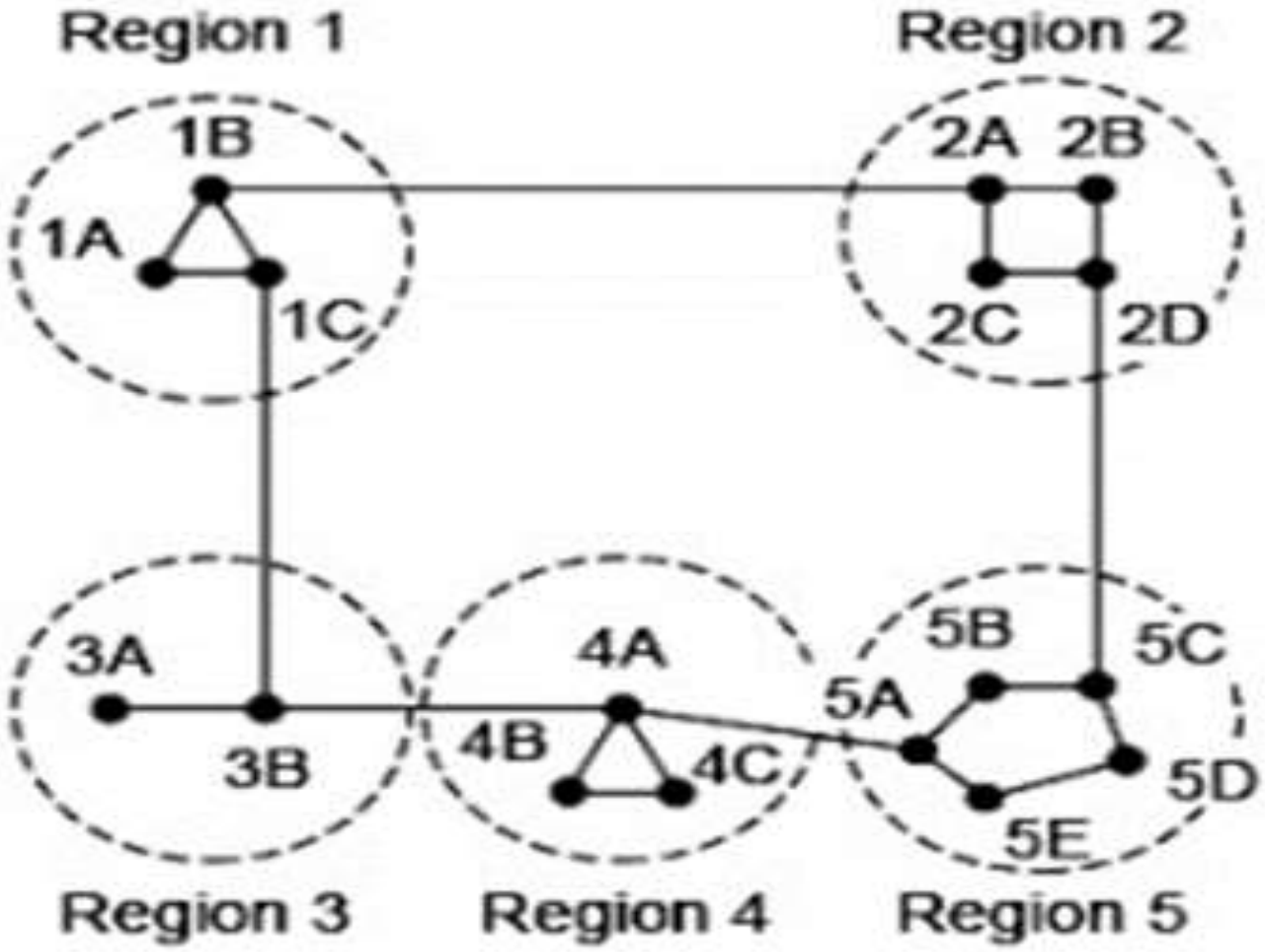
- A simple local technique is flooding, in which every incoming packet is sent out on every outgoing line except the one it arrived on.
- Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process.
- One such measure is to have a hop counter contained in the header of each packet that is decremented at each hop, with the packet being discarded when the counter reaches zero.
- Ideally, the hop counter should be initialized to the length of the path from source to destination.
- If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the full diameter of the network.
- Flooding with a hop count can produce an exponential number of duplicate packets as the hop count grows and routers duplicate packets they have seen before

- A better technique for damming the flood is to have routers keep track of which packets have been flooded, to avoid sending them out a second time.
- One way to achieve this goal is to have the source router put a sequence number in each packet it receives from its hosts.
- Each router then needs a list per source router telling which sequence numbers originating at that source have already been seen.
- If an incoming packet is on the list, it is not flooded.
- **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.

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, 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.
- 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.

Example


Region 1
Region 2
1B
2A
2B
1A
1C
2C
2D
3A
4A
5B
5C
5A
4B
4C
3B
5D
5E
Region 3
Region 4
Region 5

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

Destination	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
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

- When routing is done hierarchically then there will be only 7 entries as shown below –

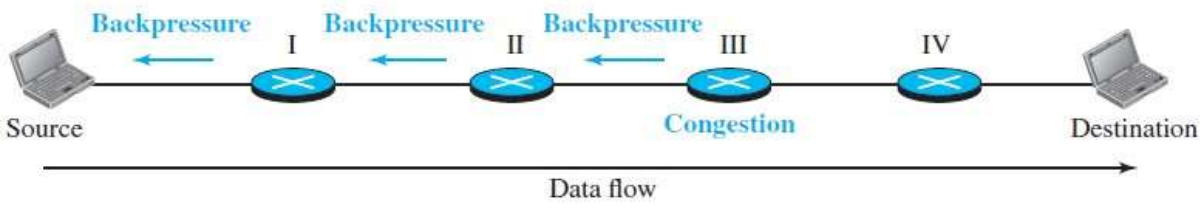
Destination	Line	Hops
1A	-	-
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

CONGESTION CONTROL ALGORITHM

- Congestion control refers to techniques and mechanisms that can either prevent congestion before it happens or remove congestion after it has happened. In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal).
- **Open-Loop Congestion Control**: In open- loop congestion control, policies are applied to prevent congestion before it happens. In these mechanisms, congestion control is handled by either the source or the destination.
  - Retransmission Policy
  - Window Policy
  - Acknowledgment Policy
  - Discarding Policy
  - Admission Policy

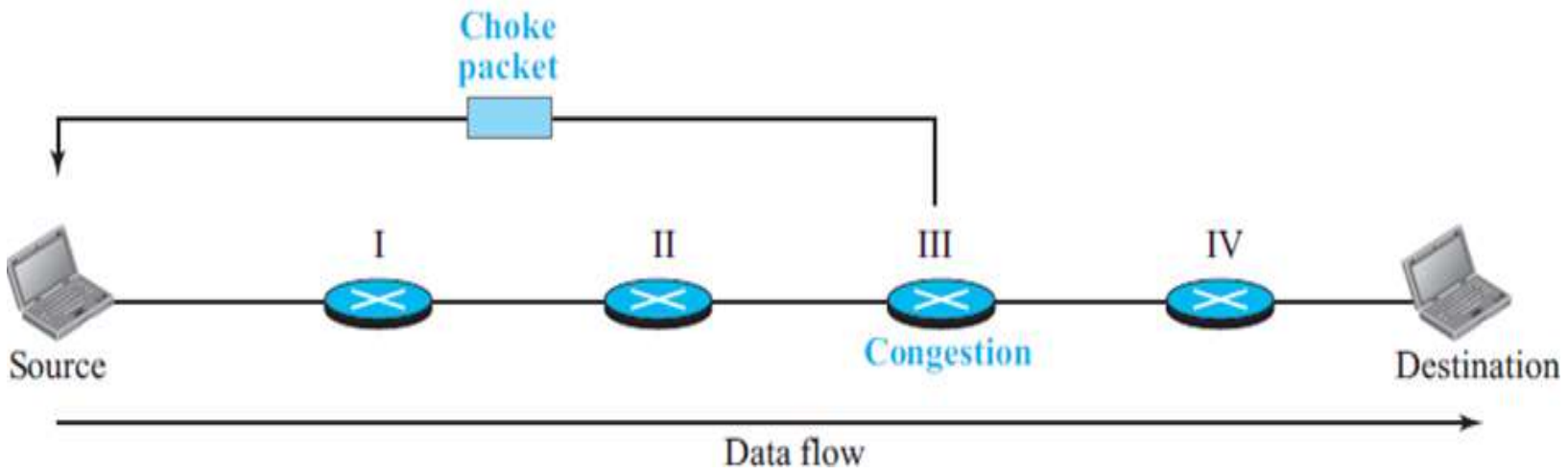
- **Closed-Loop Congestion Control** :Closed-loop congestion control mechanisms try to alleviate congestion after it happens.
- **Backpressure:**

Figure 18.14 *Backpressure method for alleviating congestion*



- **Choke Packet**

Figure 18.15 *Choke packet*



- **Implicit Signaling** In implicit signaling, there is no communication between the congested node or nodes and the source. The source guesses that there is congestion somewhere in the network from other symptoms.
- For example, when a source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested. The delay in receiving an acknowledgment is interpreted as congestion in the network; the source should slow down.
- **Explicit Signaling** The node that experiences congestion can explicitly send a signal to the source or destination. The explicit-signaling method, however, is different from the choke-packet method. In the choke-packet method, a separate packet is used for this purpose; in the explicit-signaling method, the signal is included in the packets that carry data. Explicit signaling can occur in either the forward or the backward direction. This type of congestion control can be seen in an ATM network

- **<u>QoS(Quality of Service)</u>** is nothing but a quality that every flow seeks to attain smooth movement without any congestion.
- **Techniques to Improve QoS**
- Some techniques that can be used to improve the quality of service. The four common methods: **scheduling, traffic shaping, admission control, and resource reservation.**
- **Scheduling:**Packets from different flows arrive at a switch or router for processing. A good scheduling technique treats the different flows in a fair and appropriate manner. Several scheduling techniques are designed to improve the quality of service. We discuss three of them here: FIFO queuing, priority queuing, and weighted fair queuing.

i. FIFO Queuing

- In first-in, first-out (FIFO) queuing, packets wait in a buffer (queue) until the node (router or switch) is ready to process them. If the average arrival rate is higher than the average processing rate, the queue will fill up and new packets will be discarded. A FIFO queue is familiar to those who have had to wait for a bus at a bus stop.

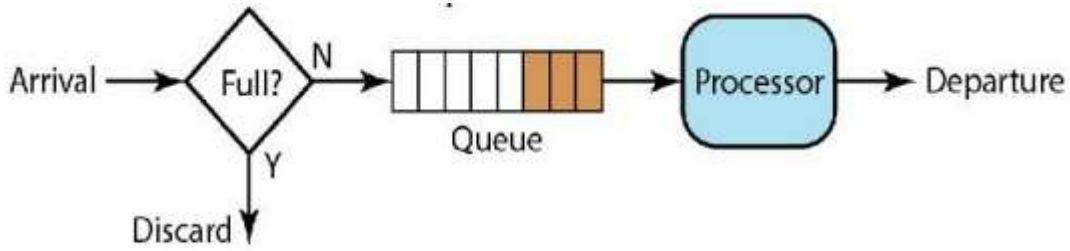


Figure 4.31 FIFO queue

ii. Priority Queuing

- In priority queuing, packets are first assigned to a priority class. Each priority class has its own queue. The packets in the highest-priority queue are processed first. Packets in the lowest- priority queue are processed last. Note that the system does not stop serving a queue until it is empty. Figure 4.32 shows priority queuing with two priority levels (for simplicity).
- A priority queue can provide better QoS than the FIFO queue because higher priority traffic, such as multimedia, can reach the destination with less delay. However, there is a potential drawback. If there is a continuous flow in a high-priority queue, the packets in the lower-priority queues will never have a chance to be processed. This is a condition called starvation

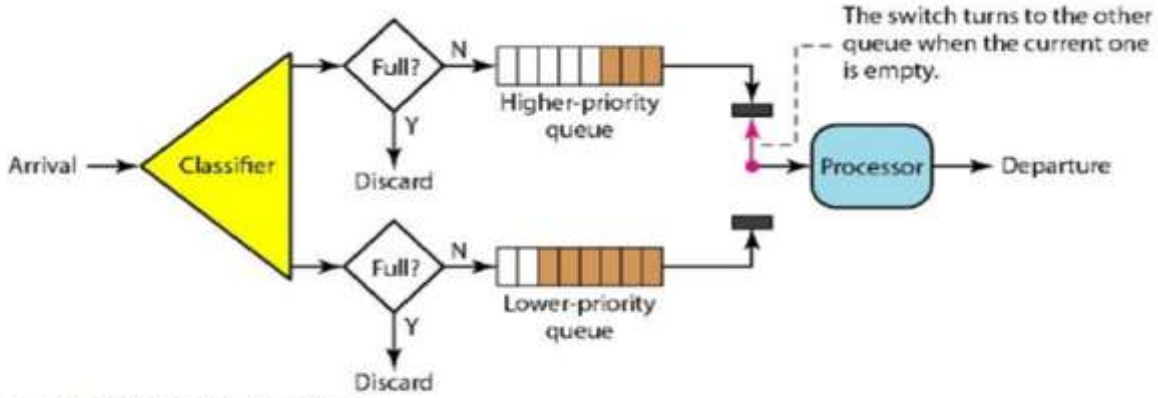


Figure 4.32 Priority queuing

iii. Weighted Fair Queuing

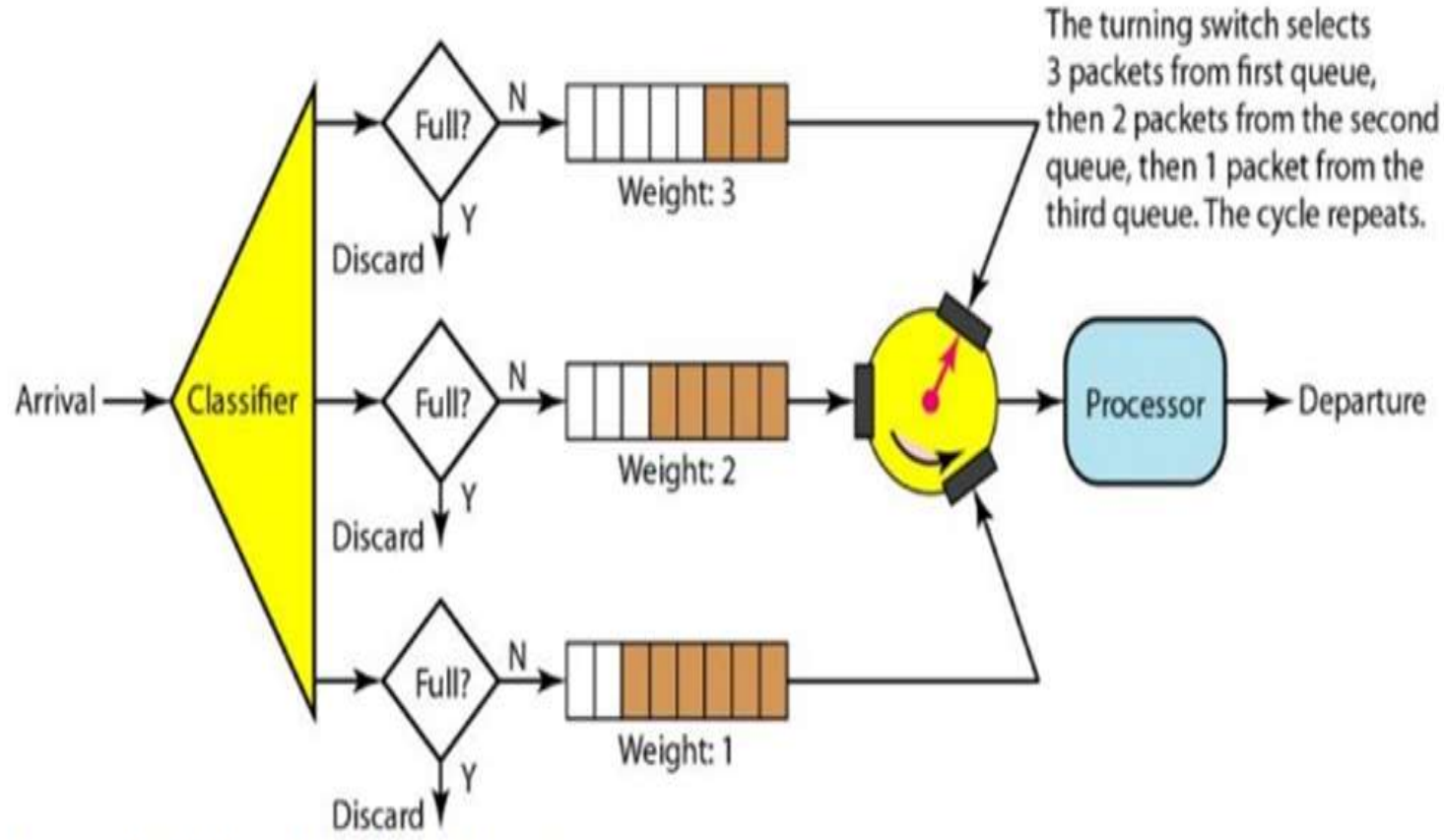


Figure 4.33 Weighted Fair Queuing

Leaky Bucket

- If a bucket has a small hole at the bottom, the water leaks from the bucket at a constant rate as long as there is water in the bucket. The rate at which the water leaks does not depend on the rate at which the water is input to the bucket unless the bucket is empty. 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. Figure 4.34 shows a leaky bucket and its effects.

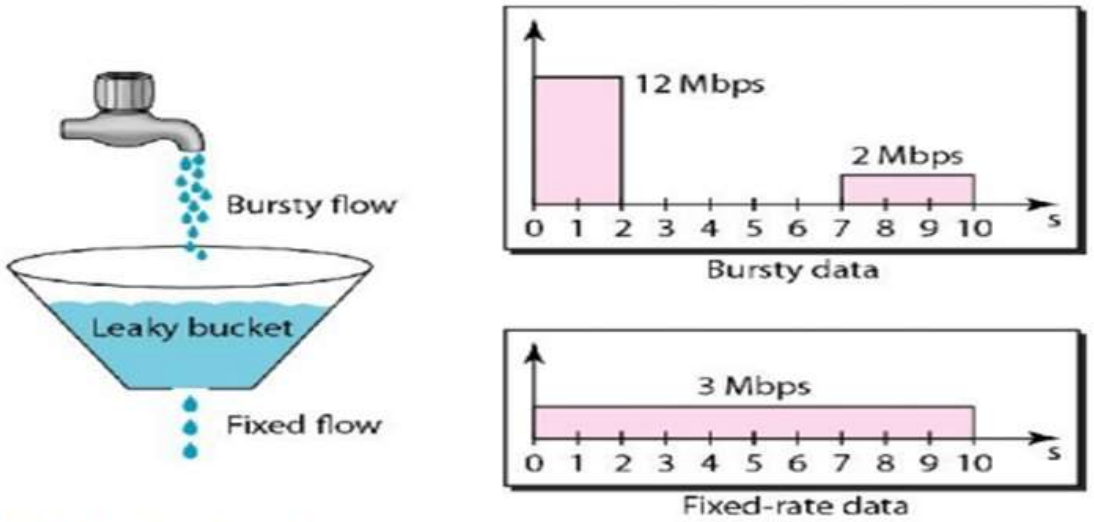


Figure 4.34 Leaky Bucket

- We assume that the network has committed a bandwidth of 3 Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment. In Figure 4.34 the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data. The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data. In all, the host has sent 30 Mbits of data in lOs. The leaky bucket smooth's the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

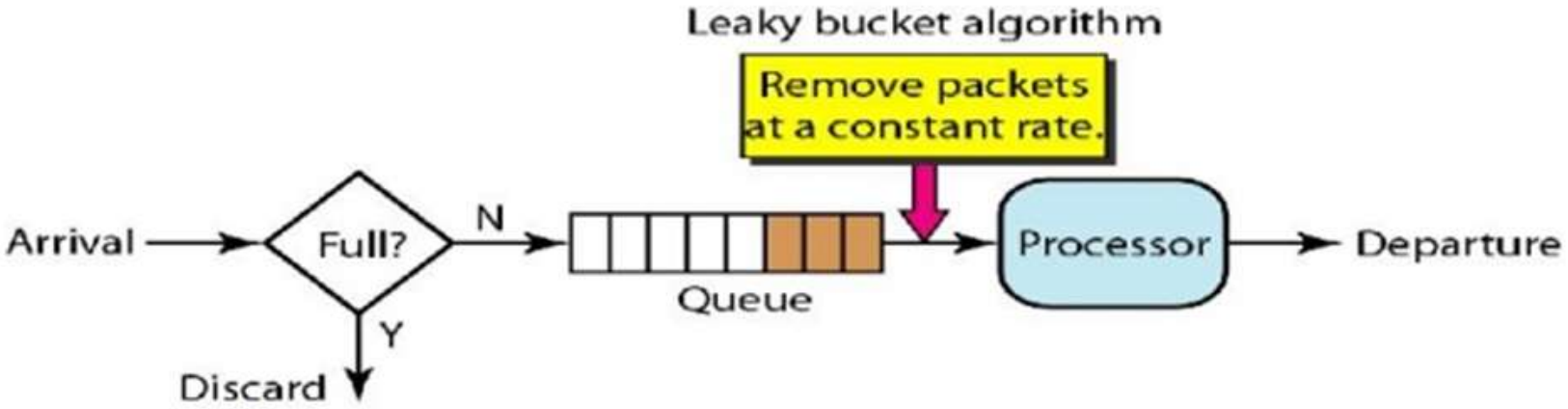


Figure 4.35 Leaky bucket implementation

Token Bucket

- The leaky bucket is very restrictive. It does not credit an idle host.
- For example, if a host is not sending for a while, its bucket becomes empty.
- Now if the host has bursty data, the leaky bucket allows only an average rate.
- The time when the host was idle is not taken into account.
- On the other hand, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens.
- For each tick of the clock, the system sends n tokens to the bucket. The system removes one token for every cell (or byte) of data sent. For example, if n is 100 and the host is idle for 100 ticks, the bucket collects 10,000 tokens.

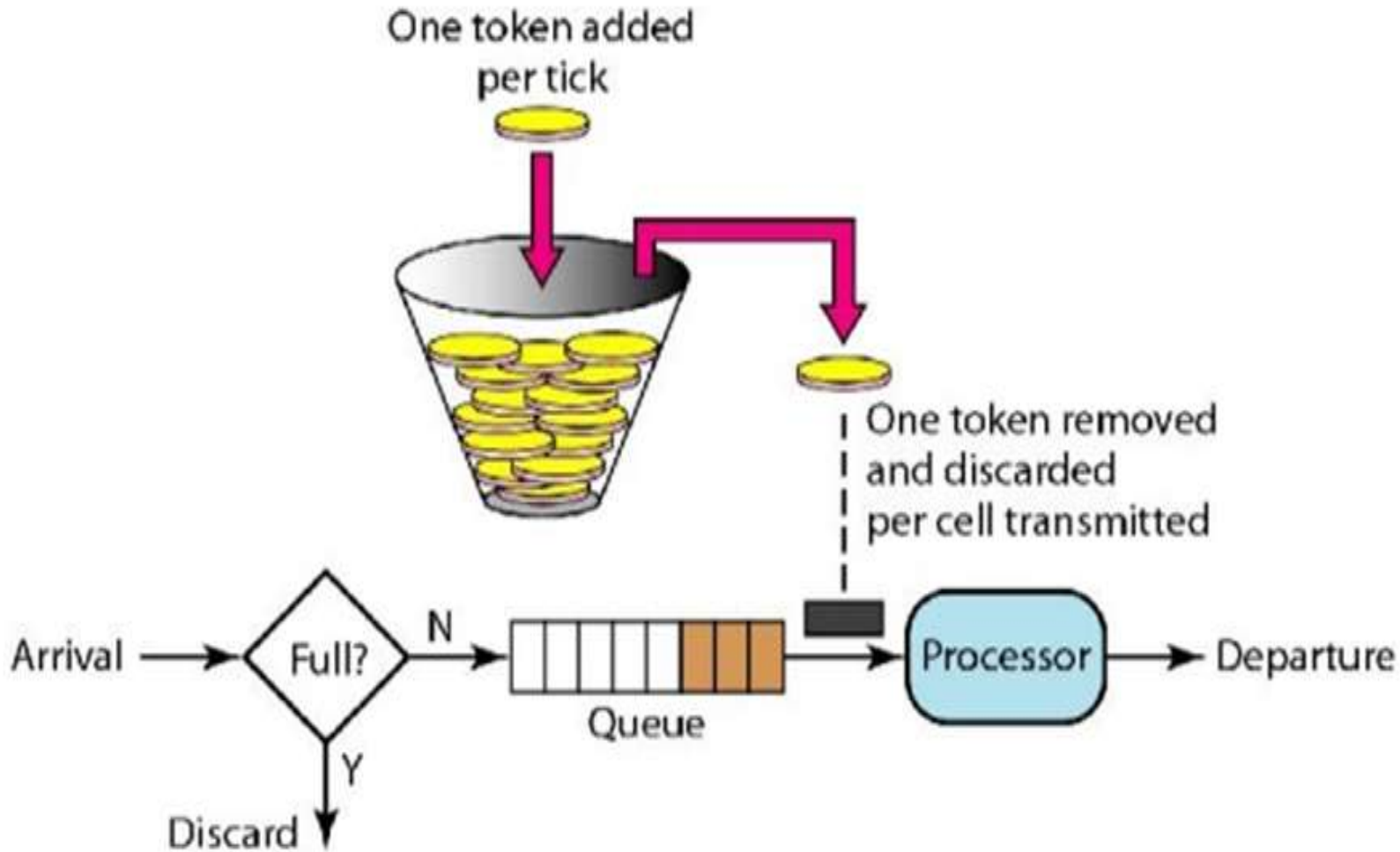


Figure 4.36 Token bucket

- The token bucket can easily be implemented with a counter. The token is initialized to zero. Each time a token is added, the counter is incremented by 1. Each time a unit of data is sent, the counter is decremented by 1. When the counter is zero, the host cannot send data.

INTERNETWORKING

- Internetworking is process or technique of connecting different networks by using intermediary devices such as routers or gateways devices. Internetworking ensures data communication among networks owned & operated by different entities using a common data communication & internet routing protocol
- Internetworking is a term used by CISCO. Any inter-connection among or between public, private or commercial, industrial or government. Computer may also defined as Internetwork or Internetworking The standard reference model for Internetworking is OSI/ ISO model.
- Two architectural models are commonly used to describe protocols & methods used in Internetworking. An internetwork is a collection of individual networks, connected by intermediate networking devices, that functions as a single large network.
- Internetworking refers other industry, products and procedures that meet the challenge of creating and administering internetworks.

History of Internetworking

- The first networks were time-sharing networks that used mainframes and attached terminals. Such environments were implemented by both IBM's System Network Architecture (SNA) and Digital's network architecture.
- Local area networks (LANs) evolved around the PC revolution. LANs enabled multiple users in a relatively small geographical area to exchange files and messages, as well as access shared resources such as file servers.
- Wide area networks (WANs) interconnect LANs across normal telephone lines (and other media), thereby interconnecting geographically dispersed users.

Today, high-speed LANs and switched internetworks are becoming widely used, largely because they operate at very high speeds and support such high-bandwidth applications as voice and video conferencing.

Internetworking evolved as a solution to three key problems:

- isolated LANs
- duplication of resources
- lack of network management.

- Isolated LANS made electronic communication between different offices or departments impossible.
- Duplication of resources meant that the same hardware and software had to be supplied to each office or department, as did a separate support staff.
- This lack of network management meant that no centralized method of managing and troubleshooting networks existed.

- There are three types in internetworking are as follows:
- **Extranet**: An Extranet is a network of internetwork or internetworking that is limited in scope to a single organization or entity but has limited connections to networks of one or more other. Usually, but not necessarily, trusted organization or entities. Extranet may also be categorized as a MAN, Wan or other type network
- **Intranet**: An intranet is a set of interconnected networks or internetworking using internet protocol & uses IP-based tools such as web browser & FTP tools that is under control of single administrative entity. A large Intranet will have its own web – server to provide users with information
- **Internet**: A specific internetworking consists of a worldwide inter-connection of government, academic, public &private networks based upon ARPANET (**Advanced Research Projects Agency Network**) developed by ARPA of U.S. department of defense also home to WWW (World wide Web) & referred as 'Internet'.

- **Internetworking Challenges**
- Implementing a functional internetwork is no simple task. Many challenges must be faced, especially in the areas of connectivity, reliability, network management, and flexibility. Each area is key in establishing an efficient and effective internetwork.
- The challenge when connecting various systems is to support communication between disparate technologies. Different sites, for example, may use different types of media, or they might operate at varying speeds.
- Another essential consideration, reliable service, must be maintained in any internetwork. Individual users and entire organizations depend on consistent, reliable access to network resources.
- Furthermore, network management must provide centralized support and troubleshooting capabilities in an internetwork.
- Configuration, security, performance, and other issues must be adequately addressed for the internetwork to function smoothly.
- Flexibility, the final concern, is necessary for network expansion and new applications and services, among other factors

The Network Layer in the Internet

- The Internet layer, also known as the **network layer** or **IP layer**, accepts and delivers packets for the network. This layer includes the powerful Internet Protocol (IP), the Address Resolution Protocol (ARP), and the Internet Control Message Protocol (ICMP).
- **Network layer:** The network layer works for the transmission of data from one host to the other located in different networks. It also takes care of packet routing i.e. selection of the shortest path to transmit the packet, from the number of routes available.
- **Why Network Layer in the Internet?**
- The Internet can be viewed as a collection of subnetworks or Autonomous Systems (AS). IP (Internet Protocol) hosts the whole Internet together.
- Communication in the Internet works as follows:
- The transport layer takes data streams and breaks them up into datagrams.
- Each datagram is transmitted through the Internet.
- When all the pieces finally get to the destination machine, they are reassembled by the network layer, which inserts it into the receiving process' input stream.

IP Protocol

- The IP protocol and its associated routing protocols are possibly the most significant of the entire TCP/IP suite. IP is responsible for the following:
- **IP addressing:** The IP addressing conventions are part of the IP protocol. Designing an IPv4 Addressing Scheme introduces IPv4 addressing and IPv6 Addressing Overview introduces IPv6 addressing.
- **Host-to-host communications:** IP determines the path a packet must take, based on the receiving system's IP address.
- **Packet formatting:** IP assembles packets into units that are known as **datagram's**. Datagram's are fully described in Internet Layer: Where Packets Are Prepared for Delivery.
- **Fragmentation:** If a packet is too large for transmission over the network media, IP on the sending system breaks the packet into smaller fragments. IP on the receiving system then reconstructs the fragments into the original packet. Oracle Solaris supports both IPv4 and IPv6 addressing formats, which are described in this book.

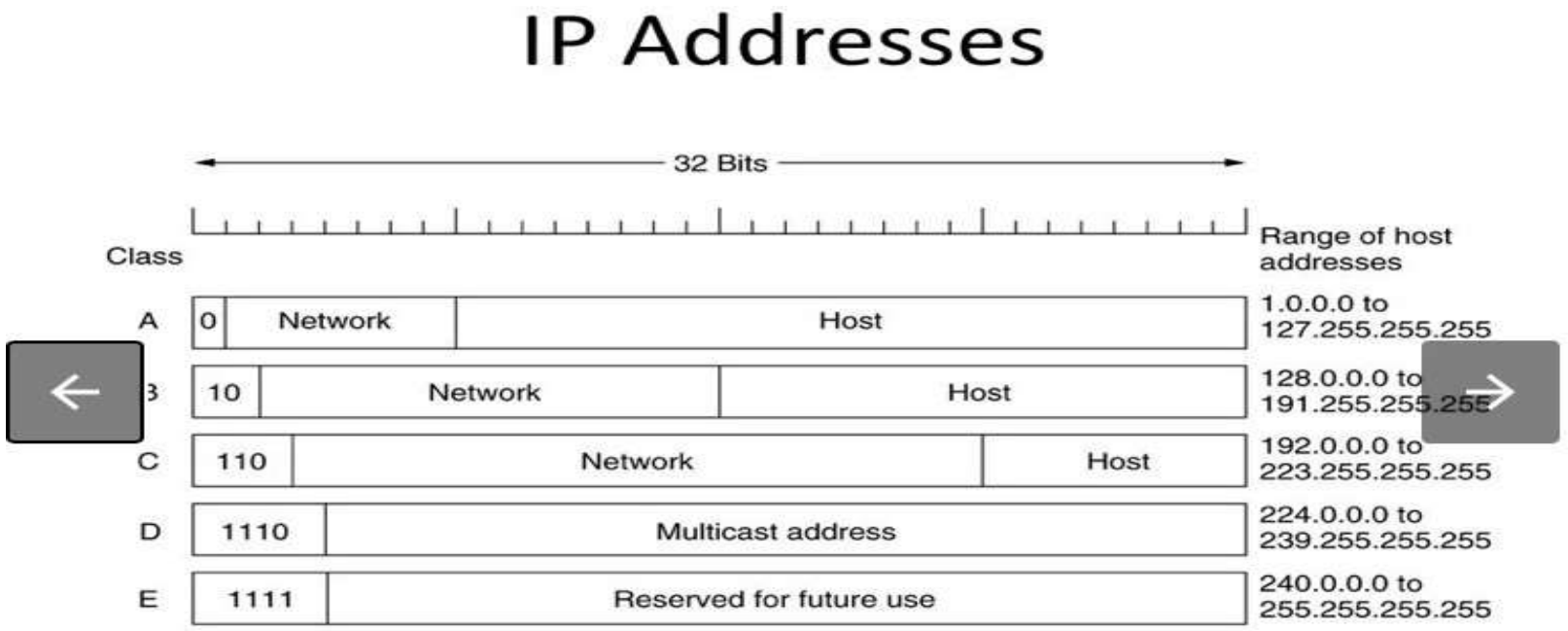
- To avoid confusion when addressing the Internet Protocol, one of the following conventions is used:
  - When the term “IP” is used in a description, the description applies to both IPv4 and IPv6.
  - When the term “IPv4” is used in a description, the description applies only to IPv4.
- When the term “IPv6” is used in a description, the description applies only to IPv6

ICMP Protocol

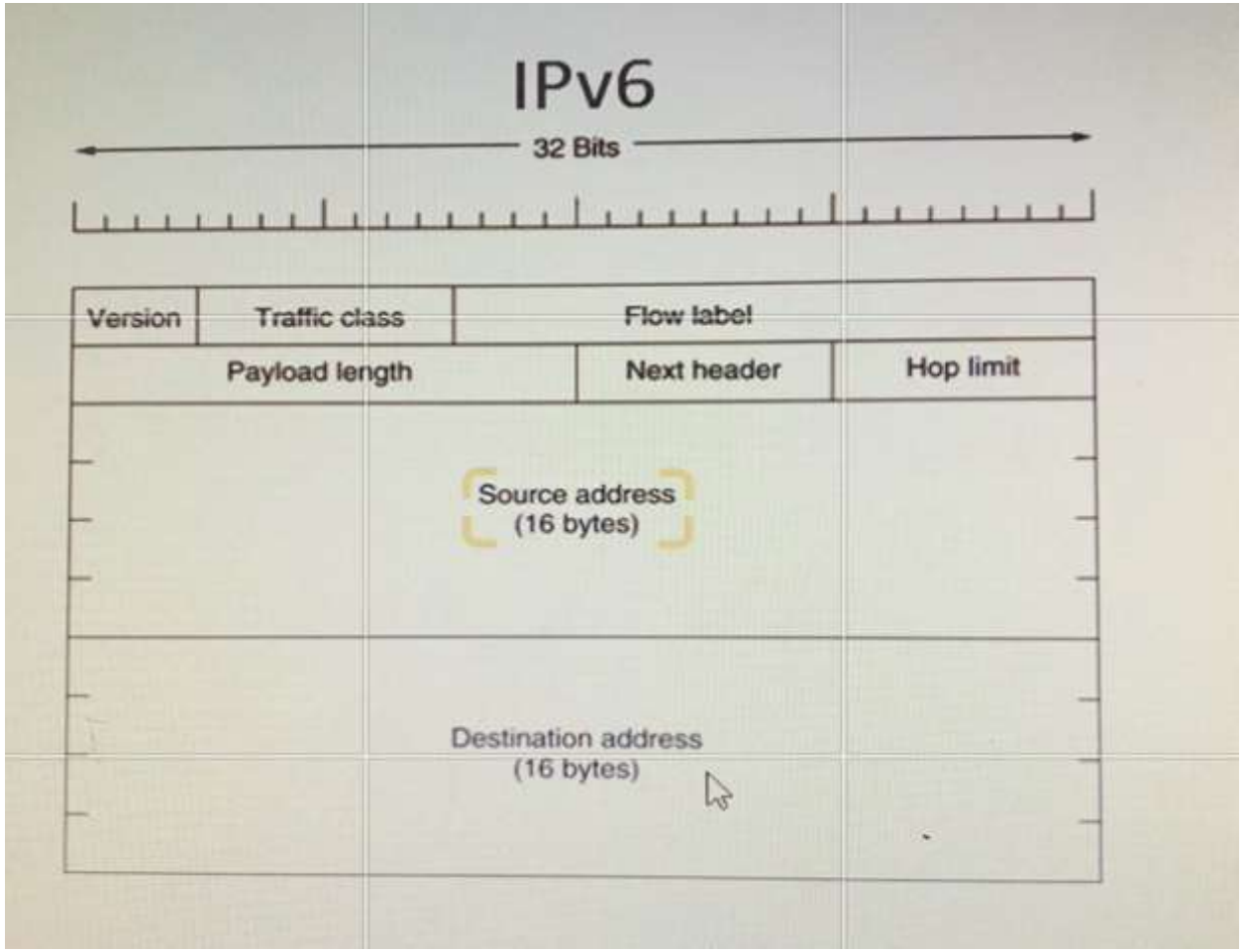
- The Internet Control Message Protocol (ICMP) detects and reports network error conditions. ICMP reports on the following:
- **Dropped packets** – Packets that arrive too fast to be processed
- **Connectivity failure** – A destination system cannot be reached
- **Redirection** – Redirecting a sending system to use another router

IP Addresses

- Traditionally, IP addresses were divided into the five categories: A, B, C, D, E. Network numbers are managed by a nonprofit corporation called ICANN (Internet Corporation for Assigned Names and Numbers) to avoid conflicts. Network address, which are 32-bit numbers, are usually written in dotted decimal notation.



- IPV6: The newest version of IP (version 6, or IPng) uses 128 bits, yielding. 2^128 unique combinations. IPv6 is slowly be integrated in the existing Internet. IPv4's 32 bits continues to be the dominant form of IP addressing



- **VERSION**. Indicates version of Internet Protocol which contains bit sequence 0110.
- **Traffic Class**. 8 bits. - The Traffic Class field indicates class or priority of IPv6 packet which is similar to Service Field in IPv4 packet. It helps routers to handle the traffic based on the priority of the packet. If congestion occurs on the router then packets with the least priority will be discarded.
- As of now, only 4-bits are being used (and the remaining bits are under research), in which 0 to 7 are assigned to Congestion controlled traffic and 8 to 15 are assigned to Uncontrolled traffic.
- Priority assignment of Congestion controlled traffic:

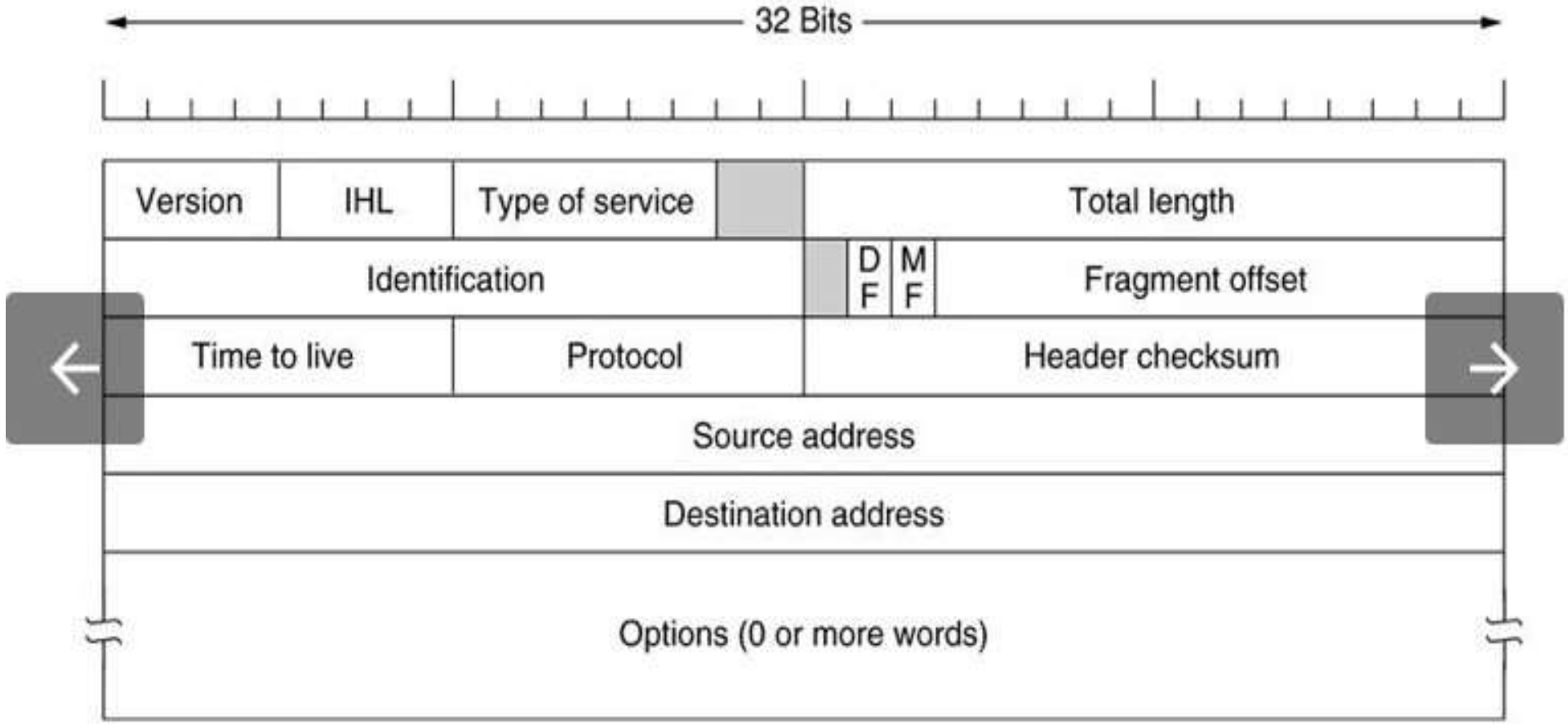
Priority	Meaning
0	No Specific traffic
1	Background data
2	Unattended data traffic
3	Reserved
4	Attended bulk data traffic
5	Reserved
6	Interactive traffic
7	Control traffic

- **Flow Label**. 20 bits. - Flow Label field is used by a source to label the packets belonging to the same flow in order to request special handling by intermediate IPv6 routers, such as non-default quality of service or real-time service. In order to distinguish the flow, an intermediate router can use the source address, a destination address, and flow label of the packets. Between a source and destination, multiple flows may exist because many processes might be running at the same time. Routers or Host that does not support the functionality of flow label field and for default router handling, flow label field is set to 0. While setting up the flow label, the source is also supposed to specify the lifetime of the flow.
- **Payload Length**. It is a 16-bit (unsigned integer) field, indicates the total size of the payload which tells routers about the amount of information a particular packet contains in its payload. The payload Length field includes extension headers(if any) and an upper-layer packet. In case the length of the payload is greater than 65,535 bytes (payload up to 65,535 bytes can be indicated with 16-bits), then the payload length field will be set to 0 and the jumbo payload option is used in the Hop-by-Hop options extension header. .
- **Next Header(8-bit)**. Next Header indicates the type of extension header(if present) immediately following the IPv6 header. Whereas In some cases it indicates the protocols contained within upper-layer packets, such as TCP, UDP. .

Order	Header Type	Next Header Code
1	Basic IPV6 Header	
2	Hop by Hop Option	0
3	Destination Options (with Routing)	60
4	Routing Header	43
5	Fragmentation Header	44
6	Authentication Header	51
7	Encapsulation Security Payload Header	50
8	Destination Options	60
9	Mobility Header	135
	No Next Header	59
Data	TCP	6
Data	UDP	17
Data	ICMPV6	58

- **Hop Limit**. 8 bits, unsigned. - Hop Limit field is the same as TTL in IPv4 packets. It indicates the maximum number of intermediate nodes IPv6 packet is allowed to travel. Its value gets decremented by one, by each node that forwards the packet and the packet is discarded if the value decrements to 0. This is used to discard the packets that are stuck in an infinite loop because of some routing error.
- **Source address**. 128 bits. - Source Address is the 128-bit IPv6 address of the original source of the packet.
- **Destination address**. **(128-bits):** The destination Address field indicates the IPv6 address of the final destination (in most cases). All the intermediate nodes can use this information in order to correctly route the packet.

The IP Header -V4



- **VERSION:** Version of the IP protocol (4 bits), which is 4 for IPv4
- **HLEN:** IP header length (4 bits), which is the number of 32 bit words in the header. The minimum value for this field is 5 and the maximum is 15.
- **Type of service:** Low Delay, High Throughput, Reliability (8 bits)
- **Total Length:** Length of header + Data (16 bits), which has a minimum value 20 bytes and the maximum is 65,535 bytes.
- **Identification:** Unique Packet Id for identifying the group of fragments of a single IP datagram (16 bits)
- **Flags:** 3 flags of 1 bit each : reserved bit (must be zero), do not fragment flag, more fragments flag (same order)
- **Fragment Offset:** Represents the number of Data Bytes ahead of the particular fragment in the particular Datagram. Specified in terms of number of 8 bytes, which has the maximum value of 65,528 bytes.

- **Time to live:** Datagram's lifetime (8 bits), It prevents the datagram to loop through the network by restricting the number of Hops taken by a Packet before delivering to the Destination.
- **Protocol:** Name of the protocol to which the data is to be passed (8 bits)
- **Header Checksum:** 16 bits header checksum for checking errors in the datagram header
- **Source IP address:** 32 bits IP address of the sender
- **Destination IP address:** 32 bits IP address of the receiver
- **Option:** Optional information such as source route, record route. Used by the Network administrator to check whether a path is working or not.