Unit-4 The Network Layer

4.1 Logical Addressing
- 4.1.1 IP v4 Addresses - Address Space - Classful Addressing - Classless Addressing

4.2. Routing Algorithm
- 4.2.1. Shortest Path
- 4.2.2. Multicast Routing

4.3. Congestion Control
- 4.3.1. Introduction to Congestion Control

4.3.2. Deadlocks

The network layer (Internet layer) provides services to the transport layer. It can be based on either virtual circuits or datagrams. In both cases, its main job is routing packets from the source to the destination. In virtual circuit subnets, a routing decision is made when the virtual circuit is set up. In datagram subnets, it is made on every packet.

Functions of Network Layer (Internet layer)

- **Path determination:** route taken by packets from source to destination (Routing Algorithm).
- **Forwarding:** more packets from router‘s input to appropriate router output.
- **Call setup:** some n/w architectures require router cell setup along the path before data flows.

4.1 Logical Addressing

A logical address, also known as an IP (Internet Protocol) address, is a unique identifier assigned to each device on a network. It is used to identify the location of a device and enable routing of data packets from the source to the destination across different networks. Logical addresses are essential for ensuring that data packets reach the correct destination, even if they have to pass through multiple routers and networks.

4.1.1. IP v4 Addresses

An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a device (for example, a computer or a router) to the Internet. IPv4 addresses are unique. They are unique in the sense that each address defines one, and only one, connection to the Internet. Two devices on the Internet can never have the same address at the same time. The IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet. The address space of IPv4 is 2^{32} or 4,294,967,296.

IPv4 is 32-bit addressing scheme used as TCP/IP host addressing mechanism. IP addressing enables every host on the TCP/IP network to be uniquely identifiable.

IPv4 provides hierarchical addressing scheme which enables it to divide the network into sub-networks, each with well-defined number of hosts. IP addresses are divided into many categories:

- ✓ **Class A** - it uses first octet for network addresses and last three octets for host addressing
- ✓ **Class B** - it uses first two octets for network addresses and last two for host addressing
- ✓ **Class C** - it uses first three octets for network addresses and last one for host addressing
- ✓ **Class D** - it provides flat IP addressing scheme in contrast to hierarchical structure for above three.
- ✓ **Class E** - It is used as experimental.

A) Address Space

A protocol such as IPv4 that defines addresses has an address space. An address space is the total number of addresses used by the protocol. If a protocol uses *N* bits to define an address, the address space is 2^N because each bit can have two different values (0 or 1) and *N* bits can have 2^N values.

IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than 4 billion). This means that, theoretically, if there were no restrictions, more than 4 billion devices could be connected to the Internet.

B) Classful Addressing

IPv4 addressing,used the concept of classes. This architecture is called classful addressing. In classful addressing, the address space is divided into five classes: A, B, C, D, and E.

We can find the class of an address when given the address in binary notation or dotted-decimal notation. If the address is given in binary notation, the first few bits can immediately tell us the class of the address. If the address is given in decimal-dotted notation, the first byte defines the class. Both methods are shown in following Figure-1.

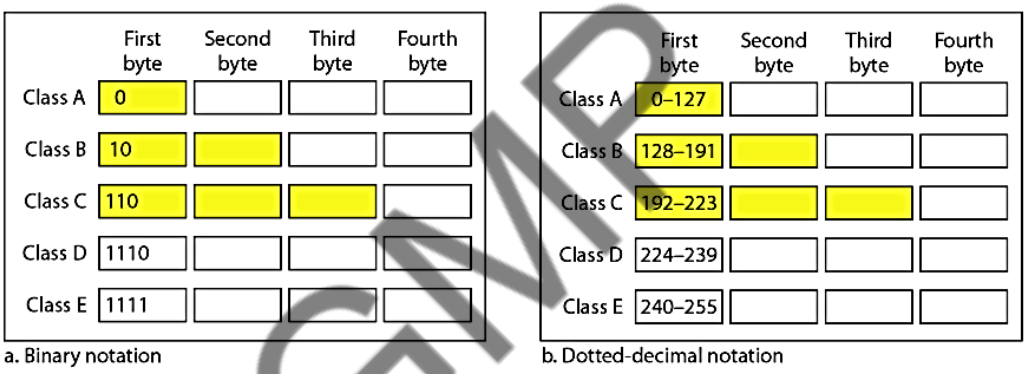


Figure-1: Finding the classes in binary and dotted-decimal notation.

Example:Find the class of each address.	**Solution:**
a)00000001 00001011 00001011 11101111	a) First bit is 0. The class is A.
b)11000001 10000011 00011011 11111111	b) First 2 bits are 1; third bit is 0. The class is C.
c) 14.23.120.8	c) First byte is 14 (between 0 and 127); the class is A.
d)252.5.15.111	d) First byte is 252 (between 240 and 255); the class is E.

a) Classes and Blocks

One problem with classful addressing is that each class is divided into a fixed number of blocks with each block having a fixed size as shown in following Table.

Table-1: Number of blocks and block size in classful IPv4 addressing

Class	Number of Blocks	Block Size	Application
A	**128**	**16,777,216**	**Unicast**
B	**16,384**	**65,536**	**Unicast**
C	**2,097,152**	**256**	**Unicast**
D	**1**	**268,435,456**	**Multicast**
E	**1**	**268,435,456**	**Reserved**

Class A addresses were designed for large organizations with a large number of attached hosts or routers. **Class B** addresses was designed for midsize organizations with tens of thousands of attached hosts or routers. **Class C** addresses were designed for small organizations with a small number of attached hosts or routers. **Class D** addresses were designed for multicasting. The C**lass E** addresses were reserved for future use. In c1assful addressing, a large part of the available addresses were wasted.

b)Netid and Hostid

In classful addressing, an IP address in class A, B, or C is divided into netid and hostid.

These parts are of varying lengths, depending on the class of the address. The netid is in color, the hostid is in white. Note that the concept does not apply to classes D and E. In class A, one byte defines the netid and three bytes define the hostid. In class B, two bytes define the netid and two bytes define the hostid. In class C, three bytes define the netid and one byte defines the hostid.

c) Mask

Although the length of the netid and hostid (in bits) is predetermined in classful addressing, we can also use a mask (also called the default mask), a 32-bit number made of contiguous 1's followed by contiguous 0's. The masks for classes A, B, and C are shown in Table-2. The concept does not apply to classes D and E.

Table-2: Default masks for classful addressing.

Class	*Binary*	*Dotted-Decimal*	*CIDR*
A	**11111111** 00000000 00000000 00000000	**255**.0.0.0	/8
B	**11111111 11111111** 00000000 00000000	**255.255**.0.0	/16
C	**11111111 11111111 11111111** 00000000	**255.255.255**.0	/24

The mask can help us to find the netid and the hostid. For example, the mask for a class A address has eight 1s, which means the first 8 bits of any address in class A define the netid; the next 24 bits define the hostid. The last column of Table shows the mask in the form */n* where *n* can be 8, 16, or 24 in classful addressing. This notation is also called slash notation orClassless Inter-domain Routing (CIDR) notation. The notation is used in classless addressing,

d)Subnetting If an organization was granted a large block in class A or B, it could divide the addressesinto several contiguous groups and assign each group to smaller networks (called subnets). Subnetting increases the number of 1's in the mask.

e)Supernetting

In supernetting, an organization can combine several class C blocks to create a larger range of addresses. In other words, several networks are combined to create a supernetwork or a supemet. An organization can apply for a set of class C blocks instead of just one.

f) Address Depletion

The flaws in classful addressing scheme combined with the fast growth of the Internet led to the near depletion of the available addresses. Yet the number of devices on the Internet is much less than the 232 address space. We have run out of class A and B addresses, and a class C block is too small for most midsize organizations. One solution that has alleviated the problem is the idea of classless addressing.

C) Classless Addressing

To overcome address depletion and give more organizations access to the Internet, classless addressing was designed and implemented. In this scheme, there are no classes, but the addresses are still granted in blocks.

a) **Address Blocks**

In classless addressing, when an entity, small or large, needs to be connected to the Internet, it is granted a block (range) of addresses. The size of the block varies based on the nature and size of the entity. For example, a household may be given only two addresses; a large organization may be given thousands of addresses. An ISP, as the Internet service provider, may be given thousands or hundreds of thousands based on the number of customers it may serve.

Restrictions to simplify the handling of addresses, the Internet authorities impose three restrictions on classless address blocks:

1. The addresses in a block must be contiguous, one after another.
2. The number of addresses in a block must be a power of 2 (1, 2, 4, 8 ...).
3. The first address must be evenly divisible by the number of addresses.

Example:

Figure-2 shows a block of addresses, in both binary and dotted-decimal notation, granted to a small business that needs 16 addresses.

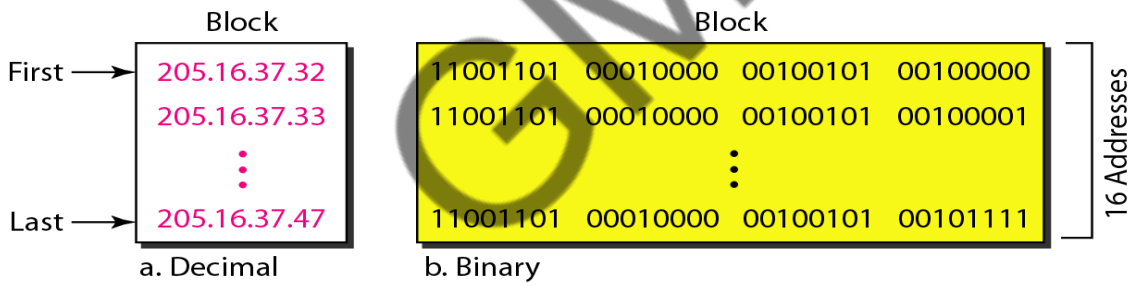


Figure-2: A block of 16 addresses granted to small business.

We can see that the restrictions are applied to this block. The addresses are contiguous. The number of addresses is a power of 2 (16 = 24), and the first address is divisible by 16. The first address, when converted to a decimal number, is 3,440,387,360, which when divided by 16 results in 215,024,210.

b) **Mask**

A better way to define a block of addresses is to select any address in the block and the mask. As we discussed before, a mask is a 32-bit number in which the n leftmost bits are 1s and the 32 - n rightmost bits are 0s. However, in classless addressing the mask for a block can take any value from 0 to 32. It is very convenient to give just the value of n preceded by a slash.

First Address: The first address in the block can be found by setting the 32 - n rightmost bits inthe binary notation of the address to 0s. The first address in the block can be found by setting the rightmost 32 - n bits to 0s.

Example:A block of addresses is granted to a small organization. We know that one of the addresses is 205.16.37.39/28. What is the first address in the block?

Solution:

The binary representation of the given address is 11001101 00010000 00100101 00100 1 11. If we set 32 - 28 rightmost bits to 0, we get11001101 000100000100101 0010000 or 205.16.37.32.

Last Address: The last address in the block can be found by setting the 32 - n rightmost bits inthe binary notation of the address to 1s. The last address in the block can be found by setting the rightmost 32 - n bits to 1s.

Example: Find the last address for the block in addresses is 205.16.37.39/28.

Solution:

The binary representation of the given address is 11001101 000100000010010100100111. If we set 32 - 28 rightmost bits to 1, we get 11001101 00010000 001001010010 1111 or 205.16.37.47.

Number of Addresses: The number of addresses in the block is the difference between the lastand first address. It can easily be found using the formula 232- n.

Example: Find the number of addresses in Example addresses is 205.16.37.39/28.

Solution: The value of n is 28, which means that number of addresses is 232- 28 or 16.

Example: Another way to find the first address, the last address, and the number of addresses is to represent the mask as a 32-bit binary (or 8-digit hexadecimal) number. This is particularly useful when we are writing a program to find these pieces of information. For address 205.16.37.39/28, the /28 can be represented as 11111111 11111111 11111111 11110000 (twenty-eight 1s and four 0s).

Find

a. The first address

b. The last address

c. The number of addresses

Solution:

a. The first address can be found by ANDing the given addresses with the mask. ANDing here is done bit by bit. The result of ANDing 2 bits is 1 if both bits are 1s; the result is 0 otherwise.

Address: 11001101 00010000 00100101 00100111
Mask: 11111111 11111111 11111111 11110000
First address: 11001101 00010000 00100101 00100000

b. The last address can be found by ORing the given addresses with the complement of the mask. ORing here is done bit by bit. The result of ORing 2 bits is 0 if both bits are 0s; the result is 1 otherwise. The complement of a number is found by changing each 1 to 0 and each 0 to 1.

Address: 11001101 00010000 00100101 00100111
Mask complement: 00000000 00000000 00000000 00001111
Last address: 11001101 00010000 00100101 00101111

c. The number of addresses can be found by complementing the mask, interpreting it as a decimal number, and adding 1 to it.

Mask complement: 000000000 00000000 00000000 00001111

Number of addresses: 15 + 1 =16

c)Network Addresses

A very important concept in IP addressing is the network address. When an organizationis given a block of addresses, the organization is free to allocate the addresses to the devices that need to be connected to the Internet. The first address in the class, however, is normally (not always) treated as a special address. The first address is called the network address and defines the organization network. It defines the organization itself to the rest of the world.

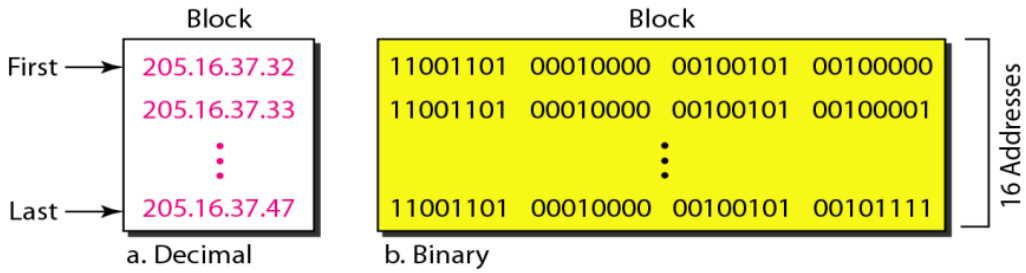


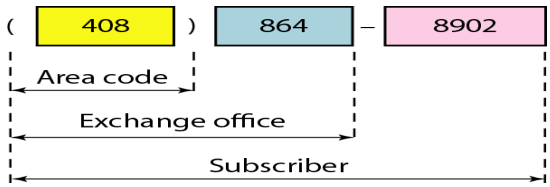
Figure: A network configuration for the block 205.16.37.32/38

The organization network is connected to the Internet via a router. The router has two addresses. One belongs to the granted block; the other belongs to the network that is at the other side of the router. We call the second address x.y.z.t/n because we do not know anything about the network it is connected to at the other side. All messages destined for addresses in the organization block (205.16.37.32 to 205.16.37.47) are sent, directly or indirectly, to x.y.z.t/n. We say directly or indirectly because we do not know the structure of the network to which the other side of the router is connected. The first address in a block is normally not assigned to any device; it is used as the network address that represents the organization to the rest of the world.

d) Hierarchy

IP addresses, like other addresses or identifiers we encounter these days, have levels of hierarchy. For example, a telephone network in North America has three levels of hierarchy. The leftmost three digits define the area code, the next three digits define the exchange, the last four digits define the connection of the local loop to the central office. Figure-3 shows the structure of a hierarchical telephone number

Figure-3: Hierarchy in a telephone network in North America.



Two-Level Hierarchy: No Subnetting

An IP address can define only two levels of hierarchy when not subnetted. The n leftmost bits of the address x.y.z.t/n define the network (organization network); the 32 – n rightmost bits define the particular host (computer or router) to the network. The two common terms are prefix and suffix. The part of the address that defines the network is called the prefix; the part that defines the host is called the suffix. Figure-4 shows the hierarchical structure of an IPv4 address.

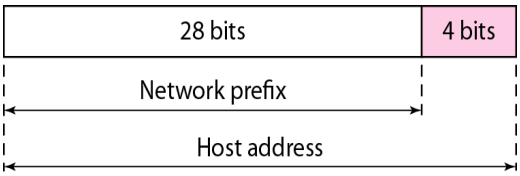


Figure-4: Two levels of hierarchy in an IPv4 address.

The prefix is common to all addresses in the network; the suffix changes from one device to another. Each address in the block can be considered as a two-level hierarchical structure: the leftmost n bits (prefix) define the network; the rightmost 32 - n bits define the host.

Note that applying the mask of the network, /26 to any of the addresses gives us the network address 17.12.14.0/26. We leave this proof to the reader. We can say that through subnetting, we have three levels of hierarchy. Note that in our example, the subnet prefix length can differ for the subnets as shown in Figure-5.

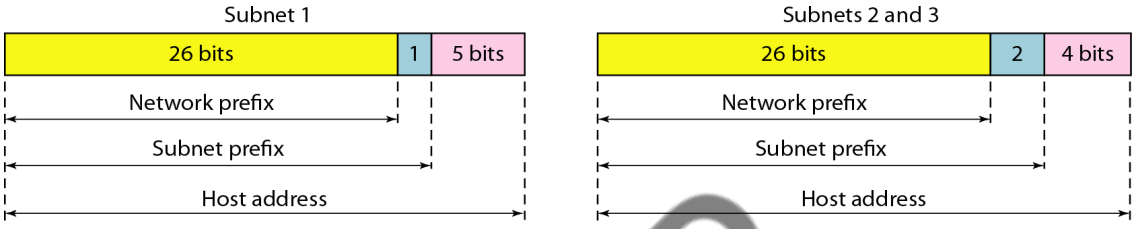


Figure-5: Three-levels of hierarchy in an IPv4 address.

More Levels of Hierarchy

The structure of classless addressing does not restrict the number of hierarchical levels. An organization can divide the granted block of addresses into subblocks. Each subblock can in turn be divided into smaller subblocks. And so on. One example of this is seen in the ISPs. A national ISP can divide a granted large block into smaller blocks and assign each of them to a regional ISP. A regional ISP can divide the block received from the national ISP into smaller blocks and assign each one to a local ISP. A local ISP can divide the block received from theregional ISP into smaller blocks and assign each one to a different organization. Finally, an organization can divide the received block and make several subnets out of it.

4.2. Routing Algorithm

In order to transfer the packets from source to the destination, the network layer must determine the best route through which packets can be transmitted.The best route is the route that has the "least-cost route/path" from source to the destination.

Routing is the process of forwarding the packets from source to the destination but the best route to send the packets is determined by the routing algorithm.

Routing performed by layer 3 (or network layer) devices to deliver the packet by choosing an optimal path from one network to another. It is an autonomous process handled by the network devices to direct a data packet to its intended destination. The node here refers to a network device called **Router**.

Whether the network layer provides datagram service or virtual circuit service, the main job of the network layer is to provide the best route. The routing protocol provides this job. The routing protocol is a routing algorithm.

Routing algorithms are software programs that implement different routing protocols (set of rules). They work by assigning a cost number to each link. The cost number is calculated using various network metrics. Every router tries to forward the data packet to the next best link with the lowest cost.

There are many different routing algorithms, each with its own strengths and weaknesses. Selecting the right routing algorithm for a given network is a critical task, as the algorithm can have a significant impact on the performance of the network.

4.2.1. Shortest Path

Shortest Path Algorithm is feasible and easy to understand routing algorithms.The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (often called a link). To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.The shortest path is the one that minimizes the sum of the weights of the edges traversed between the source and destination nodes. There are several algorithms designed to solve the shortest path problem, with Dijkstra's algorithm and the Bellman-Ford algorithm being among the most prominent.

Dijkstra's algorithm is based on greedy approach also known as the least cost path approach. The working of Dijkstra's algorithm is as follows:

1) Select the source node as the start node (S).

2) Mark the direct neighbour nodes as tentative nodes (Initially all the nodes are considered as tentative nodes).

3) Choose the node from tentative nodes list with lowest cost from the source node and mark it as permanent nodes and make it as source node.

4) If, the destination node is covered or there is no more nodes in the tentative list(i.e. no more nodes to be explored) then stop, otherwise go to step number 2.

Example: Consider the following Figure-6 and apply Dijkstra's routing algorithm for finding the best path or the shortest path fromsource node A tothe destination node E.

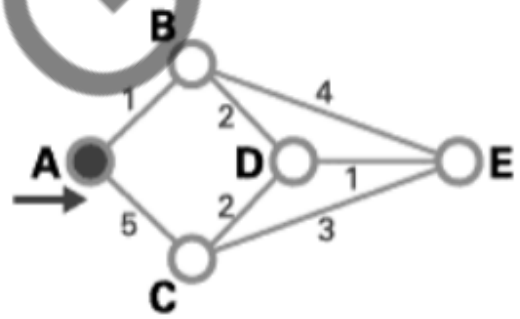


Figure-6: Shortest path using Dijkstra's routing algorithm.

The steps followed are as follows:

1) Node A is selected as source node.

2) Direct neighbour nodes B and C are marked as tentative node.

3) Node B has the lowest cost path (cost 1) from source node A, so it is marked as permanent node.

4) Node B is not the destination node and node D, E are available nodes in the tentative nodes lists,

5) So, Make node B as the source node now and explore the direct neighbours of it.

6) Nodes D and E are the direct neighbours of B.

7) Node D has minimum cost path from B so, node D is selected and marked as permanent node and D is not the destination node and also the tentative list is not empty.

8) Make D as the source node now and explore its direct neighbours.

9) Node C and E are the direct neighbours of D.

10) Node E has the less path cost than C from node D, so E is marked as permanent node. Node E is destination node, so stop here.

11) The shortest path from A to E is: **A – B – D – E**.

4.2.2. Multicast Routing

In the Internet at many instances there is a need to send same information to a group of clients at the same time. In such cases if, the unicast routing is used the data has to send to individual client and the server has to connect with each client independently to send the same information leading to overburden the server as well as the network. Instead of this if broadcasting is used to send the information, even it is also a wastage of resources computing as well as networking as not all the clients are interested in sent information or sometimes the information is not supposed to be disclosed to them. Hence, in both situations broadcasting approach is not suitable. **Multicast routing algorithm is used to handle such cases (sending a message to a group of users/ clients).** In multicast routing **the most important part is the group management.** Under group management tasks like group creation, deletion and management of membership to the group (like join and leave)are performed. When a host joins/leaves a group, the information is propagated to its router. Multicast routing is shown in Figure-7.

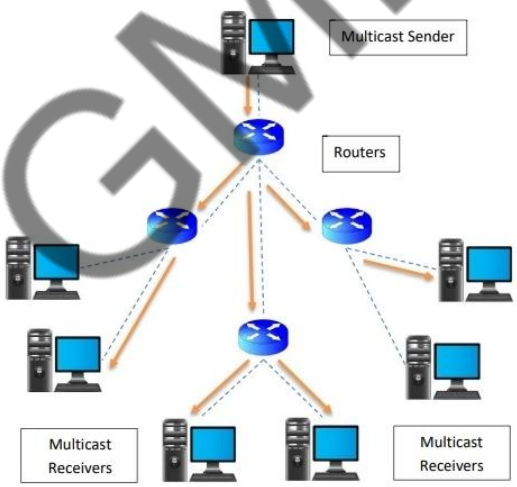


Figure-7: Multicast Routing

A router may be a member of one group, many groups or not a member of any group. While performing routing to send a message to a group, router requires the information about the members of the group. The multicast routing algorithm has to maintain the information of group membership. This information can be maintained and propagated in two ways: either the host informs to router about their membership, or routers send a query to their hosts periodically. Each router periodically shares their group management information to their neighbours, and like this the information propagated through the subnet.

In addition to group management, a logical spanning tree is constructed of the topology to perform the multicast routing. Once the spanning tree is constructed pruning is performed for each of the group. For

there may exist more than one spanning tree of a graph, so each node will construct its own spanning tree. Once the spanning tree is constructed it is then pruned for a group.

Pruning is a technique to preserve links connecting hosts that are member of the group only. One of the methods of pruning the spanning tree can be: starting from the last node of the path and moving towards the root, remove all routers that do not belong to the group under consideration.

Let's consider an example to understand the working of multicast routing. Here, there are two groups 1 and 2. Some of the hosts belong to group 1 and some to group 2 and some belongs to both 1 and 2 simultaneously. Here to perform multicast routing, each router constructs a spanning tree covering all other routers. Figure-8(a) shows network containing two groups i.e., 1 and 2 with three routers R1, R2, R3.

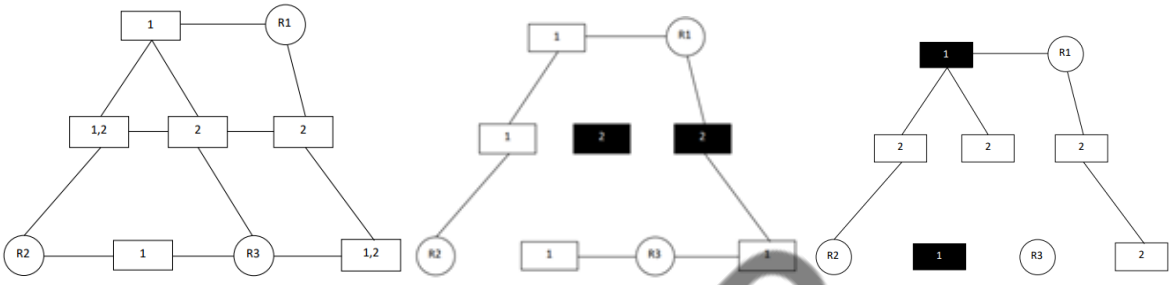


Figure-8(a) **Figure-8(b)** **Figure-8(c)**

Figure-8(b) shows the pruned spanning tree for group 1 for the spanning tree constructed for R1 above. Similarly, the pruned spanning tree for the group 2 of spanning tree of router R1 is as shown in Figure-8(c).

After pruning is completed, the multicast packets are forwarded only along the corresponding spanning tree. As the basic requirement of this algorithm is to store separate pruned spanning tree for each member of every group. Hence this method is not suitable for large networks.

Applications: Multicasting is used in many areas like:

1. Internet protocol (IP)
2. Streaming Media
3. It also supports video conferencing applications and webcasts.

4.3. Congestion Control

Congestion in a computer network happens when there is too much data being sent at the same time, causing the network to slow down. Just like traffic congestion on a busy road, network congestion leads to delays and sometimes data loss. When the network can't handle all the incoming data, it gets "clogged," making it difficult for information to travel smoothly from one place to another.

4.3.1. Introduction to Congestion Control Uses of Congestion control in Computer Network

Improved Network Stability: Congestion control helps keep the network stable by preventing it from getting overloaded. It manages the flow of data so the network doesn't crash or fail due to too much traffic.

Reduced Latency and Packet Loss: Without congestion control, **data transmission** can slow down, causing delays and data loss. Congestion control helps manage traffic better, reducing these delays and ensuring fewer data packets are lost, making data transfer faster and the network more responsive.

Enhanced Throughput: By avoiding congestion, the network can use its resources more effectively. This means more data can be sent in a shorter time, which is important for handling large amounts of data and supporting high-speed applications.

Fairness in Resource Allocation: Congestion control ensures that network resources are shared fairly among users. No single user or application can take up all the bandwidth, allowing everyone to have a fair share.

Better User Experience: When data flows smoothly and quickly, users have a better experience. Websites, online services, and applications work more reliably and without annoying delays.

Mitigation of Network Congestion Collapse: Without congestion control, a sudden spike in data traffic can overwhelm the network, causing severe congestion and making it almost unusable. Congestion control helps prevent this by managing traffic efficiently and avoiding such critical breakdowns.

Congestion Control is a mechanism that controls the entry of data packets into the network, enabling a better use of a shared network infrastructure and avoiding congestive collapse.
There are two congestion control algorithms which are: A leaky bucket algorithms and a token bucket algorithm.

4.3.2. Deadlocks

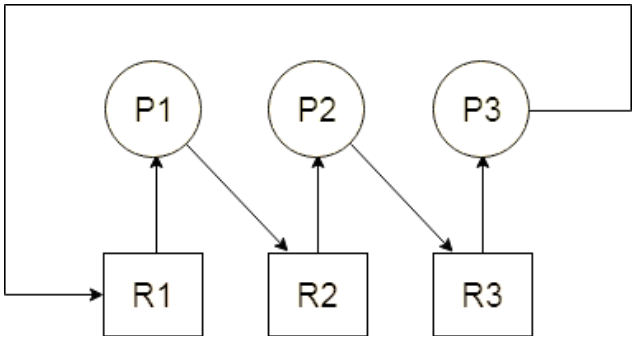
Every process needs some resources to complete its execution.
The resource is allocated in following sequential order.

1. The process requests for some resource.
2. OS grant the resource if it is available otherwise let the process waits.
3. The process uses it and release on the completion.

A **Deadlock** is a situation where each of the computer process waits for a resource which is being assigned to some another process. In this situation, none of the process gets executed since the resource it needs, is held by some other process which is also waiting for some other resource to be released.

Deadlock occurs when processes are blocked waiting for resources held by other processes in a circular chain. There are four necessary conditions for deadlock: mutual exclusion, hold and wait, no preemption, and circular wait. Approaches to handle deadlock include prevention, avoidance, and detection/recovery. Prevention methods modify resource allocation to break one of the conditions, avoidance methods allocate resources in a way that avoids unsafe states that could lead to deadlock, and detection/recovery finds and resolves deadlocks after they occur.

Let us assume that there are three processes P1, P2 and P3. There are three different resources R1, R2 and R3. R1 is assigned to P1, R2 is assigned to P2 and R3 is assigned to P3. After some time, P1 demands for R1 which is being used by P2. P1 halts its execution since it can't complete without R2. P2 also demands for R3 which is being used by P3. P2 also stops its execution because it can't continue without R3. P3 also demands for R1 which is being used by P1 therefore P3 also stops its execution. In this scenario, a cycle is being formed among the three processes. None of the process is progressing and they are all waiting. The computer becomes unresponsive since all the processes got blocked.



Necessary conditions for Deadlocks:

1. **Mutual Exclusion:** resource can only be shared in mutually exclusive manner. It implies, if two processes cannot use the same resource at the same time.
2. **Hold and Wait:** A process waits for some resources while holding another resource at the same time.
3. **No preemption:** The process which once scheduled will be executed till the completion. No other process can be scheduled by the scheduler meanwhile.
4. **Circular Wait:** All the processes must be waiting for the resources in a cyclic manner so that the last which is process is waiting for the resource being held by the first process.

Strategies for handling Deadlock:

1. Deadlock Ignorance: In this approach, the Operating system assumes that deadlock never occurs. It simply ignores deadlock. This approach is best suitable for a single end user system where User uses the system only for browsing and all other normal stuff. In these types of systems, the user has to simply restart the computer in the case of deadlock. Windows and Linux are mainly using this approach.

2. Deadlock prevention: Deadlock happens only when Mutual Exclusion, hold and wait, No pre-emption and circular wait holds simultaneously. If it is possible to violate one of the four conditions at anytime then the deadlock can never occur in the system. The idea behind the approach is very simple that we have to fail one of the four conditions.

3. Deadlock avoidance: In deadlock avoidance, the operating system checks whether the system is in safe state or in unsafe state at every step which the operating system performs. The process continues until the system is in safe state. Once the system moves to unsafe state, the OS has to backtrack one step. In simple words, The OS reviews each allocation so that the allocation doesn't cause the deadlock in the system.

4. Deadlock detection and recovery: This approach let the processes fall in deadlock and then periodically check whether deadlock occur in the system or not. If it occurs then it applies some of the recovery methods to the system to get rid of deadlock.

-----X----