Unit-8
Network Layer and Internet Layer

Introduction

The network layer is concerned with getting packets from the source all the way to the destination with minimal cost. Unlike the Data Link Layer, which has the more modest goal of just moving frames from one end of a wire to the other. Network Layer is the lowest layer that deals with end-to-end transmission.

Network layer design issues

1. **Store-and-Forward Packet Switching:**

The major components of the system are the carrier's equipment (routers connected by transmission lines), shown inside the shaded oval, and the customers' equipment, shown outside the oval. Host H1 is directly connected to one of the carrier's routers, A, by a leased line. In contrast, H2 is on a LAN with a router, F, owned and operated by the customer. This router also has a leased line to the carrier's equipment. We have shown F as being outside the oval because it does not belong to the carrier, but in terms of construction, software, and protocols, it is probably no different from the carrier's routers.

This equipment is used as follows. A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the carrier. The packet is stored there until it has fully arrived so the checksum can be verified. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered. This mechanism is store-and forward packet switching.

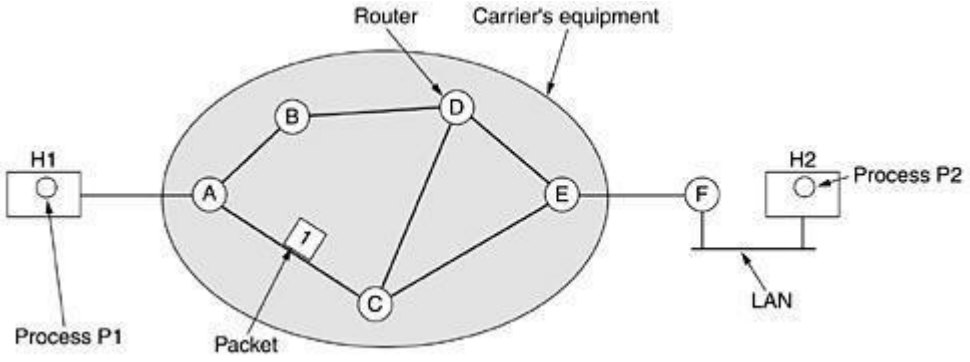


Fig: The environment of network layers protocol

2. **Services Provided to the Transport Layer:**

The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is what kind of services the network layer provides to the transport layer.

The network layer services have been designed with the following goals in mind.

a. The services should be independent of the router technology.
b. The transport layer should be shielded from the number, type, and topology of the routers present.
c. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

Given these goals, the designers of the network layer have a lot of freedom in writing detailed specifications of the services to be offered to the transport layer. This freedom often degenerates into a raging battle between two warring factions.

The other camp argues that the subnet should provide a reliable, connection-oriented service. They claim that 100 years of successful experience with the worldwide telephone system is an excellent guide. In this view, quality of service is the dominant factor, and without connections in the subnet, quality of service is very difficult to achieve, especially for real-time traffic such as voice and video.

These two camps are best exemplified by the Internet and ATM. The Internet offers connectionless network-layer service; ATM networks offer connection-oriented network-layer service. However, it is interesting to note that as quality-of-service guarantees are becoming more and more important, the Internet is evolving.

3. Implementation of Connectionless Service:

Two different organizations are possible, depending on the type of service offered. If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed.

In this context, the packets are frequently called datagrams (in analogy with telegrams) and the subnet is called a datagram subnet. If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent.

This connection is called a VC (virtual circuit), in analogy with the physical circuits set up by the telephone system, and the subnet is called a virtual-circuit subnet. In this section we will examine datagram subnets; in the next one we will examine virtual-circuit subnets.

Let us now see how a datagram subnet works. Suppose that the process P1 in Fig. 3-2 has a long message for P2. It hands the message to the transport layer with instructions to deliver it to process P2 on host H2. The transport layer code runs on H1, typically within the operating system. It prepends a transport header to the front of the message and hands the result to the network layer, probably just another procedure within the operating system.

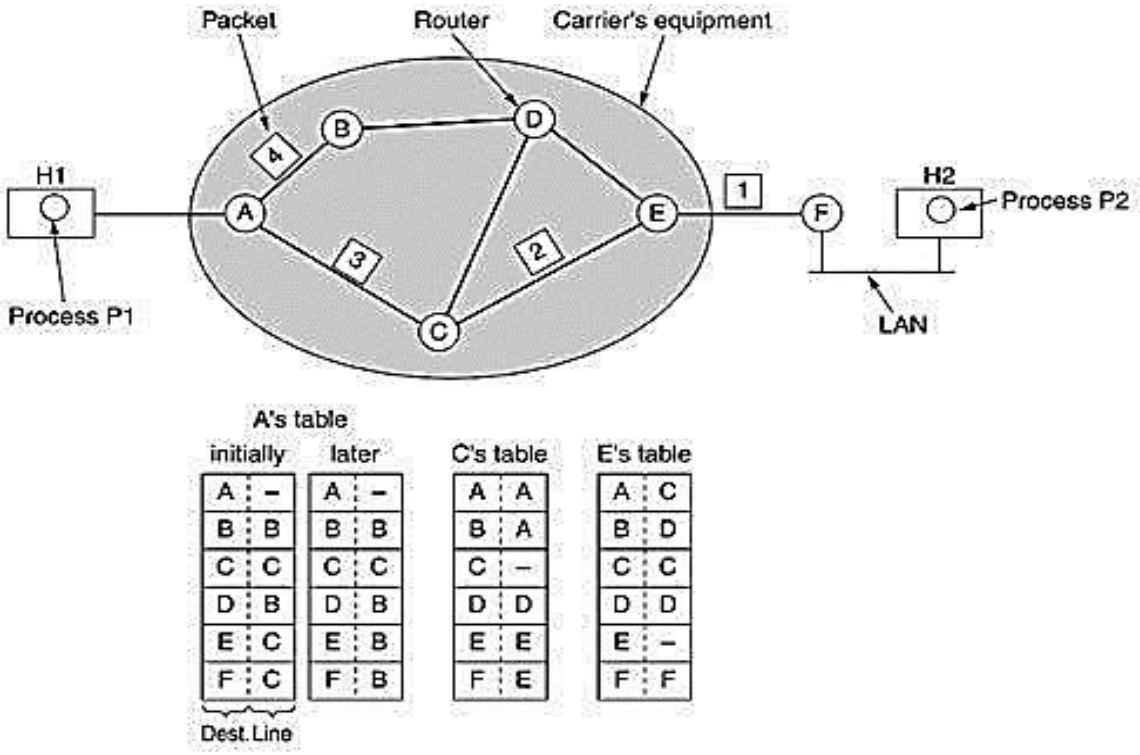


Fig: Routing within Datagram subnet

Let us assume that the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4 and sends each of them in turn to router A using some point-to-point protocol, for example, PPP.

At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination.

Only directly-connected lines can be used. For example, in Fig. 5-2, A has only two outgoing lines to B and C so every incoming packet must be sent to one of these routers, even if the ultimate destination is some other router. A's initial routing table is shown in the figure under the label "initially."

However, something different happened to packet 4. When it got to A it was sent to router B, even though it is also destined for F. For some reason, A decided to send packet 4 via a different route than that of the first three.

Perhaps it learned of a traffic jam somewhere along the ACE path and updated its routing table, as shown under the label "later." The algorithm that manages the tables and makes the routing decisions is called the routing algorithm.

4. Implementation of Connection-Oriented Service:

For connection-oriented service, we need a virtual-circuit subnet. The idea behind virtual circuits is to avoid having to choose a new route for every packet sent, as in Figure above.

Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers. That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.

When the connection is released, the virtual circuit is also terminated. With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to. As an example, consider the situation of Figure below. Here, host H1 has established connection 1 with host H2.

It is remembered as the first entry in each of the routing tables. The first line of A's table says that if a packet bearing connection identifier 1 comes in from H1, it is to be sent to router C and given connection identifier 1. Similarly, the first entry at C routes the packet to E, also with connection identifier 1.

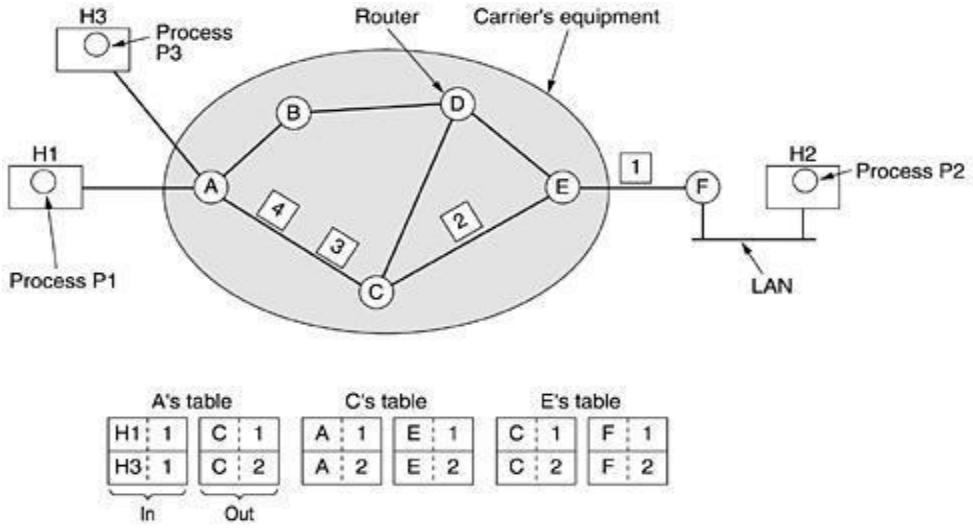


Fig: Routing within virtual circuit subnet

Now let us consider what happens if H3 also wants to establish a connection to H2. It chooses connection identifier 1 and tells the subnet to establish the virtual circuit. This leads to the second row in the tables.

Note that we have a conflict here because although A can easily distinguish connection 1 packets from H1 from connection 1 packets from H3, C cannot do this. For this reason, A assigns a different connection identifier to the outgoing traffic for the second connection.

Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets. In some contexts, this is called label switching.

5. Comparison of Virtual-Circuit and Datagram Subnets:

Both virtual circuits and datagrams have their supporters and their detractors. We will now attempt to summarize the arguments both ways. The major issues are listed in Figure below, although purists could probably find a counter example for everything in the figure.

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Fig: comparison of Datagram and Virtual-Circuit Subnets

Inside the subnet, several trade-offs exist between virtual circuits and datagrams. One trade-off is between router memory space and bandwidth. Virtual circuits allow packets to contain circuit numbers instead of full destination addresses.

If the packets tend to be fairly short, a full destination address in every packet may represent a significant amount of overhead and hence, wasted bandwidth. The price paid for using virtual circuits internally is the table space within the routers.

Depending upon the relative cost of communication circuits versus router memory, one or the other may be cheaper. Another trade-off is setup time versus address parsing time. Using virtual circuits requires a setup phase, which takes time and consumes resources.

However, figuring out what to do with a data packet in a virtual-circuit subnet is easy: the router just uses the circuit number to index into a table to find out where the packet goes. In a datagram subnet, a more complicated lookup procedure is required to locate the entry for the destination.

For transaction processing systems (e.g., stores calling up to verify credit card purchases), the overhead required to set up and clear a virtual circuit may easily dwarf the use of the circuit. If the

majority of the traffic is expected to be of this kind, the use of virtual circuits inside the subnet makes little sense.

On the other hand, permanent virtual circuits, which are set up manually and last for months or years, may be useful here. Virtual circuits also have a vulnerability problem. If a router crashes and loses its memory, even if it comes back up a second later, all the virtual circuits passing through it will have to be aborted.

In contrast, if a datagram router goes down, only those users whose packets were queued in the router at the time will suffer, and maybe not even all those, depending upon whether they have already been acknowledged.
The loss of a communication line is fatal to virtual circuits using it but can be easily compensated for if datagrams are used. Datagrams also allow the routers to balance the traffic throughout the subnet, since routes can be changed partway through a long sequence of packet transmissions.

Routing Algorithms:

The main function of the network layer is routing packets from source to destination. The algorithms that choose the routes and the data structures that they use area major area of network layer design.

The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagrams internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. In the subnet using virtual circuits such decision is made ones per session.

Routing algorithms can be grouped into two major classes: non-adaptive and adaptive:

1. Non-adaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J is computed in advance, of-line, and downloaded to the routers when the network is booted. This procedure is sometimes called static routing.
2. Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where the get their information, when they change the routes, and what metric is used for optimization. They are also called dynamic.

Some Routing Algorithms:

1. Shortest Path Routing Algorithm:

The following technique is widely used in many forms, because it is simple and easy to understand. The idea is to build a graph of the subnet, with each node of the graph representing a router and

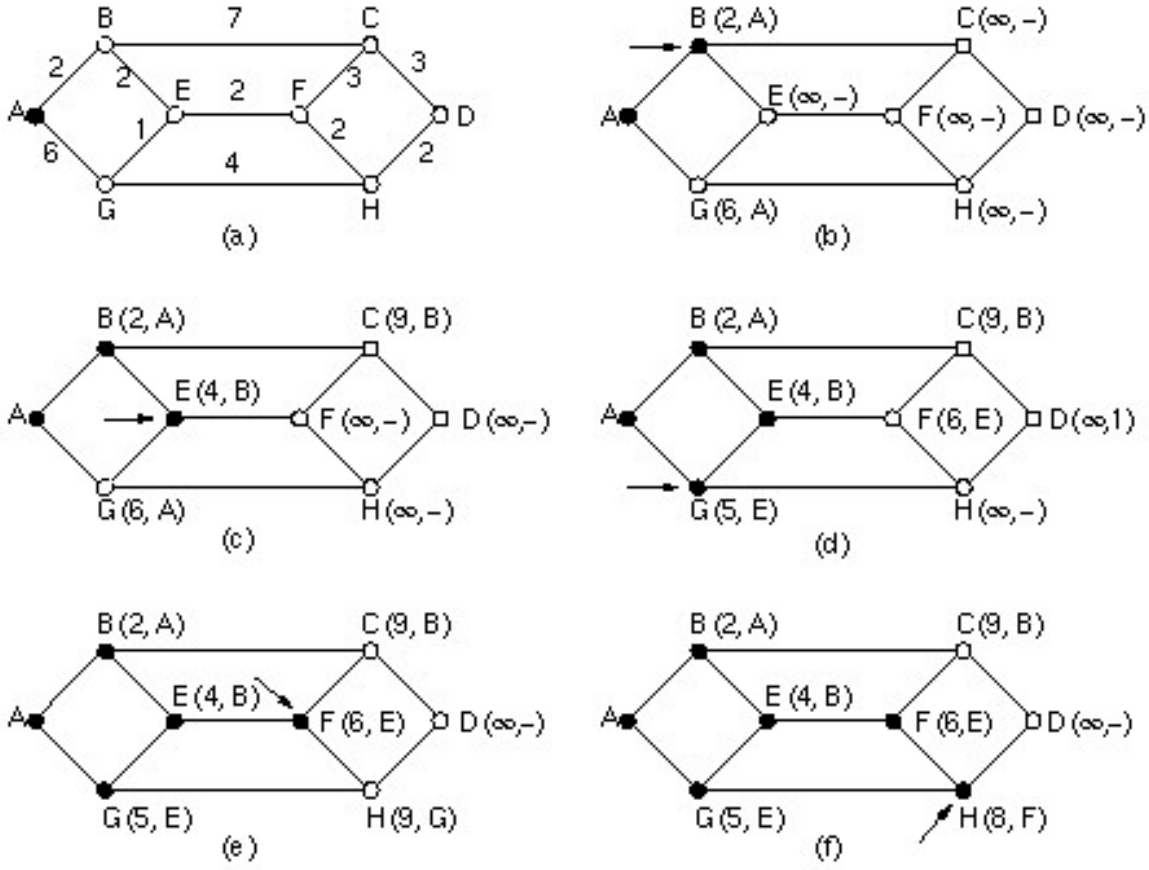
each arc representing a communication line (link).To choose a rout between a given pair of routers, the algorithm just finds the shortest path between them on the graph. The shortest path concept includes definition of the way of measuring path length. Deferent metrics like number of hops, geographical distance, and the mean queuing and transmission delay of router can be used. In the most general case, the labels on the arcs could be computed as a function of the distance, bandwidth, average traffic, communication cost, mean queue length, measured delay, and other factors.

There are several algorithms for computing shortest path between two nodes of a graph. One of them is Dijkstra's Algorithm.

Steps:

1. Start with a local node (router) i.e. the root of the tree.
2. Assign a cost '0' to the node and make the first permanent node.
3. Examine each neighbor node of the node that was last permanent node.
4. Assign a cumulative cost to each node and make them tentative.
5. Among the list of tentative nodes:
   - Find the node with the smallest cumulative cost and make it permanent.
   - If a node can be reached from more than one direction, then select the direction with shortest cumulative cost.
6. Repeat step 3 to 5 until every node becomes permanent.

Example

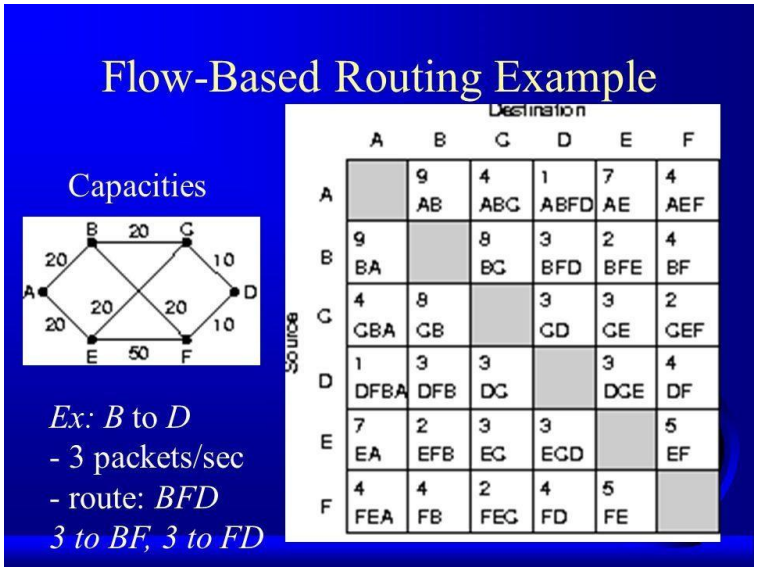


2. Flow-Based Routing:

Flow-based routing took topology as well as load into account for routing. In some networks, the mean data flow between each pair of nodes is relatively stable and predictable. Under conditions in which the average traffic from i to j is known in advance and to a reasonable approximation, constant in time, it is possible to analyze the flows mathematically to optimize the routing.

The idea behind the analysis is that for a given line, if the capacity and average flow are known, it is possible to compute the mean packet delay on that line from queuing theory. From the mean delays on all the lines, it is straightforward to calculate a flow-weighted average to get the mean packet delay for the whole subnet. The routing problem then reduces to finding the routing algorithm that produces the minimum average delay for the subnet.

This technology demands certain information in advance. First the subnet topology, second the traffic matrix, third the capacity matrix and finally a routing algorithm must be chosen.



2. Distance Vector Routing:

Modern computer networks generally use dynamic routing algorithms rather than static ones described above. Two dynamic algorithms in particular, **distance vector** & **link state** routing are the most popular. In this section we will look at the former algorithm. In the following one we will study the later one.

Distance vector routing algorithms operate by having each router maintain a table giving the best known distance to each destination and which line to use to get there. These table are updated by exchanging information with the neighbors.

The distance vector routing algorithm is sometimes called by other names including **BellmanFord** or **Ford-Fulkerson.** It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP and in early versions of DECnet and Novell's IPX. AppleTalk & CISCO routers use improved distance vector protocols.

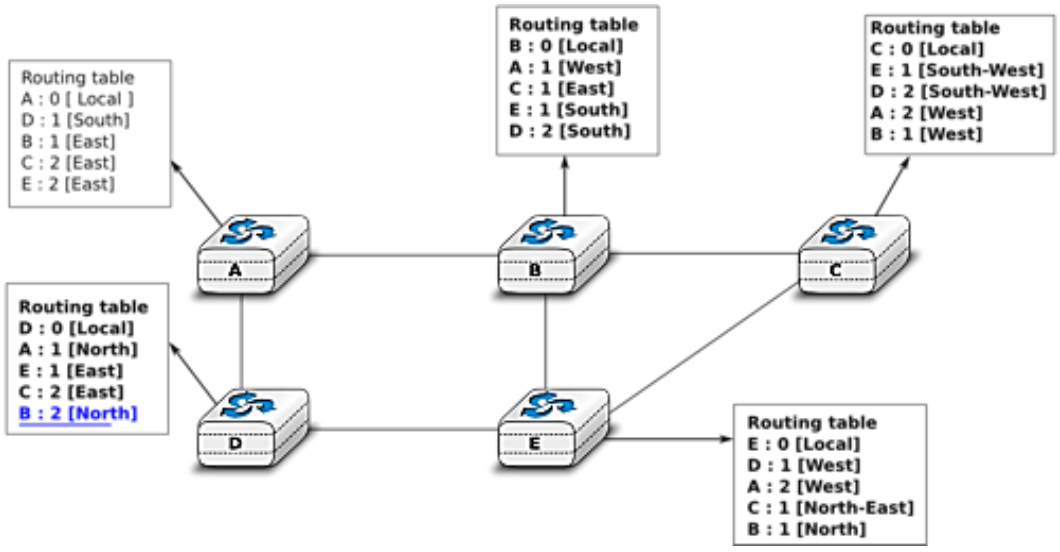
In that algorithm each router maintains a routing table indexed by and containing one entry for each router in the subnet. This entry contains two parts: the preferred outgoing line to use for that destination and an estimate of the time or distance to that destination. The metric used might be number of hops, time delay in milliseconds, and total number of packets queued along the path or something similar.

The router is assumed to know the "distance" to each of its neighbors. In the hops metric the distance is one hop, for queue length metrics the router examines each queue, for the delay metric the route can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can.

Distance vector routing works in theory, but has a serious drawback in practice: although it converges to the correct answer, it may be done slowly. Good news propagates at linear time through the subnet, while bad ones have the **count-to-infinity problem:** no router ever has a value

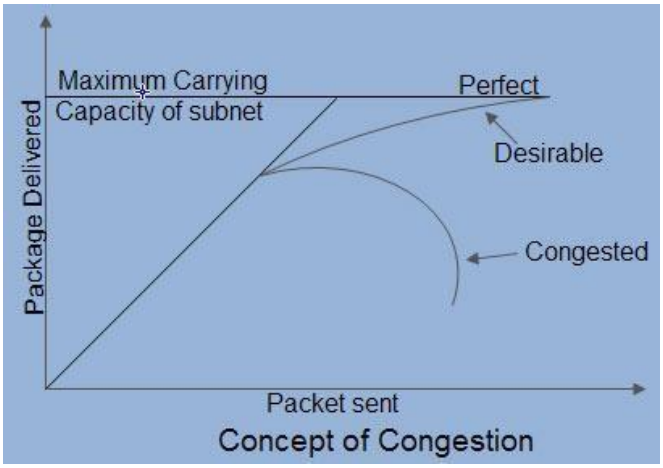
more than one higher than the minimum of all its neighbors. Gradually, all the routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinity. One of the solutions to this problem is **split horizon** algorithm that defines the distance to the X router is reported as infinity on the line that packets for X are sent on. Under that behavior bad news propagate also at linear speed through the subnet.

Examples



Congestion Control:

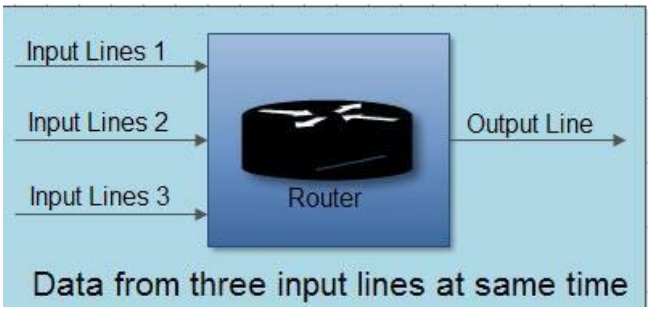
Congestion is an important issue that can arise in packet switched network. Congestion is a situation in Communication Networks in which too many packets are present in a part of the subnet, performance degrades. Congestion in a network may occur when the load on the network *(i.e.* the number of packets sent to the network) is greater than the capacity of the network *(i.e.* the number of packets a network can handle.) In other words when too much traffic is offered, congestion sets in and performance degrades sharply.



Causing of Congestion:

The various causes of congestion in a subnet are:

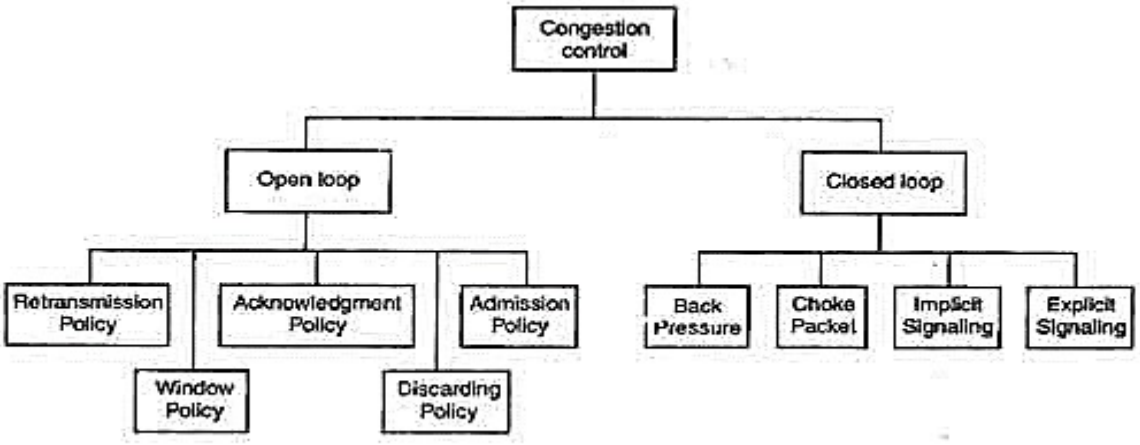
1. The input traffic rate exceeds the capacity of the output lines. If suddenly, a stream of packet start arriving on three or four input lines and all need the same output line. In this case, a queue will be built up. If there is insufficient memory to hold all the packets, the packet will be lost. Increasing the memory to unlimited size does not solve the problem. This is because, by the time packets reach front of the queue, they have already timed out (as they waited the queue). When timer goes off source transmits duplicate packet that are also added to the queue. Thus same packets are added again and again, increasing the load all the way to the destination.



2. The routers are too slow to perform bookkeeping tasks (queuing buffers, updating tables, etc.).
3. The routers' buffer is too limited.
4. Congestion in a subnet can occur if the processors are slow. Slow speed CPU at routers will perform the routine tasks such as queuing buffers, updating table etc. slowly. As a result of this, queues are built up even though there is excess line capacity.
5. Congestion is also caused by slow links. This problem will be solved when high speed links are used. But it is not always the case. Sometimes increase in link bandwidth can further deteriorate the congestion problem as higher speed links may make the network more unbalanced. Congestion can make itself worse. If a route!" does not have free buffers, it start ignoring/discarding the newly arriving packets. When these packets are discarded, the sender may retransmit them after the timer goes off. Such packets are transmitted by the sender again and again until the source gets the acknowledgement of these packets. Therefore multiple transmissions of packets will force the congestion to take place at the sending end.

How to Correct the Congestion Problem:

Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. Congestion control mechanisms are divided into two categories, one category prevents the congestion from happening and the other category removes congestion after it has taken place.



Types of Congestion Control Methods

These two categories are:

1. **<u>Open Loop Congestion Control:</u>**

In this method, policies are used to prevent the congestion before it happens. Congestion control is handled either by the source or by the destination. The various methods used for open loop congestion control are:

1. <u>Retransmission Policy:</u>

- The sender retransmits a packet, if it feels that the packet it has sent is lost or corrupted.
- However retransmission in general may increase the congestion in the network. But we need to implement good retransmission policy to prevent congestion.
- The retransmission policy and the retransmission timers need to be designed to optimize efficiency and at the same time prevent the congestion.

2. Window Policy

- To implement window policy, selective reject window method is used for congestion control.
- Selective Reject method is preferred over Go-back-n window as in Go-back-n method, when timer for a packet times out, several packets are resent, although some may have arrived safely at the receiver. Thus, this duplication may make congestion worse.
- Selective reject method sends only the specific lost or damaged packets.

3. Acknowledgement Policy:

- The acknowledgement policy imposed by the receiver may also affect congestion.
- If the receiver does not acknowledge every packet it receives it may slow down the sender and help prevent congestion.

- Acknowledgments also add to the traffic load on the network. Thus, by sending fewer acknowledgements we can reduce load on the network.
- To implement it, several approaches can be used:
    - A receiver may send an acknowledgement only if it has a packet to be sent.
    - A receiver may send an acknowledgement when a timer expires.
    - A receiver may also decide to acknowledge only N packets at a time.

4. Discarding Policy:

- A router may discard less sensitive packets when congestion is likely to happen.
- Such a discarding policy may prevent congestion and at the same time may not harm the integrity of the transmission.

5. Admission Policy:

- An admission policy, which is a quality-of-service mechanism, can also prevent congestion in virtual circuit networks.
- Switches in a flow first check the resource requirement of a flow before admitting it to the network.
- A router can deny establishing a virtual circuit connection if there is congestion in the "network or if there is a possibility of future congestion.

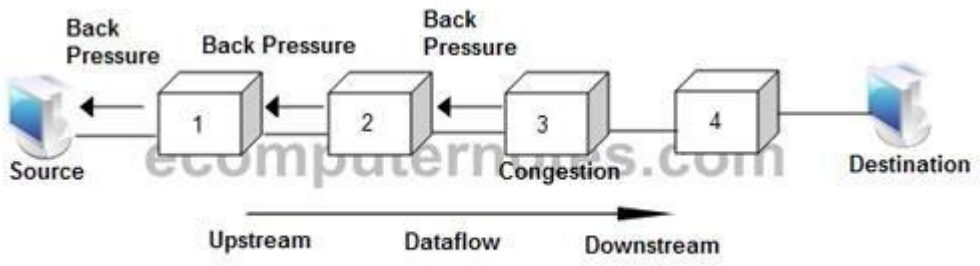
2. Closed Loop Congestion Control:

Closed loop congestion control mechanisms try to remove the congestion after it happens. The various methods used for closed loop congestion control are:

1. Backpressure:

- Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow.

  The backpressure technique can be applied only to virtual circuit networks. In such virtual circuit each node knows the upstream node from which a data flow is coming.
- In this method of congestion control, the congested node stops receiving data from the immediate upstream node or nodes.

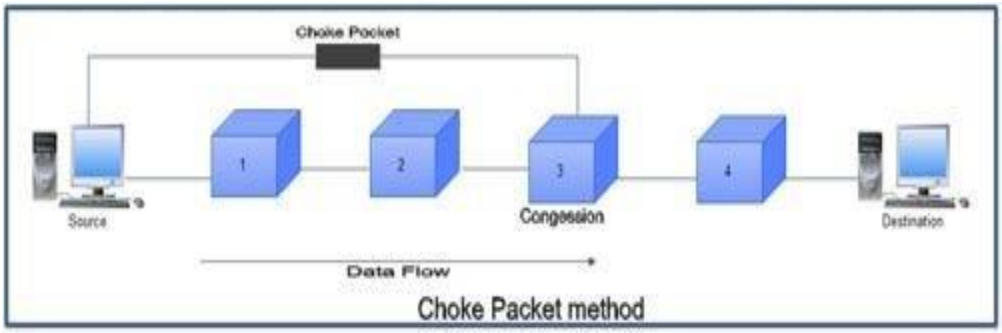


Backpressure Method

- This may cause the upstream node on nodes to become congested, and they, in turn, reject data from their upstream node or nodes.
- As shown in figure node 3 is congested and it stops receiving packets and informs its upstream node 2 to slow down. Node 2 in turns may be congested and informs node 1 to slow down. Now node 1 may create congestion and informs the source node to slow down. In this way the congestion is alleviated. Thus, the pressure on node 3 is moved backward to the source to remove the congestion.

2. Choke Packet:

- In this method of congestion control, congested router or node sends a special type of packet called choke packet to the source to inform it about the congestion.
- Here, congested node does not inform its upstream node about the congestion as in backpressure method.
- In choke packet method, congested node sends a warning directly to the source station *i.e.* the intermediate nodes through which the packet has traveled are not warned.



3. 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 when it does not receive any acknowledgment. Therefore the delay in receiving an acknowledgment is interpreted as congestion in the network.
- On sensing this congestion, the source slows down.
- This type of congestion control policy is used by TCP.

4. Explicit Signaling:

- In this method, the congested nodes explicitly send a signal to the source or destination to inform about the congestion.
- Explicit signaling is different from the choke packet method. In choke packed method, a separate packet is used for this purpose whereas in explicit signaling method, the signal is included in the packets that carry data.
- Explicit signaling can occur in either the forward direction or the backward direction.

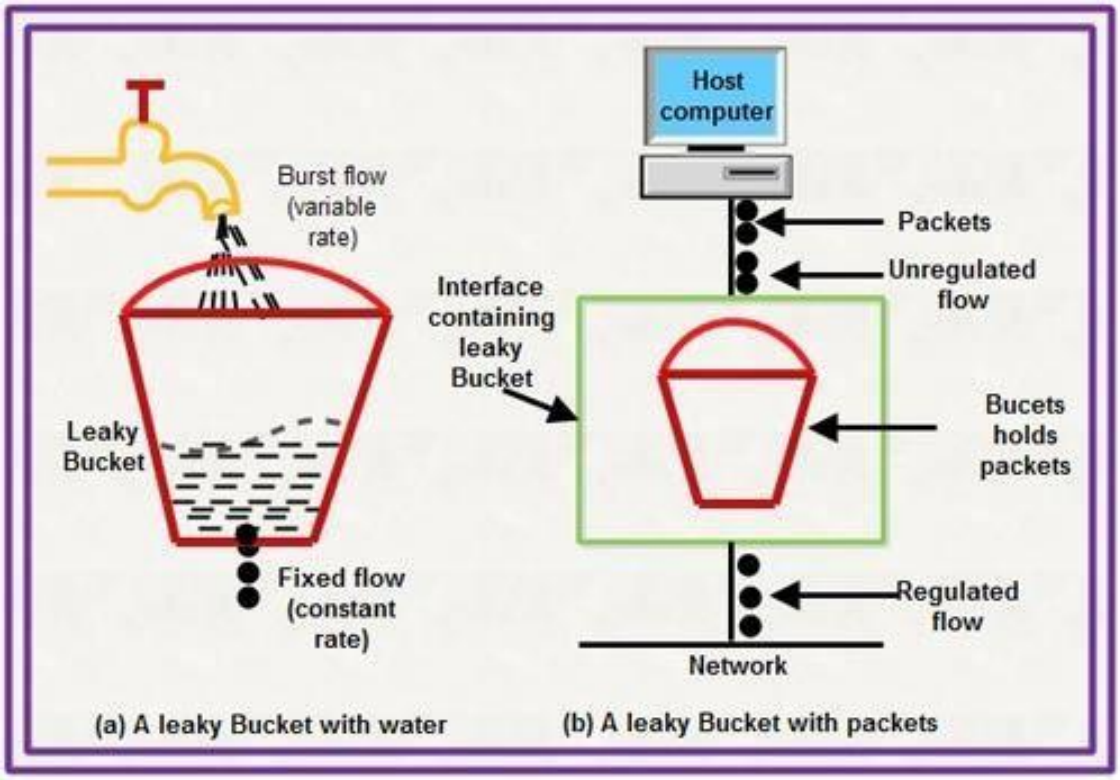
In backward signaling, a bit is set in a packet moving in the direction opposite to the congestion. This bit warns the source about the congestion and informs the source to slow down.

- In forward signaling, a bit is set in a packet moving in the direction of congestion. This bit warns the destination about the congestion. The receiver in this case uses policies such as slowing down the acknowledgements to remove the congestion.

Congestion Control Algorithms:

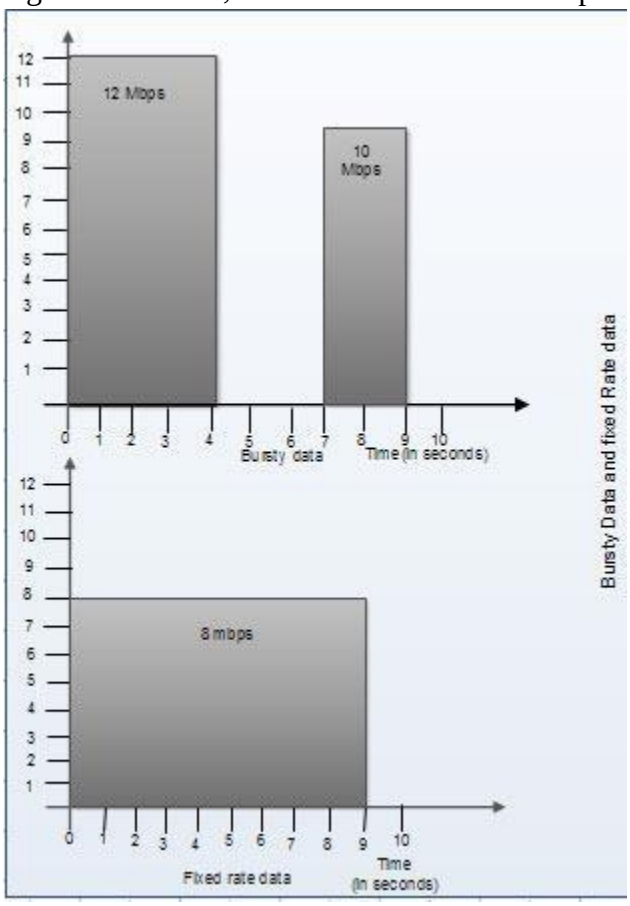
1. **Leaky Bucket Algorithm:**

It is a traffic shaping mechanism that controls the amount and the rate of the traffic sent to the network. A leaky bucket algorithm shapes bursty traffic into fixed rate traffic by averaging the data rate. Imagine a bucket with a small hole at the bottom. The rate at which the water is poured into the bucket is not fixed and can vary but it leaks from the bucket at a constant rate. Thus (as long as water is present in bucket), the rate at which the water leaks does not depend on the rate at which the water is input to the bucket.



Also, when the bucket is full, any additional water that enters into the bucket spills over the sides and is lost. The same concept can be applied to packets in the network. Consider that data is coming from the source at variable speeds. Suppose that a source sends data at 12 Mbps for 4 seconds. Then there is no data for 3 seconds. The source again transmits data at a rate of 10 Mbps for 2

seconds. Thus, in a time span of 9 seconds, 68 Mb data has been transmitted. If a leaky bucket algorithm is used, the data flow will be 8 Mbps for 9 seconds. Thus constant flow is maintained.



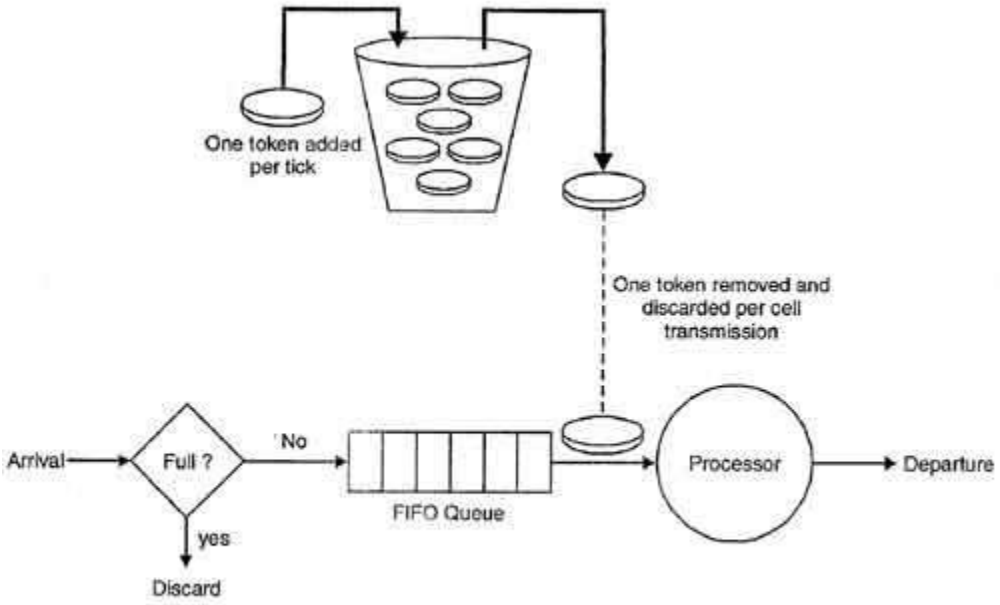
1. **Token Bucket Algorithm:**

The leaky bucket algorithm allows only an average (constant) rate of data flow. Its major problem is that it cannot deal with bursty data. A leaky bucket algorithm does not consider the idle time of the host. For example, if the host was idle for 10 seconds and now it is willing to sent data at a very high speed for another 10 seconds, the total data transmission will be divided into 20 seconds and average data rate will be maintained. The host is having no advantage of sitting idle for 10 seconds.

To overcome this problem, a token bucket algorithm is used. A token bucket algorithm allows bursty data transfers. A token bucket algorithm is a modification of leaky bucket in which leaky bucket contains tokens. In this algorithm, a token(s) are generated at every clock tick. For a packet to be transmitted, system must remove token(s) from the bucket. Thus, a token bucket algorithm allows idle hosts to accumulate credit for the future in form of tokens.

For example, if a system generates 100 tokens in one clock tick and the host is idle for 100 ticks. The bucket will contain 10,000 tokens. Now, if the host wants to send bursty data, it can consume

all 10,000 tokens at once for sending 10,000 cells or bytes. Thus a host can send bursty data as long as bucket is not empty.



Token bucket algorithm