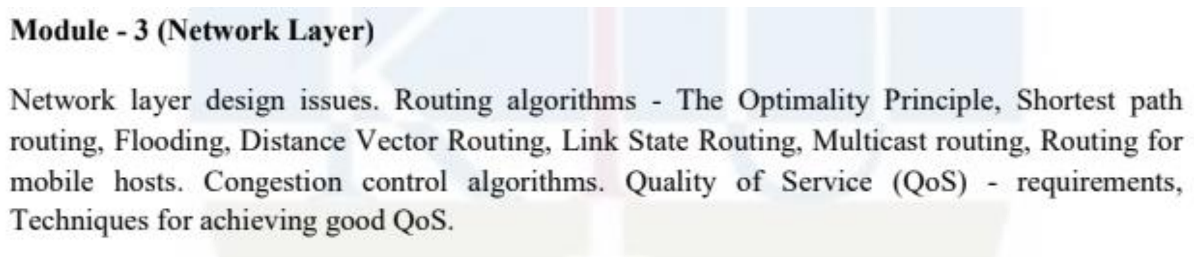
MODULE 3

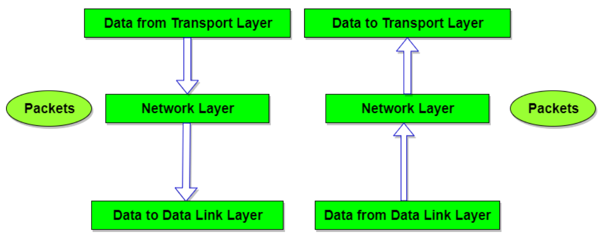


Module - 3 (Network Layer)

Network layer design issues. Routing algorithms - The Optimality Principle, Shortest path routing, Flooding, Distance Vector Routing, Link State Routing, Multicast routing, Routing for mobile hosts. Congestion control algorithms. Quality of Service (QoS) - requirements, Techniques for achieving good QoS.

Network layer

- The network layer is concerned with getting packets from the source all the way to the destination. Getting to the destination may require making many hops at intermediate routers along the way.
- deals with end-to-end transmission.
- The network Layer controls the operation of the subnet.
- The main aim of this layer is to deliver packets from source to destination across multiple links (networks). If two computers (system) are connected on the same link, then there is no need for a network layer. It routes the signal through different channels to the other end and acts as a network controller.
- It also divides the outgoing messages into packets and to assemble incoming packets into messages for higher levels.



FUNCTIONS OF NETWORK LAYER

1. It translates logical network address into physical address. Concerned with circuit, message or packet switching.
2. Routers and gateways operate in the network layer. Mechanism is provided by Network Layer for routing the packets to final destination.
3. Connection services are provided including network layer flow control, network layer error control and packet sequence control.
4. Breaks larger packets into small packets.

NETWORK LAYER DESIGN ISSUES

The network layer is concerned with getting packets from the source all the way to the destination with minimal coast.

• Unlike the DLL 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.

- Store-and-Forward Packet Switching
- Services Provided to the Transport Layer
- Implementation of Connection less Service
- Implementation of Connection-Oriented Service
- Comparison of Virtual-Circuit and Datagram Networks

Store-and-Forward Packet Switching

• A host with a packet to send transmits it to the nearest router.

• The packet is stored there until it has fully arrived.

• the link has finished its processing by verifying the checksum.

• Then it is forwarded to the next router along the path until it reaches the destination host.

• This mechanism is store-and- forward packet switching

Services Provided to the Transport Layer

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

Connection-oriented service : is a network communication mode, where a communication session or a semi-permanent connection is established before any useful data can be transferred, and where a stream of data is delivered in the same order as it was sent.

- Need a virtual-circuit network. The idea behind virtual circuits is to avoid having to choose a new route for every packet sent.
- A virtual circuit : is a means of transporting data over a packet switched computer network in such a way that it appears as though there is a dedicated physical layer link between the source and destination end systems of this data.
- With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to.
- 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 process is called label switching

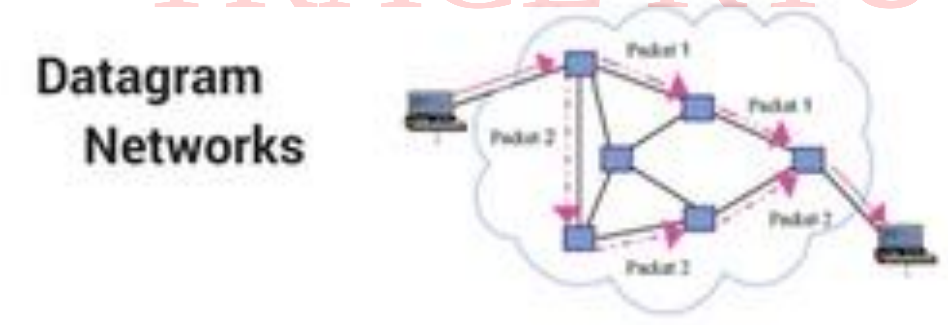
- Connectionless service

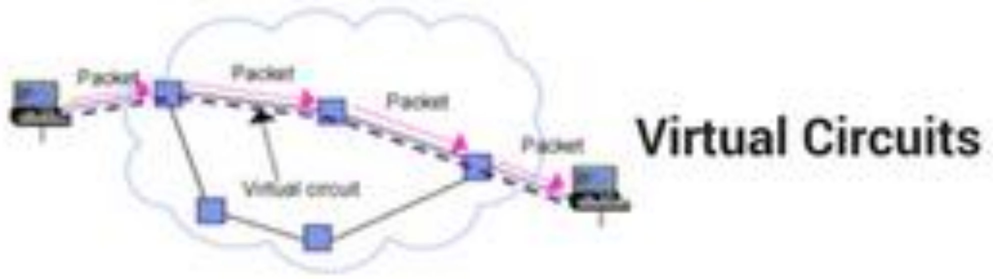
- packets are injected into the network individually and routed independently of each other. No advance setup is needed. In this context, the packets are frequently called datagrams.

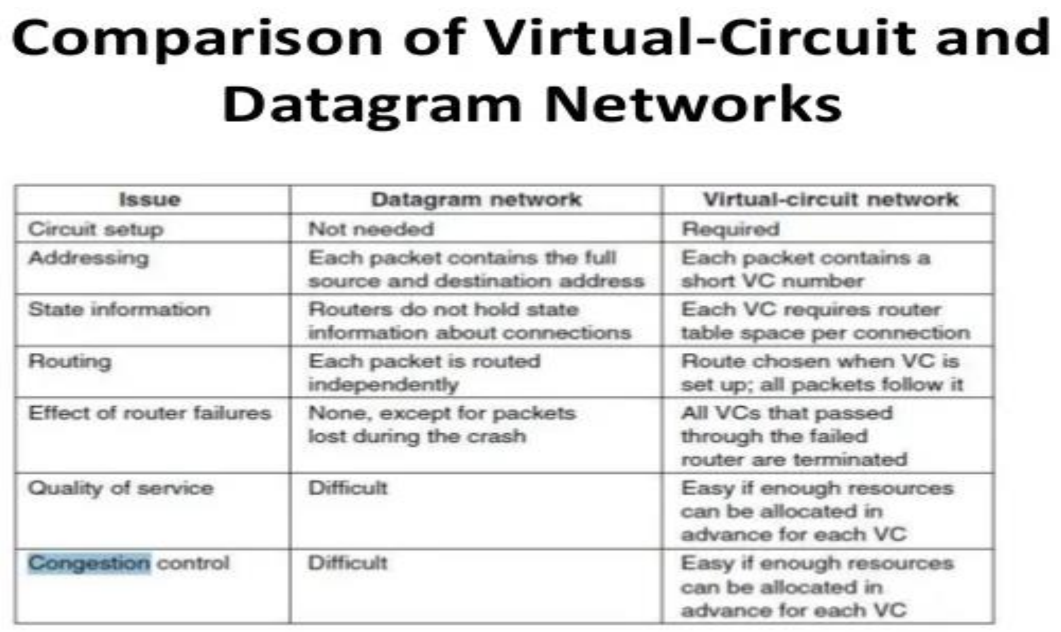
- No packet ordering and flow control should be done

- .Packet switching is a digital networking communications method that groups all transmitted data into suitably sized blocks, called packets

- A datagram is a basic transfer unit associated with a packet-switched network. The delivery, arrival time, and order of arrival need not be guaranteed by the network.







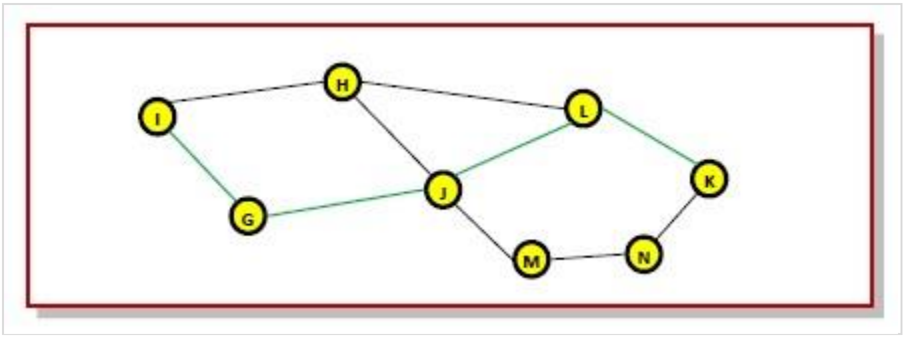
Comparison of Virtual-Circuit and Datagram Networks

Issue	Datagram network	Virtual-circuit network
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

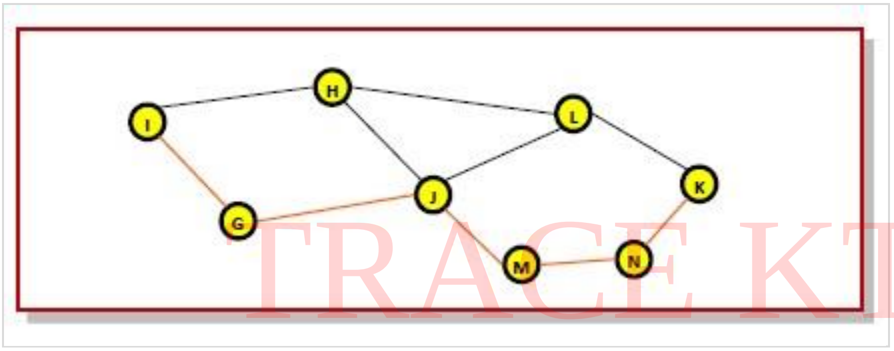
OPTIMALITY PRINCIPLE

It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route. To see this, call the part of the route from I to J r1 and the rest of the route r2. If a route better than r2 existed from J to K, it could be concatenated with r1 to improve the route from I to K, contradicting our statement that r1r2 is optimal.

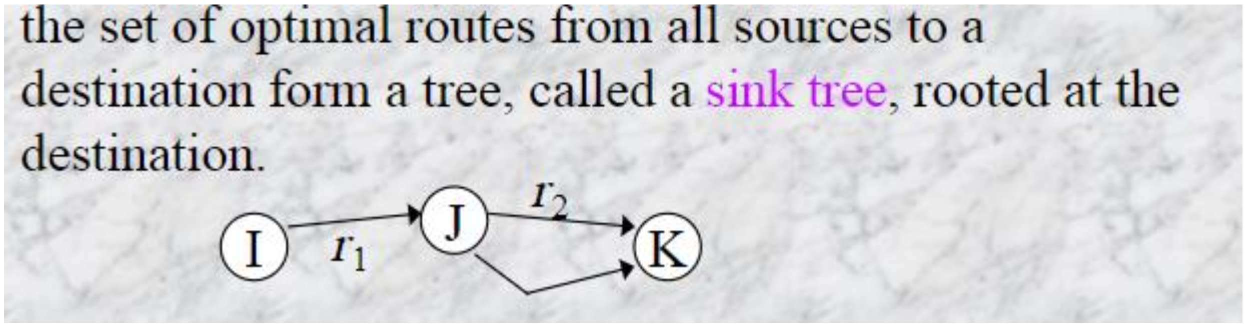
Consider a network of routers, {G, H, I, J, K, L, M, N} as shown in the figure. Let the optimal route from I to K be as shown via the green path, i.e. via the route I-G-J-L-K. According to the optimality principle, the optimal path from J to K with be along the same route, i.e. J-L-K.



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

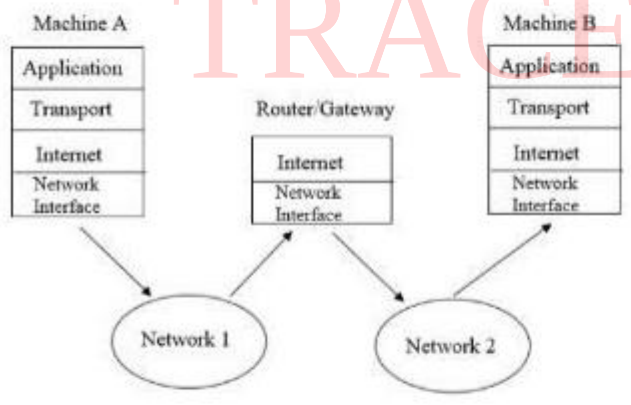


If a better route could be found between router J and router K, the path from router I to router K via J would be updated via this route. Thus, the optimal path from J to K will again lie on the optimal path from I to K.


the set of optimal routes from all sources to a destination form a tree, called a sink tree, rooted at the destination.

ROUTING

- Routing is the process of moving packets across a network from one host to an another. It is usually performed by dedicated devices called routers.
- Packets are the fundamental unit of information transport in all modern computer networks, and increasingly in other communications networks as well.
- They are transmitted over packet switched networks, which are networks on which each message (i.e., data that is transmitted) is cut up into a set of small segments prior to transmission.
- Each packet is then transmitted individually and can follow the same path or a different path to the common destination. Once all of the packets have arrived at the destination, they are automatically reassembled to recreate the original message.



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

Nonadaptive algorithms :

- The choice of the route to use to get from I to J (for all I and J) is computed in advance, off-line, and downloaded to the routers when the network is booted.
- This procedure is sometimes called static routing.
- Shortest Path Routing
- Flooding

Adaptive algorithms (Dynamic)

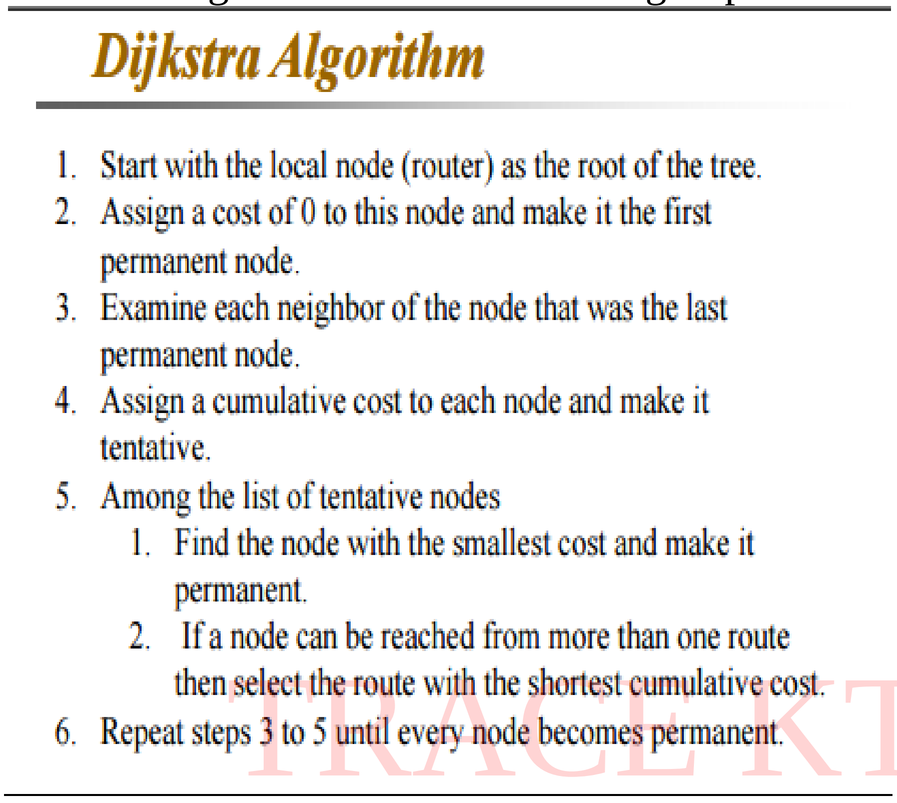
- Update their routing decisions to reflect changes in the topology, and usually the traffic as well.
- Adaptive algorithms differ in where they get their information (e.g., locally, from adjacent routers, or from all routers), when they change the routes (e.g., every ΔT sec, when the load changes or when the topology changes), and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time).
- Distance Vector Routing
- Link State Routing

SHORTEST PATH ROUTING

In computer networks, the shortest path algorithms aim **to find the optimal paths between the network nodes** so that routing cost is minimized.

- A shortest path tree is a tree in which the path between the root and every other node is the shortest. What we need for each node is a shortest path tree with that node as the root.
- The Dijkstra algorithm is used to create a shortest path tree from a given graph.

- The algorithm divides the nodes into two sets: tentative and permenant
- The algorithm uses the following steps:



Dijkstra Algorithm

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

Example

- Make node A as root and move it to tentative list

  Permanent list: empty Tentative list: A(0)

- Node A has shortest cumulative cost from all nodes in tentative list. Move A to permanent list and add all neighbors of A to tentative list

  Permanent list: A(0) Tentative list: B(5), C(2), D(3)

- Node C has shortest cumulative cost in tentative list, move C to permanent list. Node C has 3 neighbors, but A is already processed, B and E are unprocessed. B already in tentative list with cost 5. Node A can reach E via C with cumulative cost 6. Keep only minimum cost node.

Permanent list: A(0), C(2)   Tentative list: B(5), D(3), E(6)

- Node D has shortest cumulative cost, move D to permanent list. D has no unprocessed neighbors.

  Permanent list: A(0) ,C(2), D(3)   Tentative list: B(5), E(6)

- Node B has shortest cumulative cost, move B to permanent list.

  Permanent list: A(0) ,C(2), D(3) B(5),   Tentative list: E(6)

- Move E to permanent list, the final list are

  Permanent list: A(0) ,C(2), D(3) B(5), E(6)   Tentative list: empty

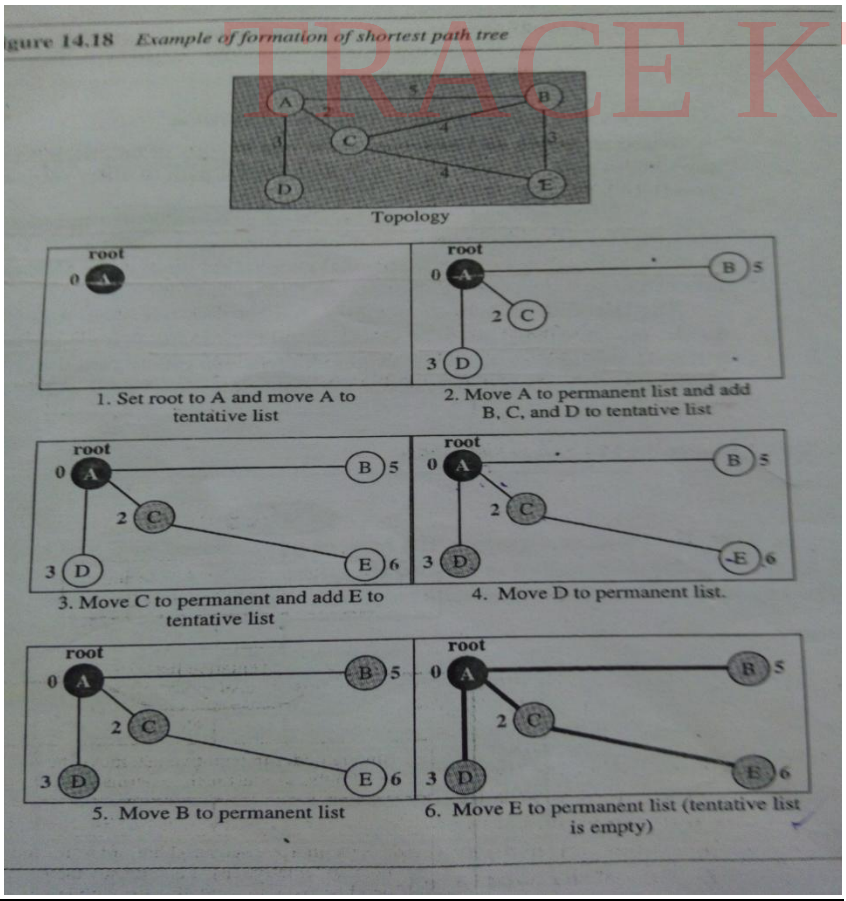


Figure 14.18 Example of formation of shortest path tree
A
B
C
D
E
5
2
3
4
3
4
Topology
root
1. Set root to A and move A to tentative list
2. Move A to permanent list and add B, C, and D to tentative list
3. Move C to permanent and add E to tentative list
4. Move D to permanent list.
5. Move B to permanent list
6. Move E to permanent list (tentative list is empty)

FLOODING

Flooding is the static routing algorithm. In this algorithm, every incoming packet is sent on all outgoing lines except the line on which it has arrived

One major problem of this algorithm is that it generates a large number of duplicate packets on the network.

Several measures are takes to stop the duplication of packets. These are:

1.) One solution is to include a hop counter in the header of each packet. This counter is decremented at each hop along the path. When this counter reaches zero the packet is discarded. Ideally, the hop counter should become zero at the destination hop, indicating that there are no more intermediate hops and destination is reached. This requires the knowledge of exact number of hops from a source to destination.

2.) Another technique is to keep the track of the packet that have been flooded, to avoid sending them a second time. For this, 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.

3) Another solution is to use selective flooding. In selective flooding the routers do not send every incoming packet out on every output line. Instead packet is sent only on those lines which are approximately going in the right direction

Flooding is used in bridging and in systems such as Usenet and peerto-peer file sharing and as part of some routing protocols, including OSPF, DVMRP, and those used in ad-hoc wireless networks (WANETs).

ADVANTAGES:

• If a packet can be delivered, it will (probably multiple times).

• Since flooding naturally utilizes every path through the network, it will also use the shortest path.

• This algorithm is very simple to implement.

DISADVANTAGES:

• Flooding can be costly in terms of wasted bandwidth. While a message may only have one destination it has to be sent to every host. In the case of a ping flood or a denial of service attack, ×it can be harmful to the reliability of a computer network.

• Messages can become duplicated in the network further increasing the load on the networks bandwidth as well as requiring an increase in processing complexity to disregard duplicate messages.

There are several variants of flooding algorithms. Most work roughly as follows:

1.Each node acts as both a transmitter and a receiver.

2.Each node tries to forward every message to every one of its neighbours except the source node. This results in every message eventually being delivered to all reachable parts of the network.

DISTANCE VECTOR ROUTING

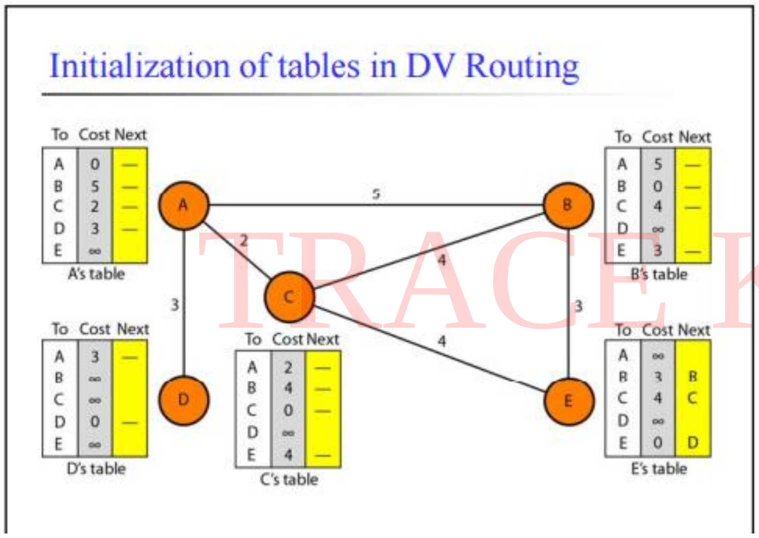
- In distance vector routing, the least cost route between any two nodes is the route with minimum distance.
- Each node maintains a vector (table) of minimum distances to every other node.

- These tables are updated by exchanging information with the neighbors.

Steps involving in distance vector routing

Initialization

- Each node knows only the distance between itself and its immediate neighbors those directly connected to it.
- The distance for any entry that is not a neighbor is marked as infinite.(unreachable)


Initialization of tables in DV Routing

A's table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	∞	

B's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	∞	
E	3	—

D's table

To	Cost	Next
A	3	—
B	∞	
C	∞	
D	0	—
E	∞	

C's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	∞	
E	4	—

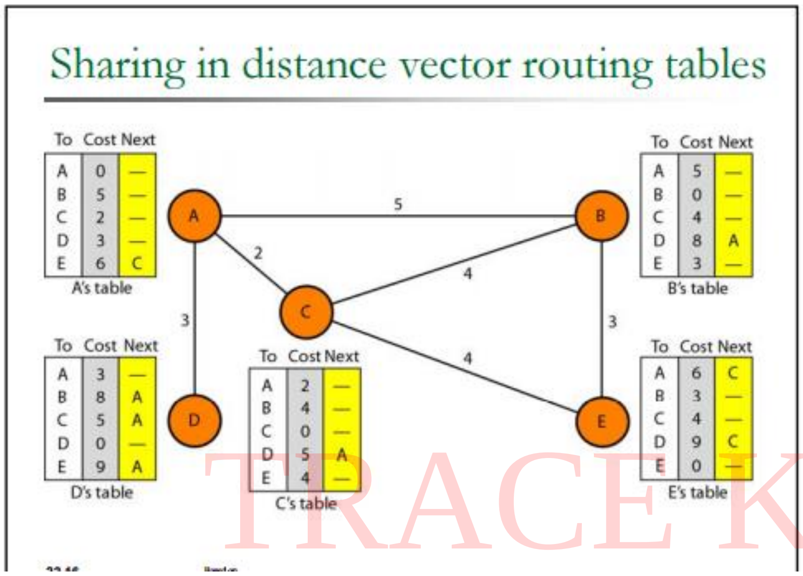
E's table

To	Cost	Next
A	∞	
B	3	B
C	4	C
D	∞	
E	0	D

Sharing

- Sharing information between neighbors.
- Each node shares its routing table with its immediate neighbor periodically and when there is a change.
- A node sends its routing table in every 30 seconds, is called periodic update.

- A node sends its routing table to its neighbors when there is a change(node failure or change in RT after updating) in its routing table is called triggered update.
- Example: node A does not know about E, but C knows. If C shares its routing table with A, node A can know how to reach E.



A's table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

B's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	8	A
E	3	—

D's table

To	Cost	Next
A	3	—
B	8	A
C	5	A
D	0	—
E	9	A

C's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	5	A
E	4	—

E's table

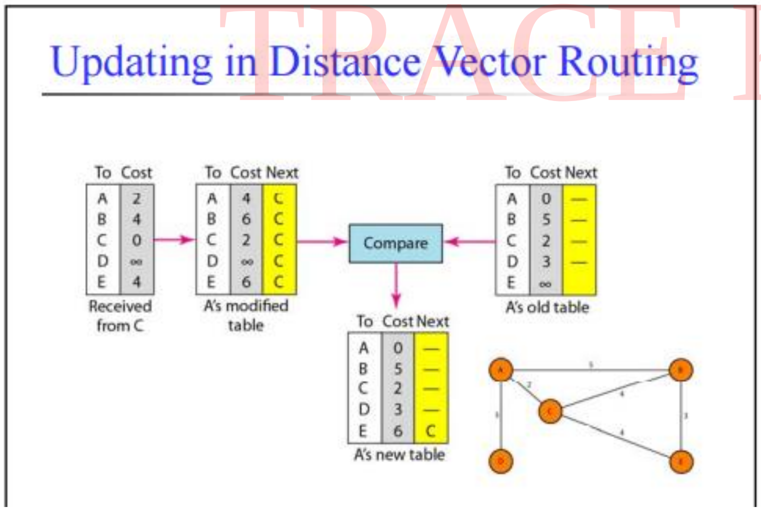
To	Cost	Next
A	6	C
B	3	—
C	4	—
D	9	C
E	0	—

Updating

- When a node receives a partial routing table from a neighbor, it needs to update its routing table
- 3 steps
  1. The receiving node needs to add the cost between itself and the sending node to each value in the second column. If C claims that its distance to destination is X and distance between A and C is Y then distance between A and destination via C is x+y.
  2. The receiving node add name of sending node to each row as third column(next)

3. The receiving node compare each of its row with the modified table
    a. If the next node entry is different, the receiving node chooses the row with smaller cost. If there is a tie, the old one is kept
    b. If the newt node entry is same, the receiving node chooses then new row

- Each node can update its table using the tables received from other nodes.
- Example : shown in figure Node A updates its routing table after receiving partial routing table from node C



Updating in Distance Vector Routing

To	Cost
A	2
B	4
C	0
D	∞
E	4

Received from C

To	Cost	Next
A	4	C
B	6	C
C	2	C
D	∞	C
E	6	C

A's modified table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	∞	—

A's old table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

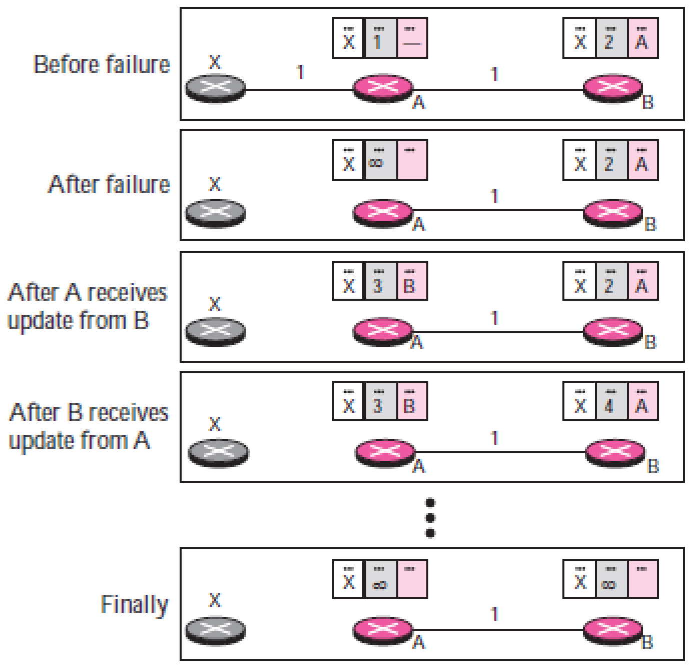
A's new table

Count to Infinity

- A problem with distance vector routing is that any decrease in cost (good news) propagates quickly, but any increase in cost (bad news) propagates slowly.

- For a routing protocol to work properly, if a link is broken (cost becomes infinity), every other router should be aware of it immediately, but in distance vector routing, this takes some time.
- The problem is referred to as count to infinity.
- It takes several updates before the cost for a broken link is recorded as infinity by all routers.
- Two node loop instability
- The figure shows a system with three nodes.
- At the beginning, both nodes A and B know how to reach node X. But suddenly, the link between A and X fails. Node A changes its table. If A can send its table to B immediately, everything is fine.
- However, the system becomes unstable if B sends its routing table to A before receiving A's routing table.
- Node A receives the update and, assuming that B has found a way to reach X, immediately updates its routing table. Now A sends its new update to B. Now B thinks that something has been changed around A and updates its routing table. The cost of reaching X increases gradually until it reaches infinity
- Node A thinks that the route to X is via B; node B thinks that the route to X is via A. If A receives a packet destined for X, it goes to B and then comes back to A. Similarly, if B receives a packet destined for X, it goes to A and comes back to B. Packets bounce between A and B, creating a two-node loop problem

Two-node instability


Before failure
After failure
After A receives update from B
After B receives update from A
Finally
X
A
B
1
∞
2
3
4

- Solutions for count to infinity problem are
- Defining Infinity The first obvious solution is to redefine infinity to a smaller number, such as 16.
- Split Horizon
- In this strategy, instead of flooding the table through each interface, each node sends only part of its table through each interface.
- If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows). Taking information from node A, modifying it, and sending it back to node A is what creates the confusion.

- In our scenario, node B eliminates the last line of its routing table before it sends it to A. In this case, node A keeps the value of infinity as the distance to X. Later, when node A sends its routing table to B, node B also corrects its routing table. The system becomes stable after the first update: both node A and B know that X is not reachable
- Split Horizon and Poison Reverse
- Node B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."
- It does not eliminate the value, but it adds some negative information

Advantages of Distance Vector routing

• It is simpler to configure and maintain than link state routing.

Disadvantages

➢React rapidly to good news when a router comes up.

➢though it finally converges to correct result, it takes long time when where is a bad news.

➢There are several attempts to solve the problem, but none is perfect.

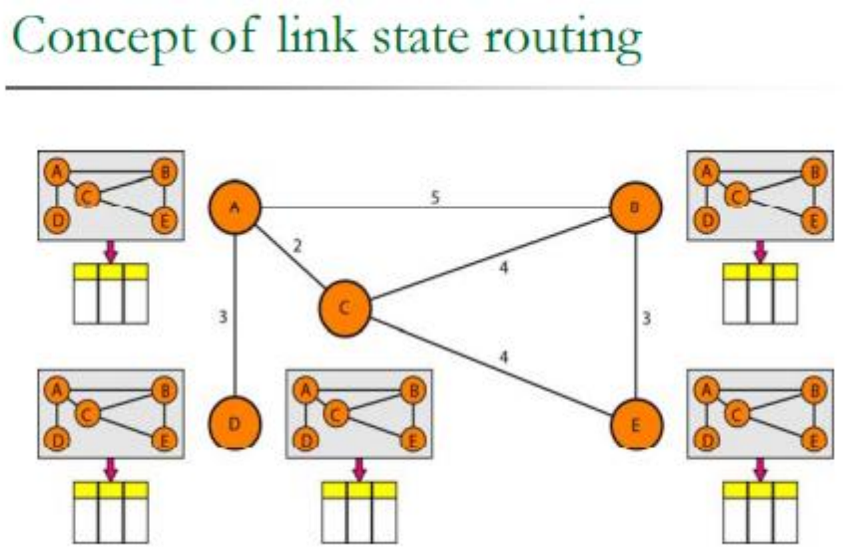
➢It does not take line bandwidth into account.

- It took too long to converge.
- It is slower to converge than link state.
- It is at risk from the count-to-infinity problem.
- It creates more traffic than link state since a hop count change must be propagated to all routers and processed on each router. Hop count updates take place on a periodic basis, even if there are no changes in the network topology, so bandwidth-wasting broadcasts still occur.
- For larger networks, distance vector routing results in larger routing tables than link state since each router must know about all other routers. This can also lead to congestion on WAN links.
- RIP(Routing Information Protocol is based on distance vector routing algorithm

LINKSTATE ROUTING

- Link-state routing is the second major class of intra domain routing protocol.
- Each node is assumed to be capable of finding out the state of the link to its neighbors (up or down) and the cost of each link.
- The basic idea behind link-state protocols is very simple: Every node knows how to reach its directly connected neighbors then every node will have enough knowledge of the network to build a

complete map of the network. This is clearly a sufficient for finding the shortest path to any point in the network.

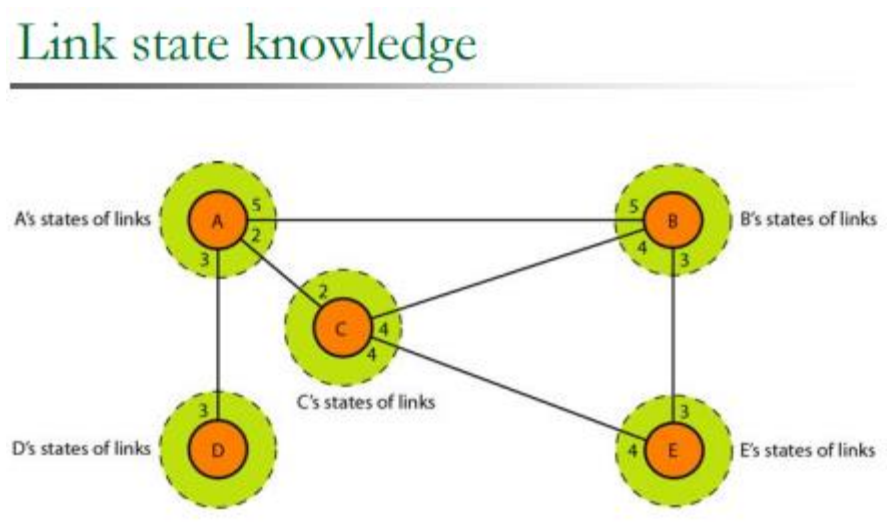


-
- Thus, link-state routing protocols rely on two mechanisms: reliable dissemination of link-state information, and the calculation of routes from the sum of all the accumulated link-state knowledge.

Steps in link state routing

In link state routing, four sets of actions are required to ensure that each node has the routing table showing the least-cost node to every other node.

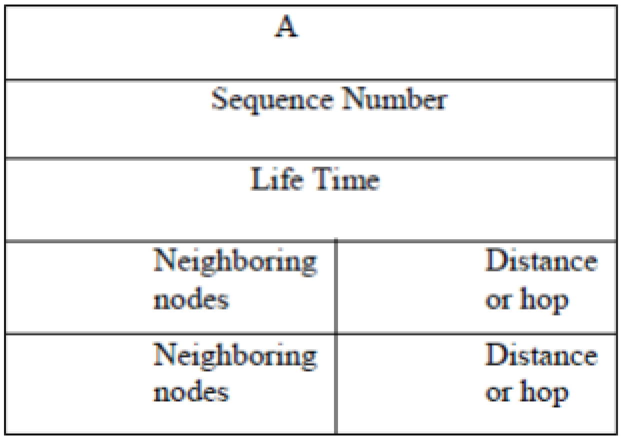
1. Creation of the states of the links by each node, called the link state packet or LSP.
2. Dissemination of LSPs to every other router, called flooding, in an efficient and reliable way.
3. Formation of a shortest path tree for each node.
4. Calculation of a routing table based on the shortest path tree.



Creation of LSP

- Each node creates an update packet, also called a link-state packet (LSP), that contains the following information:

    1. The ID of the node that created the LSP
    2. A list of directly connected neighbors of that node, with the cost of the link to each one
    3. A sequence number
    4. A time to live for this packet



A	
Sequence Number	
Life Time	
Neighboring nodes	Distance or hop
Neighboring nodes	Distance or hop

Flooding of link-state packets (Reliable Flooding)

- Reliable flooding is the process of making sure that all the nodes participating in the routing protocol get a copy of the link-state information from all the other nodes. The basic idea is for a node to send its link-state information out on all of its directly connected links, with each node that receives this information forwarding it out on all of its links. This process continues until the information has reached all the nodes in the network.
- After a node has prepared an LSP, it must be disseminated to all other nodes, not only to its neighbors. The process is called flooding and based on the following:

  1. The creating node sends a copy of the LSP out of each interface.

  2. A node that receives an LSP compares it with the copy it may already have. If the newly arrived LSP is older than the one it has (found by checking the sequence number), it discards the LSP. If it is newer, the node does the following:

  a. It discards the old LSP and keeps the new one.

  b. It sends a copy of it out of each interface except the one from which the packet arrived. This guarantees that flooding stops somewhere in the domain (where a node has only one interface).
- There are several more steps needed to reliably flood an LSP to all nodes in a network. Consider a node X that receives a copy of an LSP that originated at some other node Y.

- Note that Y may be any other router in the same routing domain as X. X checks to see if it has already stored a copy of an LSP from Y.
- If not, it stores the LSP. If it already has a copy, it compares the sequence numbers; if the new LSP has a larger sequence number, it is assumed to be the more recent, and that LSP is stored, replacing the old one. A smaller (or equal) sequence number would imply an LSP older (or not newer) than the one stored, so it would be discarded and no further action would be needed.
- If the received LSP was the newer one, X then sends a copy of that LSP to all of its neighbors except the neighbor from which the LSP was just received. The fact that the LSP is not sent back to the node from which it was received helps to bring an end to the flooding of an LSP.Most recent copy will send to all of the nodes.

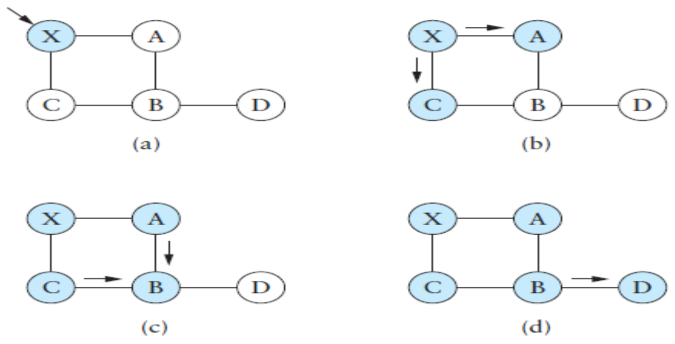


Figure 4.18 Flooding of link-state packets. (a) LSP arrives at node X; (b) X floods LSP to A and C; (c) A and C flood LSP to B (but not X); (d) flooding is complete.

- Figure 4.18 shows an LSP being flooded in a small network. Each node becomes shaded as it stores the new LSP. In Figure 4.18(a) the LSP arrives at node X, which sends it to neighbors A and C in Figure 4.18(b). A and C do not send it back to X, but send it on to B. Since B receives two identical copies of the LSP, it will accept whichever arrived first and ignore the second as a duplicate. It then passes the LSP on to D, who has no neighbors to flood it to, and the process is complete.

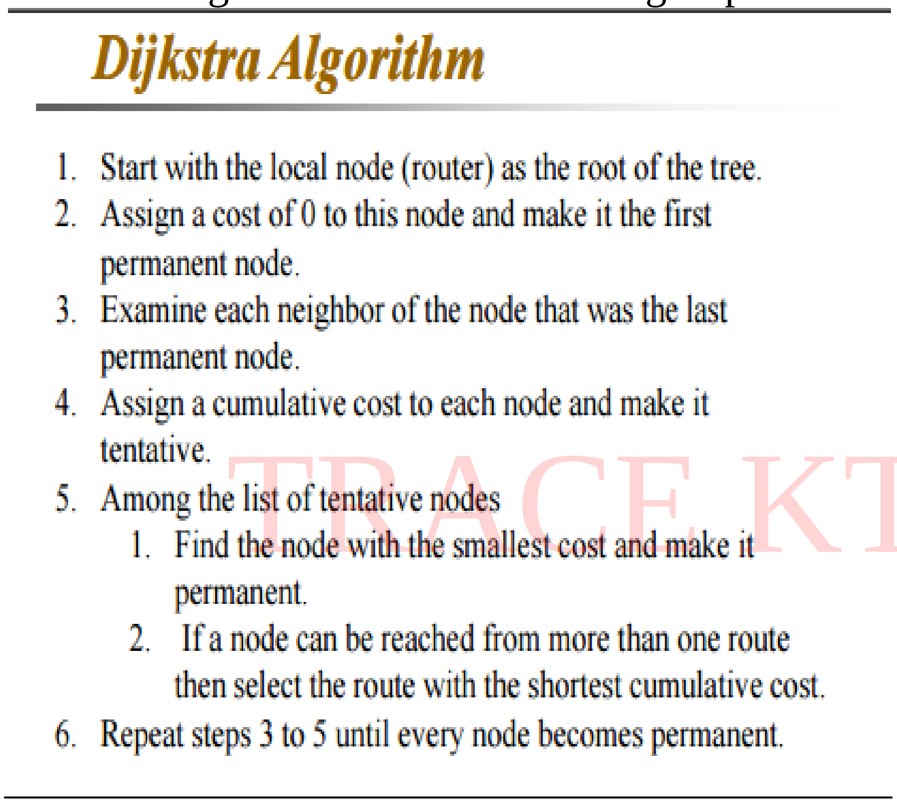
- Each node generates LSPs under two circumstances. Either the expiry of a periodic timer or a change in topology can cause a node to generate a new LSP.
- The topology-based reason for a node to generate an LSP is if one of its directly connected links or immediate neighbors has gone down. The failure of a link can be detected in some cases by the link-layer protocol. The loss of connectivity to that neighbor can be detected using periodic “hello” packets. Each node sends these to its immediate neighbors at defined intervals. If a sufficiently long time passes without receipt of a “hello” from a neighbor, the link to that neighbor will be declared down, and a new LSP will be generated to reflect this fact.

Formation of Shortest Path Tree: Dijkstra Algorithm

- After receiving all LSPs, each node will have a copy of the whole topology. However, the topology is not sufficient to find the shortest path to every other node; a shortest path tree is needed.
- A tree is a graph of nodes and links; one node is called the root. All other nodes can be reached from the root through only one single

route. A shortest path tree is a tree in which the path between the root and every other node is the shortest. What we need for each node is a shortest path tree with that node as the root. The Dijkstra algorithm is used to create a shortest path tree from a given graph.

- The algorithm divides the nodes into two sets: tentative and permenant
- The algorithm uses the following steps:



Dijkstra Algorithm

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

Example

- Make node A as root and move it to tentative list

  Permanent list: empty Tentative list: A(0)

- Node A has shortest cumulative cost from all nodes in tentative list. Move A to permanent list and add all neighbors of A to tentative list

  Permanent list: A(0) Tentative list: B(5), C(2), D(3)

- Node C has shortest cumulative cost in tentative list, move C to permanent list. Node C has 3 neighbors, but A is already processed, B and E are unprocessed. B already in tentative list with cost 5. Node A can reach E via C with cumulative cost 6. Keep only minimum cost node.

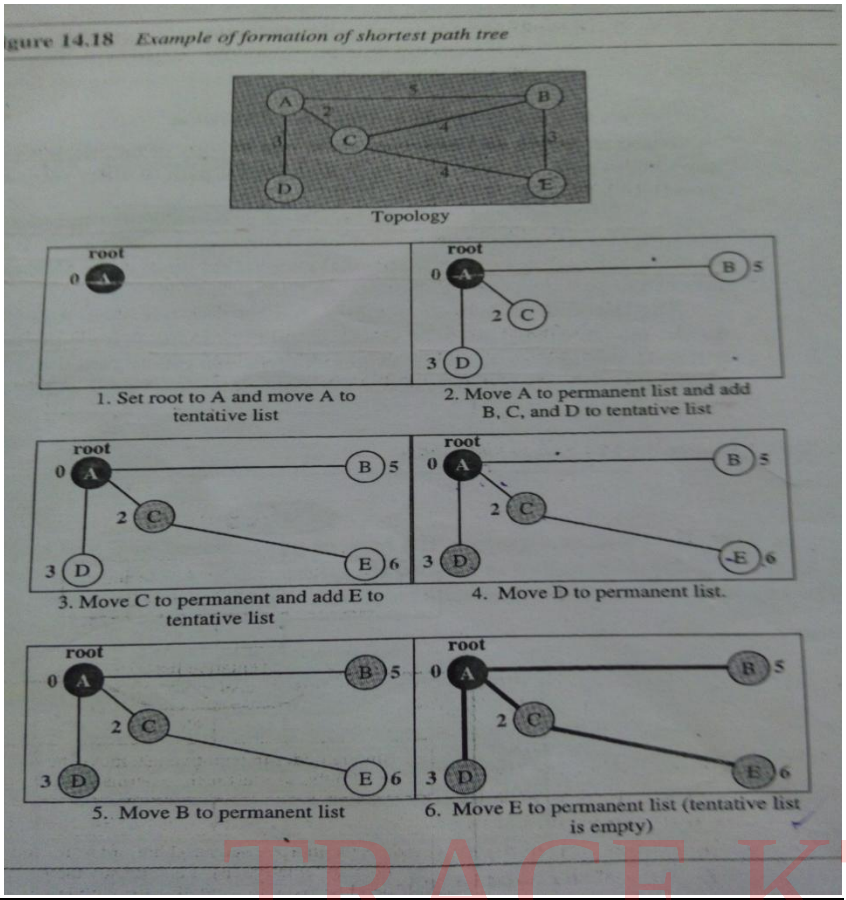
  Permanent list: A(0), C(2)   Tentative list: B(5), D(3), E(6)
- Node D has shortest cumulative cost, move D to permanent list. D has no unprocessed neighbors.

  Permanent list: A(0) ,C(2), D(3) Tentative list: B(5), E(6)
- Node B has shortest cumulative cost, move B to permanent list.

  Permanent list: A(0) ,C(2), D(3) B(5), Tentative list: E(6)
- Move E to permanent list, the final list are

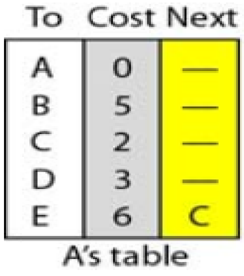
  Permanent list: A(0) ,C(2), D(3) B(5), E(6) Tentative list: empty

Figure 14.18 *Example of formation of shortest path tree*


A
B
C
D
E
5
2
4
3
4
3
Topology
root
0
A
B 5
2 C
3 D
E 6
1. Set root to A and move A to tentative list
2. Move A to permanent list and add B, C, and D to tentative list
3. Move C to permanent and add E to tentative list
4. Move D to permanent list.
5. Move B to permanent list
6. Move E to permanent list (tentative list is empty)

Calculation of routing table from shortest path tree

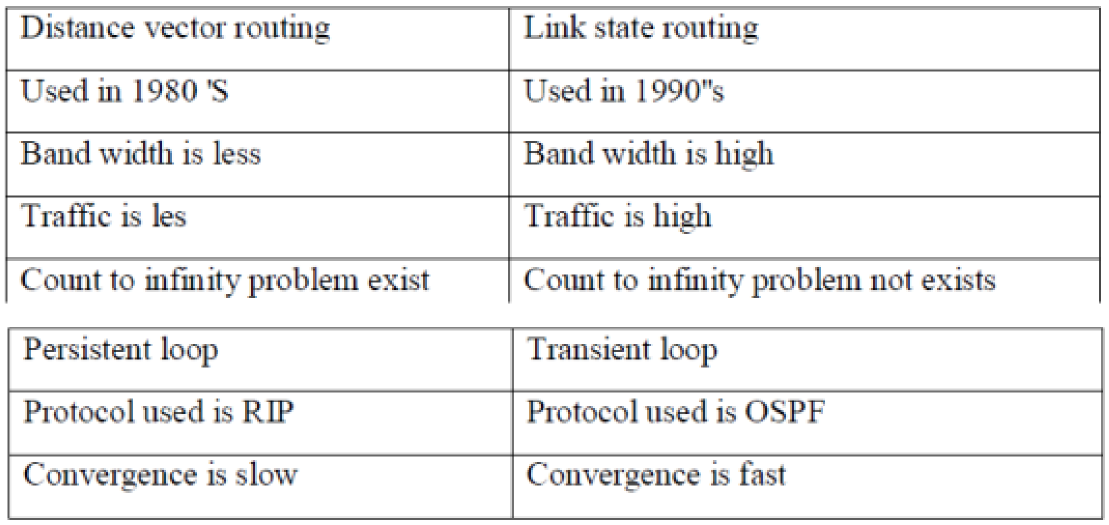
- Each node uses the shortest path tree to construct its routing table



To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

A's table

Difference between link state routing &distance vector routing



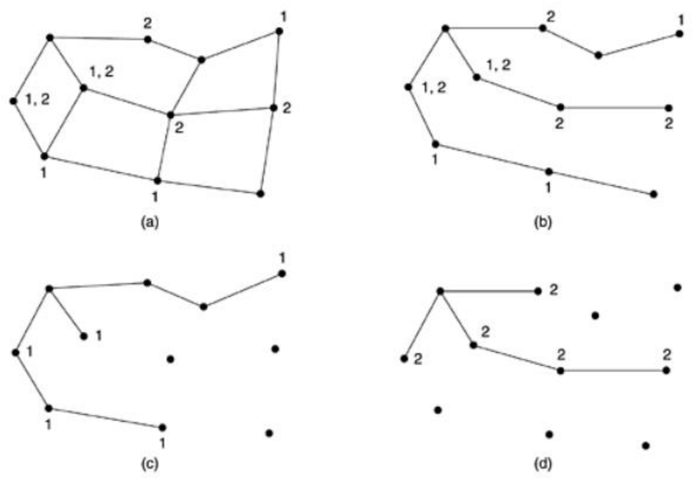
Distance vector routing	Link state routing
Used in 1980 'S	Used in 1990"s
Band width is less	Band width is high
Traffic is les	Traffic is high
Count to infinity problem exist	Count to infinity problem not exists
Persistent loop	Transient loop
Protocol used is RIP	Protocol used is OSPF
Convergence is slow	Convergence is fast

MULTICAST ROUTING

- In Multicast routing, the data is sent to only nodes which wants to receive the packets.

- Multicasting requires group management.
- Either hosts must inform their routers about changes in group membership, or routers must query their hosts periodically. Either way, routers learn about which of their hosts are in which groups. Routers tell their neighbors, so the information propagates through the subnet.
- To do multicast routing, each router computes a spanning tree covering all other routers

Example:Two groups, 1 and 2. Some routers are attached to hosts that belong to one or both of these groups, as indicated in the figure. A spanning tree for the leftmost router is shown in Fig



a) A network. (b) A spanning tree for the leftmost router. (c) A multicast tree for group 1. (d) A multicast tree for group 2.

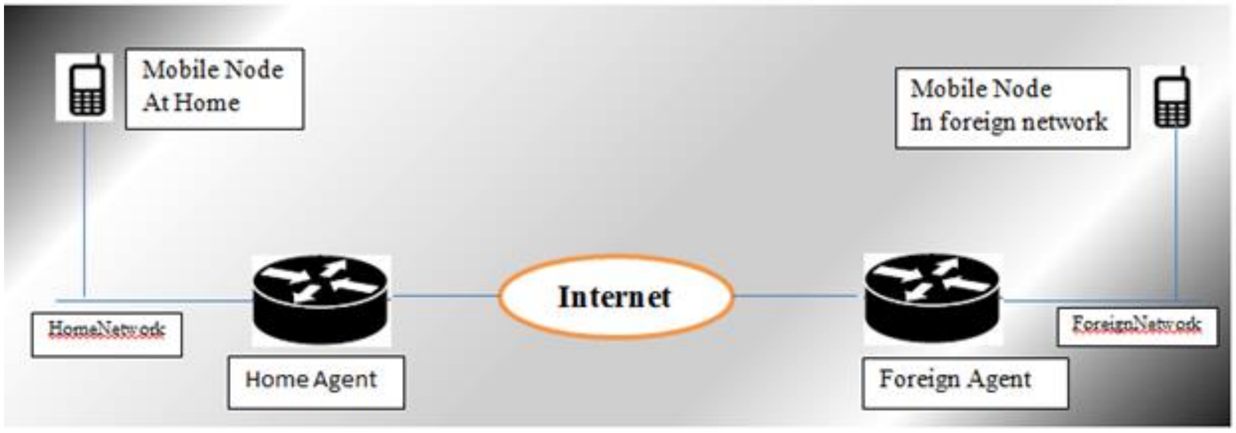
- When a process sends a multicast packet to a group, the first router examines its spanning tree and prunes it, removing all lines that do not lead to hosts that are members of the group.
- In our example, (c) shows the pruned spanning tree for group 1.Fig. (d) shows the pruned spanning tree for group 2.
- Multicast packets are forwarded only along the appropriate spanning tree.
- Pruning can be done by link state routing and distance vector routing
- In link state routing, each router is aware of the complete topology, including which hosts belong to which groups. Then the spanning tree can be pruned, starting at the end of each path, working toward the root, and removing all routers that do not belong to the group
- With distance vector routing, The basic algorithm is reverse path forwarding.

- Whenever a router with no hosts interested in a particular group and no connections to other routers receives a multicast message for that group, it responds with a PRUNE message, telling the sender not to send it any more multicasts for that group.
- One potential disadvantage of this algorithm is that it scales poorly to large networks and considerable storage is needed to store all the trees.
- An alternative design uses core-based trees. Here, a single spanning tree per group is computed, with the root (the core) near the middle of the group.
- To send a multicast message, a host sends it to the core, which then does the multicast along the spanning tree.

ROUTING FOR MOBILE HOSTS

The basic idea used for mobile routing in the Internet and cellular networks is for the mobile host to tell a host at the home location where it is now. This host, which acts on behalf of the mobile host, is called the home agent.

Once it knows where the mobile host is currently located, it can forward packets so that they are delivered.



- Each mobile node is assumed to have a permanent home location(home address) that never changes.
- The home address of a mobile device is the IP address assigned to the device within its home network.
- Home network :The home network of a mobile device is the network within which the device receives its identifying IP address (home address).

- Foreign network:A foreign network is the network in which a mobile node is operating when away from its home network.
- The world is divided up (geographically) into small units. Let us call them areas, where an area is typically a LAN or wireless cell.
- Each area has one or more foreign agents, which are processes that keep track of all mobile hosts visiting the area.
- A foreign agent is a router that stores information about mobile nodes visiting its network.
- Each area has a home agent, which keeps track of hosts whose home is in the area, but who are currently visiting another area.

- A home agent is a router on a mobile node's home network which tunnels datagrams for delivery to the mobile node when it is away from home. It maintains current location (IP address) information for the mobile node.
- The routing goal in systems with mobile hosts is to make it possible to send packets to mobile hosts using their home addresses and have the packets efficiently reach them wherever they may be.
- When a new host enters an area, either by connecting to it (e.g., plugging into the LAN) or just wandering into the cell, his computer must register itself with the foreign agent there.

The registration procedure typically works like this:

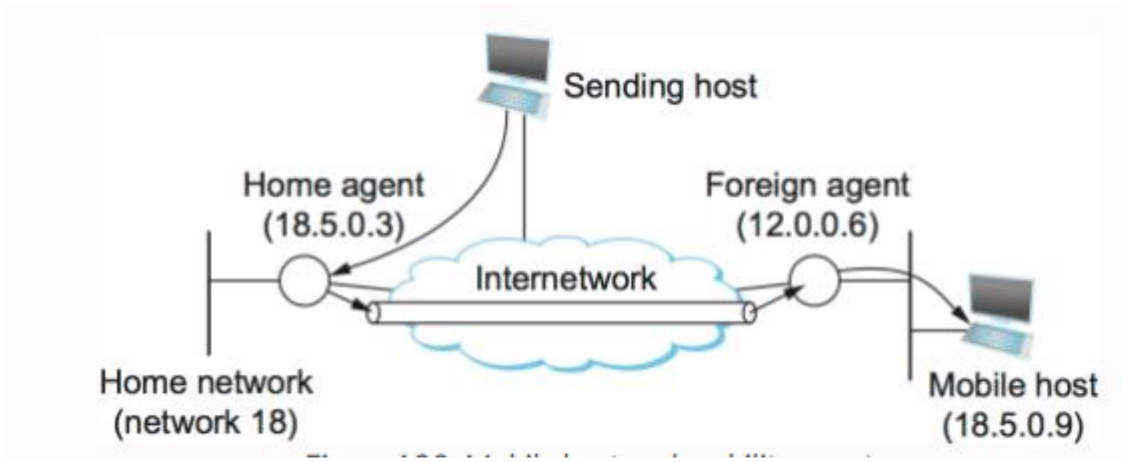
1) Periodically, each foreign agent broadcasts a packet announcing its existence and address. A newly arrived mobile host may wait for one these messages, but if one arrives quickly enough, the mobile host can broadcast a packet saying:" are there any foreign agent around? "

2) The mobile just register with the foreign agent, giving its home address, current data link layer address, and some security information.

3)The foreign agent contacts the mobile hosts home agent and says, one of your hosts is over here, the message from the foreign agent to the home agent contain the foreign agent network address. It also includes the security information, to convince the home agent that the mobile hosts are really there.

4) The home agent examines the security information, which contains a timestamp, to prove that it was generated within the past few seconds. If it is happy, it tells the foreign agent to proceed.

5) When the foreign agent gets the acknowledgement from the home agent, it makes an entry in its table and informs the mobile hosts that it is now registered.

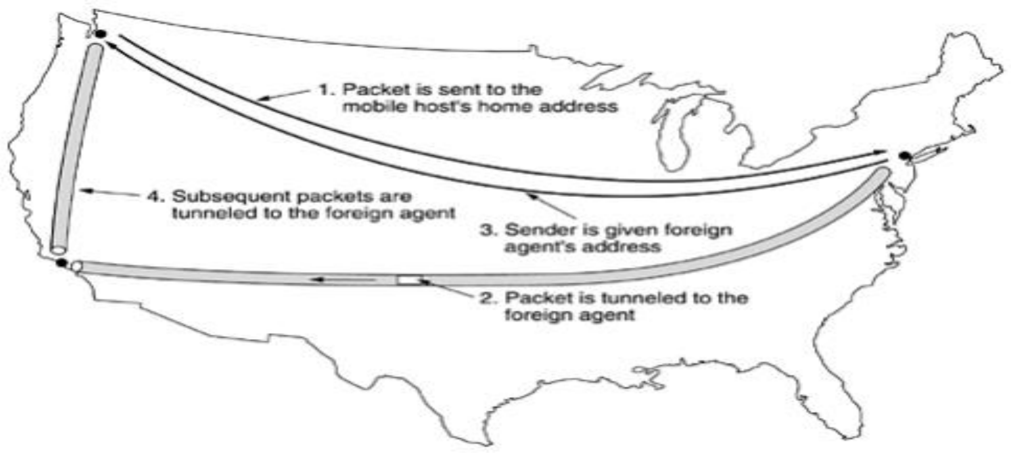


Routing of packets

- When a packet is sent to a mobile host, it is routed to the host's home LAN

- The home agent then looks up the mobile host's new (temporary) location and finds the address of the foreign agent handling the mobile host.
- The home agent then does two things. First, it encapsulates the packet in the payload field of an outer packet and sends the latter to the foreign agent (step 2 in Fig.). This mechanism is called tunneling;
- After getting the encapsulated packet, the foreign agent removes the original packet from the payload field and sends it to the mobile host as a data link frame
- Second, the home agent tells the sender to henceforth send packets to the mobile host by encapsulating them in the payload of packets explicitly addressed to the foreign agent instead of just sending them to the mobile host's home address (step 3).

- Subsequent packets can now be routed directly to the host via the foreign agent (step 4), bypassing the home location entirely.

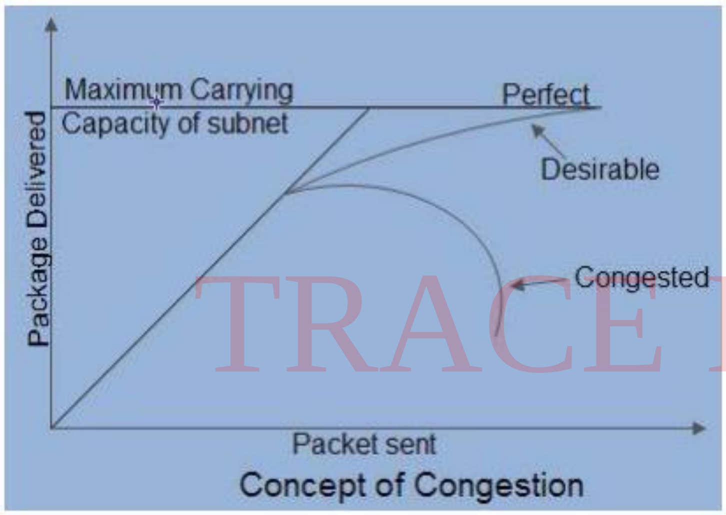

1. Packet is sent to the mobile host's home address
4. Subsequent packets are tunneled to the foreign agent
3. Sender is given foreign agent's address
2. Packet is tunneled to the foreign agent

Packet routing for mobile hosts.

CONGESTION CONTROL ALGORITHMS

CONGESTION

- 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.



Causes of Congestion

The following factors are responsible for congestion.

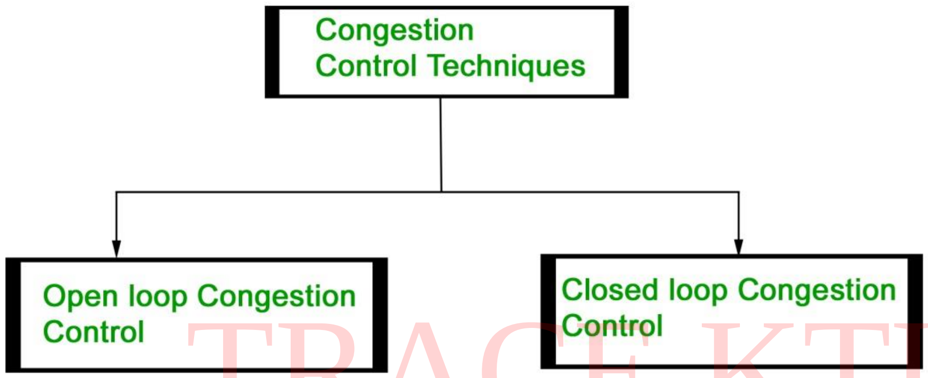
• Slow network links : Low-Bandwidth lines can also cause congestion.

• Shortage of buffer space: if all of a sudden a stream of packets arrive on several input lines and need to be out on the same output line, then a long queue will be build up for that output If there is insufficient memory to hold these packets, then packets will be lost (dropped).

• Slow processor: If the router CPU is slow at performing the task required for them (Queuing buffers, updating tables, reporting any exceptions etc.), queue can build up even if there is excess of line capacity.

CONGESTION CONTROL POLICIES

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


Congestion Control Techniques
Open loop Congestion Control
Closed loop Congestion Control

Open Loop Congestion Control

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

Policies adopted by open loop congestion control –

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

2. **Window Policy :**

The type of window at the sender side may also affect the congestion. Several packets in the Go-back-n window are resent, although some packets may be received successfully at the receiver side. This duplication may increase the congestion in the network and making it worse.

Therefore, Selective repeat window should be adopted as it sends the specific packet that may have been lost.

3. **Discarding Policy** :

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

In case of audio file transmission, routers can discard less sensitive packets to prevent congestion and also maintain the quality of the audio file.

4. **Acknowledgment Policy :**

Since acknowledgement are also the part of the load in network, the acknowledgment policy imposed by the receiver may also affect congestion. Several approaches can be used to prevent congestion related to acknowledgment.

The receiver should send acknowledgement for N packets rather than sending acknowledgement for a single packet. The receiver should send a acknowledgment only if it has to sent a packet or a timer expires.

5. **Admission Policy :**

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

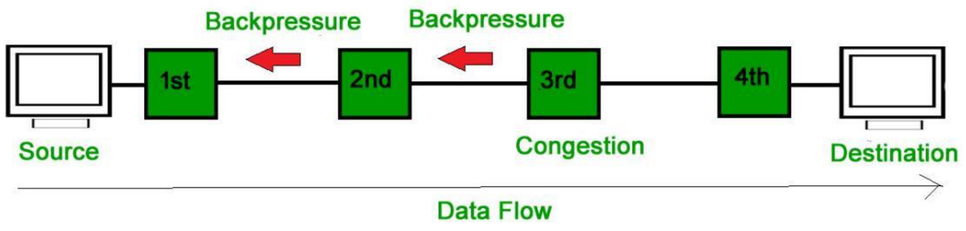
All the above policies are adopted to prevent congestion before it happens in the network.

Closed Loop Congestion Control

Closed loop congestion control technique is used to treat or alleviate congestion after it happens. Several techniques are used by different protocols; some of them are:

Back pressure :

- Backpressure is a technique in which a congested node stop receiving packet from upstream node. This may cause the upstream node or nodes to become congested and rejects receiving data from above nodes.
- Backpressure is a node-to-node congestion control technique that propagates in the opposite direction of data flow. The backpressure technique can be applied only to virtual circuit where each node has information of its above upstream node.

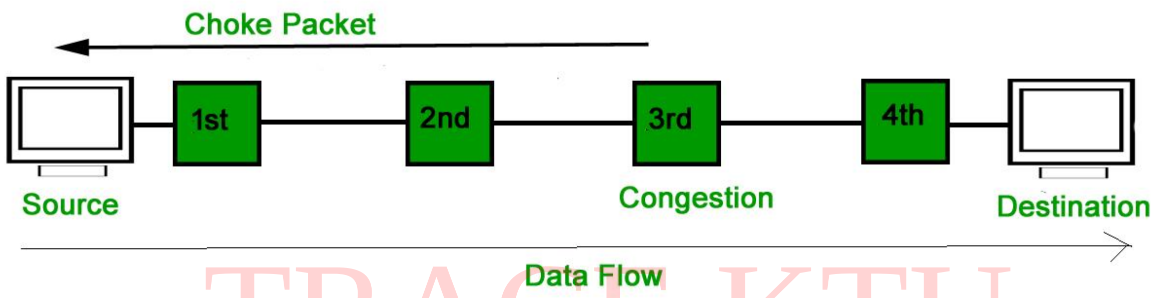


In above diagram the 3rd node is congested and stops receiving packets as a result 2nd node may be get congested due to slowing down of the output data flow. Similarly 1st node may get congested and informs the source to slow down.

Choke Packet Technique :

- Choke packet technique is applicable to both virtual networks as well as datagram subnets.

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



Implicit Signaling :

- ➢ In implicit signaling, there is no communication between the congested nodes and the source.
- ➢ The source guesses that there is congestion in a network.
- ➢ For example when sender sends several packets and there is no acknowledgment for a while, one assumption is that there is a congestion.

Explicit Signaling :

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

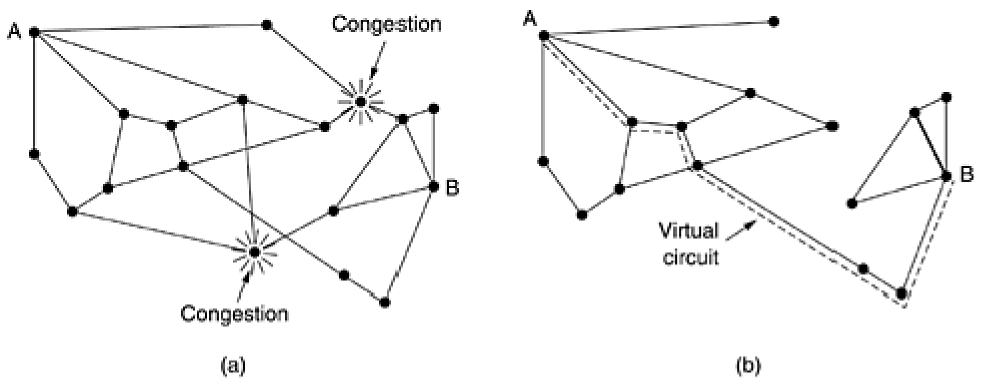
Explicit signaling can occur in either forward or backward direction.

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

CONGESTION CONTROL IN VIRTUAL CIRCUIT NETWORK

- One technique that is widely used to keep congestion that has already started from getting worse is admission control. The idea is simple: once congestion has been signaled, no more virtual circuits are set up until the problem has gone away.
  In the telephone system, when a switch gets overloaded, it also practices admission control by not giving dial tones.

- An alternative approach is to allow new virtual circuits but carefully route all new virtual circuits around problem areas. For example, consider the subnet in which two routers are congested, as indicated.



Suppose that a host attached to router A wants to set up a connection to a host attached to router B. Normally, this connection would pass through one of the congested routers. To avoid this situation, we can redraw the subnet as shown in Fig, omitting the congested routers and all of their lines. The dashed line shows a possible route for the virtual circuit that avoids the congested routers.

- Another strategy relating to virtual circuits is to negotiate an agreement between the host and subnet when a virtual circuit is set up. This agreement normally specifies the volume and shape of the traffic, quality of service required, and other parameters. To keep its part of the agreement, the subnet will typically reserve resources along the path when the circuit is set up.
- These resources can include table and buffer space in the routers and bandwidth on the lines. In this way, congestion is unlikely to occur on the new virtual circuits because all the necessary resources are guaranteed to be available.
  - This kind of reservation can be done all the time as standard operating procedure or only when the subnet is congested. A disadvantage of doing it all the time is that it tends to waste resources.

CONGESTION CONTROL IN DATAGRAM NETWORK

The Warning Bit

- The old DECNET architecture signaled the warning state by setting a special bit in the packet's header. So does frame relay.
- When the packet arrived at its destination, the transport entity copied the bit into the next acknowledgement sent back to the source. The source then cut back on traffic.
- As long as the router was in the warning state, it continued to set the warning bit, the source continued to decrease its transmission rate.
- Since every router along the path could set the warning bit, traffic increased only when no router was in trouble.

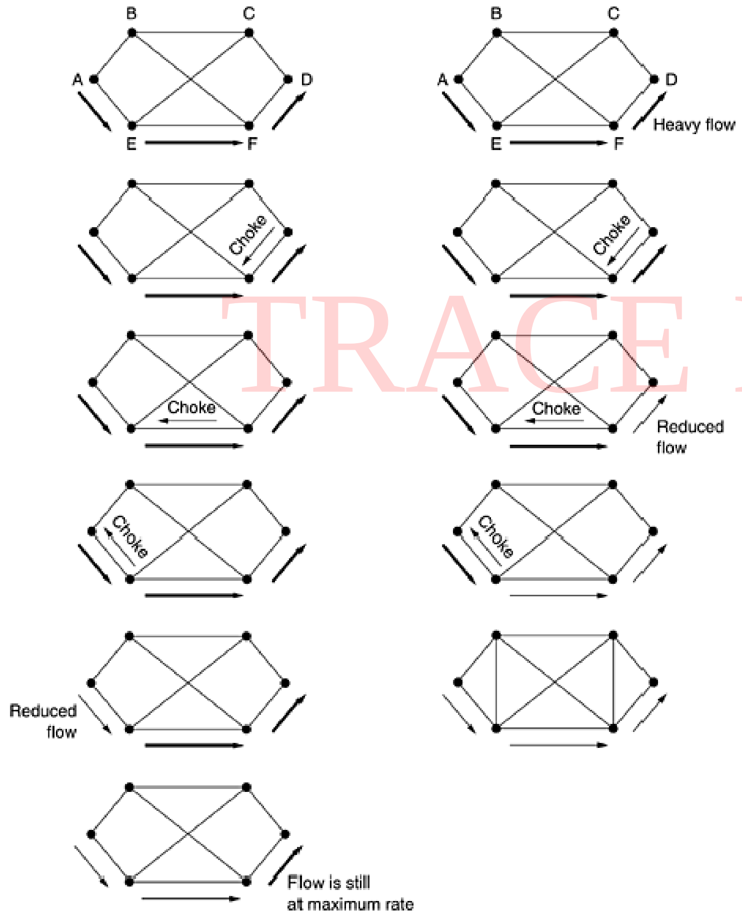
Choke Packets

- In this approach, the router sends a choke packet back to the source host,
- When the source host gets the choke packet, it is required to reduce the traffic sent to the specified destination by X percent.
- Since other packets aimed at the same destination are probably already under way and will generate yet more choke packets, the host should ignore choke packets referring to that destination for a fixed time interval.
- After that period has expired, the host listens for more choke packets for another interval. If one arrives, the line is still congested, so the host reduces the flow still more and begins ignoring choke packets again.
- If no choke packets arrive during the listening period, the host may increase the flow again.

Hop-by-Hop Choke Packets

- At high speeds or over long distances, sending a choke packet to the source hosts does not work well because the reaction is so slow.

- An alternative approach is to have the choke packet take effect at every hop it passes through, as shown in the sequence of Fig. 5-28(b).
- Here, as soon as the choke packet reaches F, F is required to reduce the flow to D. Doing so will require F to devote more buffers to the flow, since the source is still sending away at full blast, but it gives D immediate relief, like a headache remedy in a television commercial. In the next step, the choke packet reaches E, which tells E to reduce the flow to F. This action puts a greater demand on E's buffers but gives F immediate relief. Finally, the choke packet reaches A and the flow genuinely slows down.

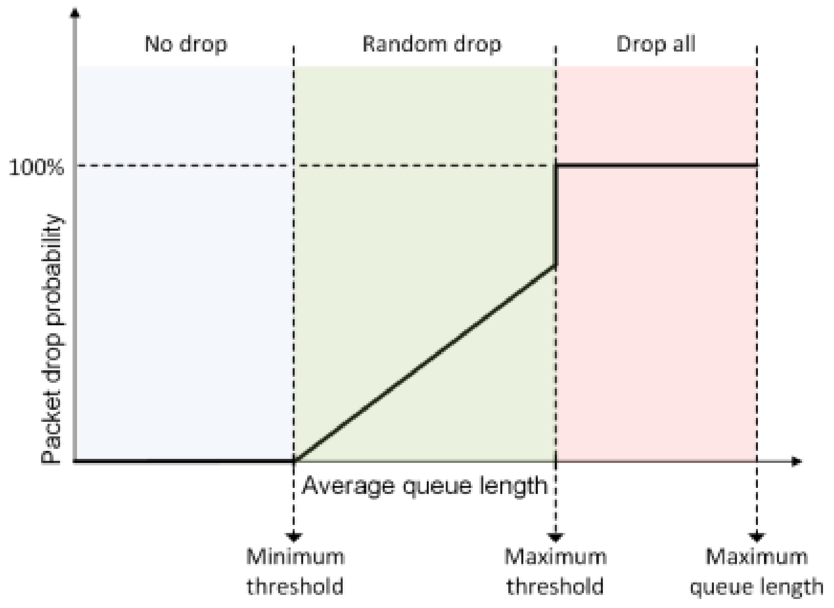


LOAD SHEDDING

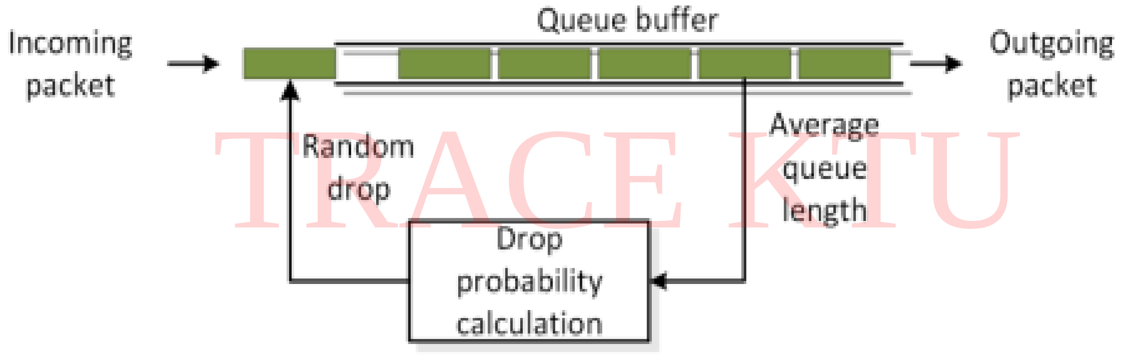
- It is one of the approaches to congestion control.
- Router contains a buffer to store packets and route it to destination.
- When the buffer is full, it simply discards some packets. It chooses the packet to be discarded based on the strategy implemented in the data link layer. This is called load shedding
- Load shedding will use **dropping the old packets than new to avoid congestion**. Dropping packets that are part of the difference is preferable because a future packet depends on full frame.
- To implement an intelligent discard policy, applications must mark their packets to indicate to the network how important they are. When packets have to be discarded, routers can first drop packets from the least important class, then the next most important class, and so on.
- For a file transfer, for, e.g. cannot discard older packets since this will cause a gap in the received data. This policy is often called as wine.
- For real-time voice or video it is probably better to throw away old data and keep new packets. This policy is often known as milk.

Random Early Detection

- This is a proactive approach in which the router discards one or more packets before the buffer becomes completely full.
- Each time a packet arrives, the RED algorithm computes the average queue length, avg.
- If avg is lower than some lower threshold, congestion is assumed to be minimal or non-existent and the packet is queued.
- If avg is greater than some upper threshold, congestion is assumed to be serious and the packet is discarded.
- If avg is between the two thresholds, this might indicate the onset of congestion. The probability of congestion is then calculated.



(b)

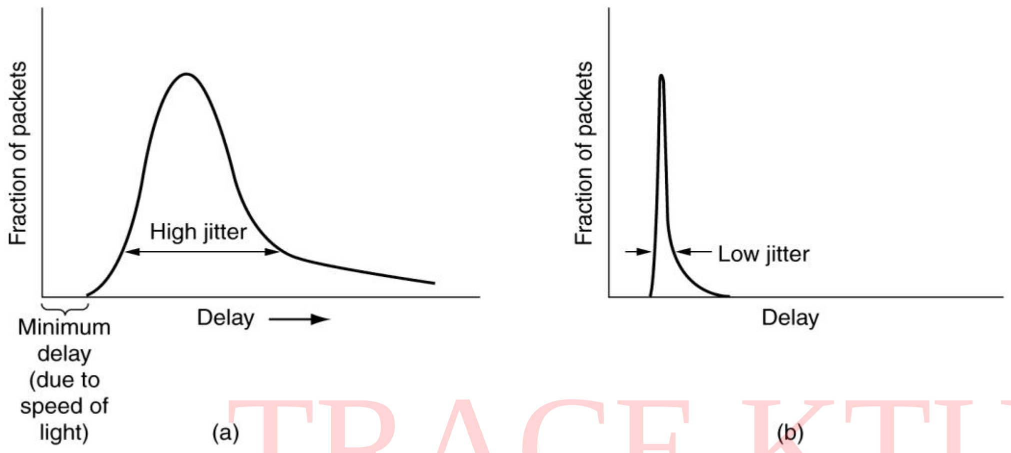


(a)

Jitter control

- The variation (i.e., standard deviation) in the packet arrival times is called jitter.
- High jitter, for example, having some packets taking 20 msec and others taking 30 msec to arrive will give an uneven quality to the sound or movie.
- The jitter can be bounded by computing the expected transit time for each hop along the path.

- When a packet arrives at a router, the router checks to see how much the packet is behind or ahead of its schedule. This information is stored in the packet and updated at each hop.
- If the packet is ahead of schedule, it is held just long enough to get it back on schedule. If it is behind schedule, the router tries to get it out the door quickly.



QUALITY OF SERVICE(QoS)

- A Stream of packets from a source to destination is called flow.
- In a connection oriented network all packets belonging to a flow follow the same route, in a connection-less service they may follow different routes
- The needs of each flow can be characterized by primary parameters reliability, delay, jitter and bandwidth. Together these determine the QoS(Quality of Service) the flow requires.
- QoS defines a set of attributes related to the performance of the connection
- Flow Characteristics
  - Reliability: Reliability is a characteristic that flow needs.
    - lack of reliability means losing a packet or acknowledgment which entails retransmission. However, the sensitivity of application programs to reliability is not the same
  - Delay: Source-to-destination delay is another characteristic Again application can tolerate delay in different degree. In this case, telephony, audio conferencing, video conferencing and remote login need minimum delay, while delay in file transfer or e-mail is less important
  - Jitter: **Jitter is the variation in delay for packets belonging to the same flow**. For example, if four packets depart at times 0,1,2 and 3 and arrive at 20,21,22 and 23, all have the same delay, 20 units of time. Jitter is defined as the variation in the packet delay. High jitter means the difference between delays is large; low jitter means the variation is small
  - Bandwidth: Different applications need different bandwidths. In video conferencing one need to send millions of bits per second to refresh a color screen while the total number of bits in an e-mail may not reach even a million.

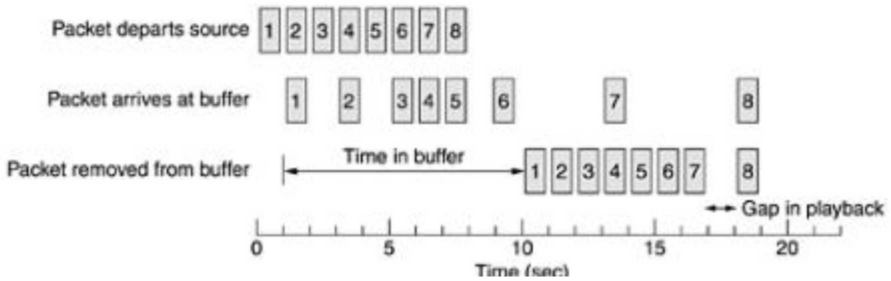
Techniques for Achieving Good Quality of Service

Overprovisioning

- An easy solution is to provide so much router capacity, buffer space, and bandwidth that the packets just fly through easily.
- The trouble with this solution is that it is expensive.
- The telephone system is overprovisioned.

Buffering

- Flows can be buffered on the receiving side before being delivered.
- Buffering them does not affect the reliability or bandwidth, and increases the delay, but it smooths out the jitter.
- For audio and video on demand, jitter is the main problem, so this technique helps.



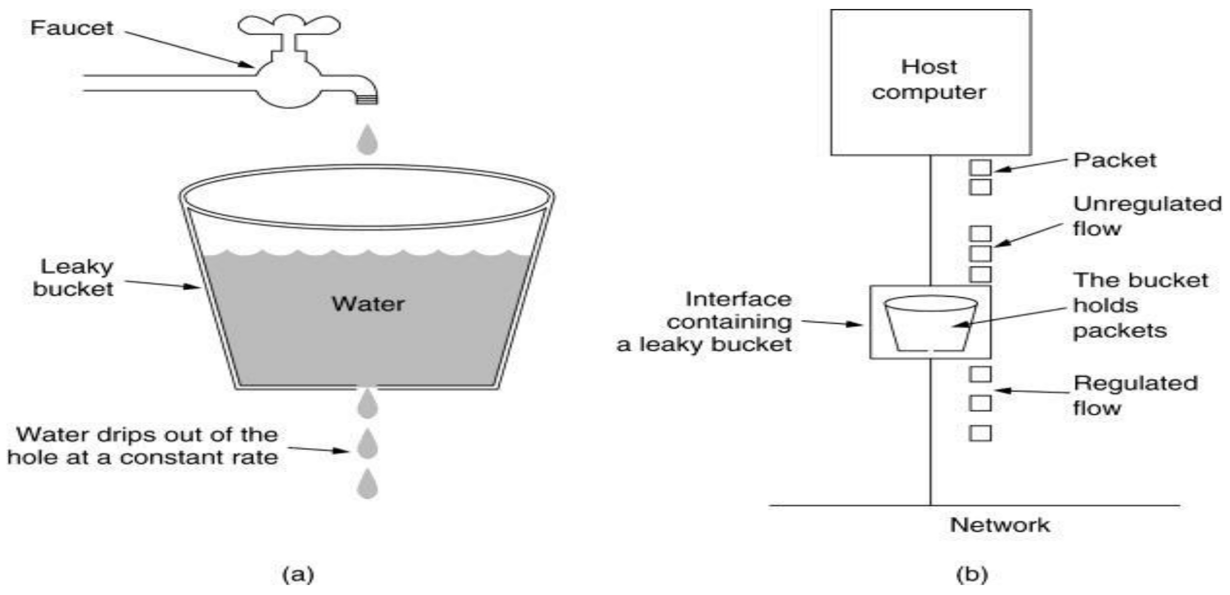
Traffic shaping

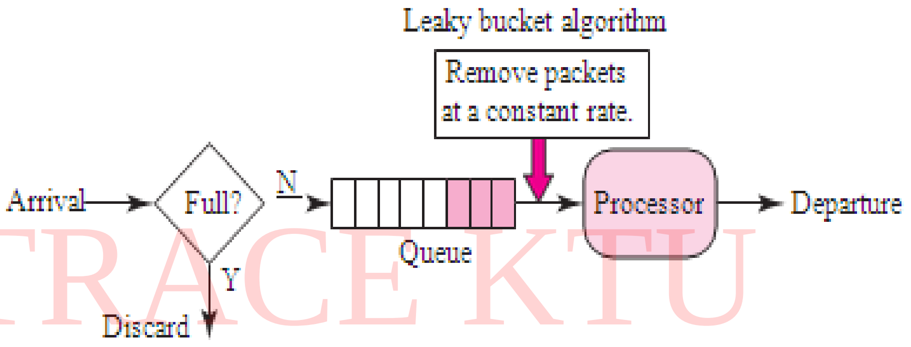
- One of the main **causes** of congestion is that traffic is often bursty.
- **Traffic shaping** is about regulating the **average rate** (and burstiness) of data transmission.

- **Monitoring a traffic flow** is called **traffic policing**.
- Agreeing to a traffic shape and policing it afterward are easier with virtual-circuit subnets than with datagram subnets.
- When a connection is set up, the **user** and **the subnet**(i.e., the customer and the carrier) agree on a certain traffic pattern (i.e., shape) for that circuit. Sometimes this is called a **service level agreement**.

Leaky bucket algorithm

- Imagine a bucket with **a small hole** in the bottom, as illustrated in Fig.4-4(a).
- No matter the rate at which water enters the bucket, the **outflow** is at a constant rate when there is any water in the bucket and zero when the bucket is empty.
- Also, once the bucket is full, any additional water entering it spills over the sides and is lost
- In practice the **bucket** is a finite queue that **outputs** at a finite rate.
- A leaky bucket algorithm shapes bursty traffic into fixed-rate traffic by averaging the data rate.
- It may drop the packets if the bucket is full.





- This can be implemented as follows.

➢ On a **sender**, a time-dependent queue ensures that an equal amount of data units is send per time interval.

➢ On the one hand, data can be put fast into the queue until it is full.

➢ On the other hand, data always leave the queue at the same rate.

➢ If the **queue is full** no more packets can be stored in it and packet loss occurs.

➢ If the **queue is empty**, no data leaves the queue.

➢ The Leaky Bucket algorithm can be implemented for packets or a constant amount of bytes, send within each time interval.

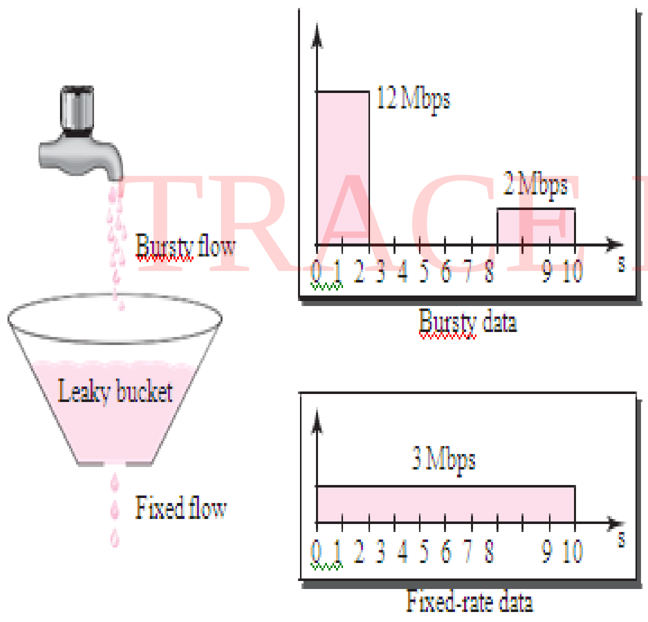
- Conceptually each **network interface** contains a leaky bucket. And the following steps are performed:

✓ When the **host** has to send a packet, the packet is thrown into the bucket.

✓ The **bucket** leaks at a constant rate, meaning the network interface transmits packets at a constant rate.

✓ **Burst traffic** is converted to a **uniform traffic** by the leaky bucket

Example

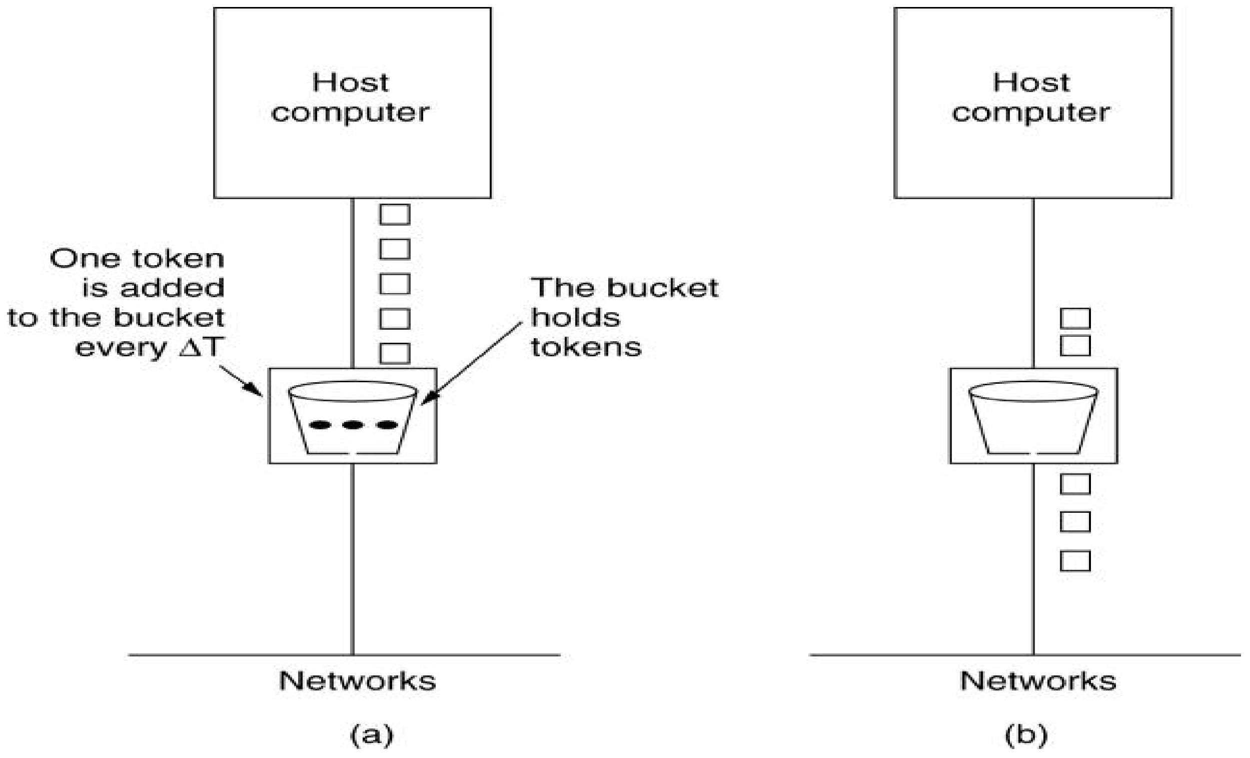


- In the figure, we assume that the network has committed a **bandwidth** of 3 Mbps for a host.
- The use of the leaky bucket shapes the input traffic to make it conform to this commitment.
- In Figure the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data.

- 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 10 s.
- The **leaky bucket** smooths the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

Tocken bucket algorithm

- The leaky bucket algorithm enforces a **rigid output pattern** at the **average rate**, no matter how bursty the traffic is.
- For many applications, it is better to allow the output to speed up somewhat when large bursts arrive.
- One such algorithm is the **token bucket algorithm**.
- In this algorithm, the **leaky bucket** holds **tokens**, generated by a clock at the rate of **one token every T sec.**
- In Fig. 4-5(a) a **bucket** is holding three tokens, with five packets waiting to be transmitted.
- For a **packet** to be transmitted, it must capture and destroy one token.
- In Fig. 4-5(b) three of the five packets have gotten through, but the other two are stuck waiting for two more tokens to be generated.



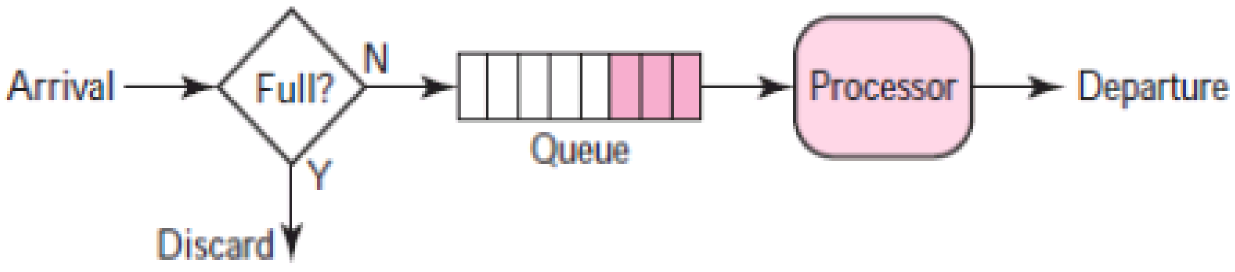
Packet scheduling

- 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.
- Three of them here:

1. FIFO queuing,
2. priority queuing,
3. weighted fair queuing.

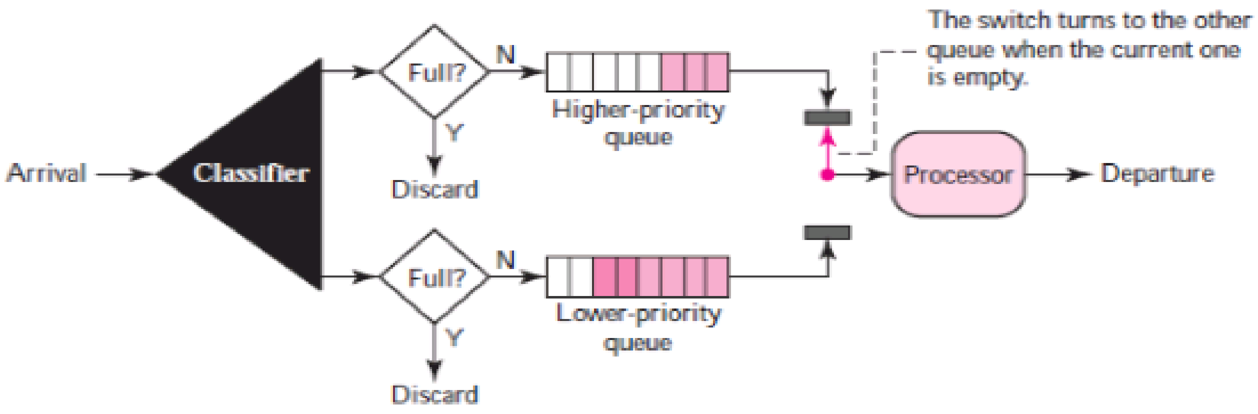
1. **FIFO Queuing:**

- **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.



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.



Weighted Fair Queuing:

- A better scheduling method is weighted fair queuing.
- In this technique, the **packets** are still assigned to different classes and admitted to different queues.
- The queues, however, are **weighted** based on the priority of the queues;

➢ higher priority means a higher weight.

- The system processes packets in each queue in a **round-robin fashion** with the number of packets selected from each queue based on the corresponding weight.

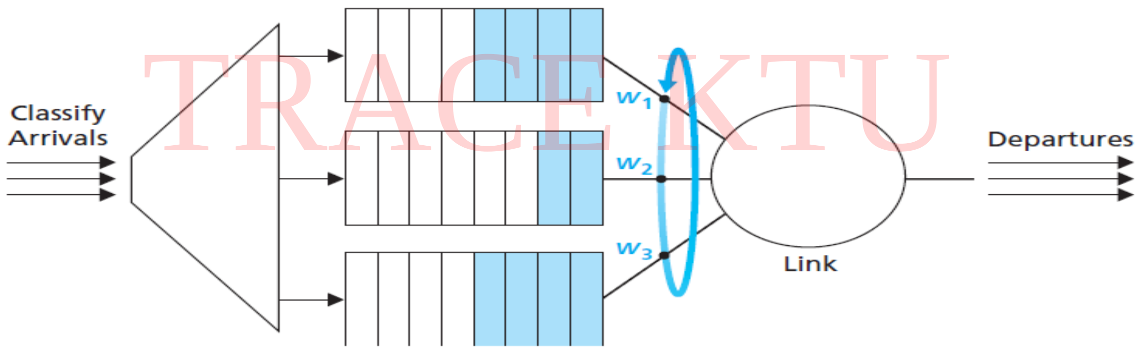


Figure Weighted fair queuing (WFQ)