THE NETWORK LAYER

3.1 NETWORK LAYER DESIGN ISSUES
1. Store-and-Forward Packet Switching
2. Services Provided to the Transport Layer
3. Implementation of Connectionless Service
4. Implementation of Connection-Oriented Service
5. Comparison of Virtual-Circuit and Datagram Networks

3.2 ROUTING ALGORITHMS
1. The Optimality Principle
2. Shortest Path Algorithm
3. Flooding
4. Distance Vector Routing
5. Link State Routing
6. Hierarchical Routing
7. Broadcast Routing
8. Multicast Routing
9. Anycast Routing
10. Routing for Mobile Hosts
11. Routing in Ad Hoc Networks

3.3 CONGESTION CONTROL ALGORITHMS
1. Approaches to Congestion Control
2. Traffic-Aware Routing
3. Admission Control
4. Traffic Throttling
5. Load Shedding

3.4 QUALITY OF SERVICE
1. Application Requirements
2. Traffic Shaping
3. Packet Scheduling
4. Admission Control
5. Integrated Services
6. Differentiated Services

MODULE-3

THE NETWORK LAYER

Network Layer is majorly focused on getting packets from the source to the destination, routing error handling and congestion control.

NETWORK LAYER DESIGN ISSUES:
The network layer comes with some design issues they are described as follows:

1. Store and Forward packet switching.
2. Services provided to Transport Layer.
3. Implementation of Connectionless Service.
4. Implementation of Connection Oriented service.
5. Comparison of Virtual-Circuit and Datagram Networks.

1. Store and Forward packet switching:

The host sends the packet to the nearest router. This packet is stored there until it has fully arrived once the link is fully processed by verifying the checksum then it is forwarded to the next router till it reaches the destination. This mechanism is called "Store and Forward packet switching."

2. Services provided to Transport Layer:

Through the network/transport layer interface, the network layer transfers its services to the transport layer. These services are described below.
But before providing these services to the transfer layer following goals must be kept in mind:- • Offering services must not depend on router technology.

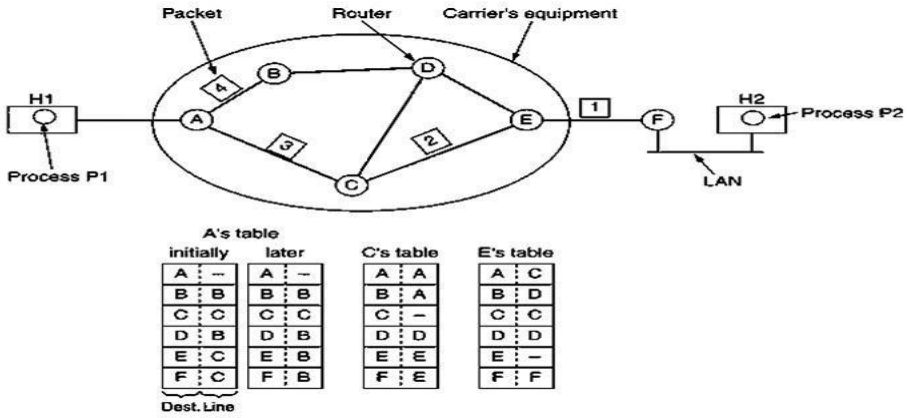
- The transport layer needs to be protected from the type, number and topology of the available router.
- The network addresses for the transport layer should use uniform numbering pattern also at LAN and WAN connections.

Based on the connections there are 2 types of services provided:

- **Connectionless –** The routing and insertion of packets into subnet is done individually. No added setup is required.
- **Connection-Oriented –** Subnet must offer reliable service and all the packets must be transmitted over a single route.

3. Implementation of Connectionless Service:

Packet are termed as "datagrams" and corresponding subnet as "datagram subnets". When the message size that has to be transmitted is 4 times the size of the packet, then the network layer divides into 4 packets and transmits each packet to router via. a few protocol.Each data packet has destination address and is routed independently irrespective of the packets.



Let us discuss how datagram network works in stepwise manner −

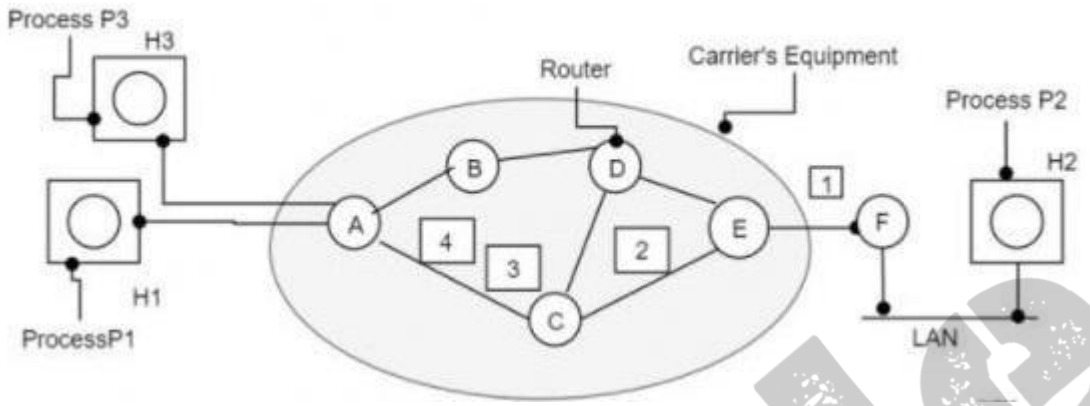
- Suppose there is a process P1 on host H1 and is having a message to deliver to P2 on host H2. P1 hands the message to the transport layer along with instructions to be delivered to P2 on H2.
- Transport Layer code is running on H1 and within the operating system. It prepends a transport header to the message and the end result is given to the network layer.
- Let us assume for this example a packet which is four times heavier than the maximum size of the packet, then the packet is broken to four different packets and each of the packet is sent to the router A using point to point protocol and from this point career takes over.
- Each router will have an internal table saying where packets to be sent. Every table entry is a pair consisting of a destination and outgoing line to use for that destination. Only directly connected lines can be used.
- For example A has only two outgoing lines to B and C, therefore every incoming packet must be sent to one of these routers, even if the ultimate destination will be some other router.
- As the packets arrived at A, packet 1,2,3 and 4 were stored in brief. Then every packet is moved to C as per A's table. Packet 1 is forwarded to E and then moved to F. When packet 1 is moved to F, then it will be encapsulated in a data link layer and sent to H2 over to LAN. Packet 2 and 3 will also follow the same route.
- When packet 4 reaches A, then it was sent to router B, even if the destination was F. For some purpose A decided to send packet 4 through a different route. It was because of the traffic jam in ACE path and the routing table was updated. Routing Algorithm decides routes, makes routing decisions and manages routing tables.

4. Implementation of Connection Oriented service:

To use a connection-oriented service, first we establishes a connection, use it and then release it. In connection-oriented services, the data packets are delivered to the receiver in the same order in which they have been sent by the sender. It can be done in either two ways :

- **Circuit Switched Connection –** A dedicated physical path or a circuit is established between the communicating nodes and then data stream is transferred.
- **Virtual Circuit Switched Connection –** The data stream is transferred over a packet switched network, in such a way that it seems to the user that there is a dedicated path from the sender to the receiver. A virtual path is established here. While, other connections may also be using the same path.

The implementation of connection-oriented services is diagrammatically represented as follows −



Consider the scenario as mentioned in the above figure.

- Host H1 has established connection 1 with host H2, which is remembered as the first entry in every routing table.
- The first line of A's infers when packet is having connection identifier 1 is coming from host H1 and has to be sent to router W and given connection identifier as 1.
- Similarly, the first entry at W routes the packet to Y, also with connection identifier 1.
- If H3 also wants to establish a connection to H2 then it chooses connection identifier 1 and tells the subnet to establish the virtual circuit. This will be appearing in the second row in the table.
- Note that we have a conflict here because although we can easily distinguish connection 1 packets from H1 from connection 1 packet from H3, W cannot do this.
- For this reason, we assign 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 Networks**

Issue	Dumb 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 a 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 or 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

ROUTING ALGORITHMS

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

1.The Optimality Principle

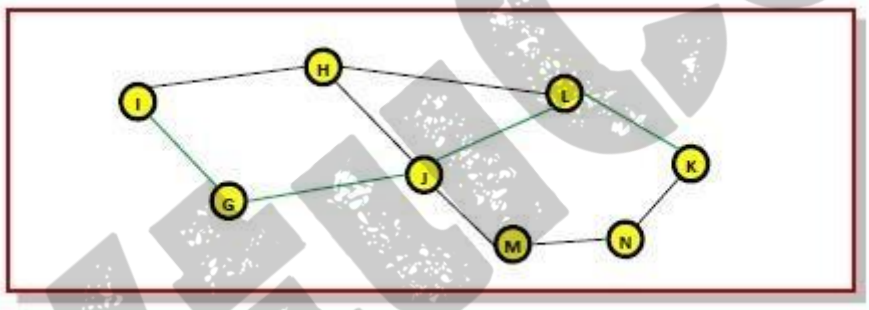
The purpose of a routing algorithm at a router is to decide which output line an incoming packet should go. The optimal path from a particular router to another may be the least cost path, the least distance path, the least time path, the least hops path or a combination of any of the above.

The optimality principle can be logically proved as follows −

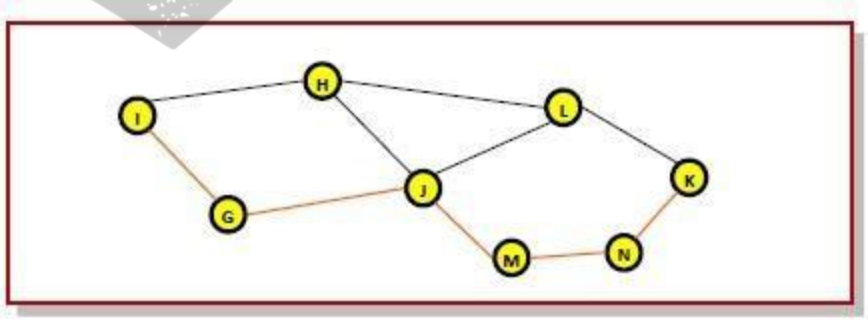
- If a better route could be found between router J and router K, the path from router I to router K via J would be updated via this route. Thus, the optimal path from J to K will again lie on the optimal path from I to K.

Example

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



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



2.Shortest Path Algorithm

Dijkstra's algorithm finds the shortest path from a source node to all other nodes in a weighted graph by iteratively selecting the node with the smallest tentative distance and updating the distances to its neighbours.
It ensures the shortest path is progressively discovered and is based on the principle of ***greedy optimization****.*

Algorithm:

- Create a Set **sptSet** (shortest path tree set) that keeps track of vertices included in the shortest path tree, i.e., whose minimum distance from the source is calculated and finalized. Initially, this set is empty.
- Assign a distance value to all vertices in the input graph. Initialize all distance values as INFINITE. Assign the distance value as **0** for the source vertex so that it is picked first.
- While Set doesn't include all vertices
- Pick a vertex '**u**' that is not there in Set and has a minimum distance value.
- Include '**u**' to **sptSet**.
- Then update the distance value of all adjacent vertices of **u**.
- To update the distance values, iterate through all adjacent vertices.
- For every adjacent vertex **v**, if the sum of the distance value of **u** (from source) and weight of edge **u-v**, is less than the distance value of **v**, then update the distance value of **v.**

```
#define MAX NODES 1024 /* maximum number of nodes */
#define INFINITY 1000000000 /* a number larger than every maximum
path */ int n, dist[MAX NODES][MAX NODES]; /* dist[i][j] is the
distance from i to j */ void shortest path(int s, int t, int path[]) { struct state
{ /* the path being worked on */ int predecessor; /* previous node */ int
length; /* length from source to this node */
enum {permanent, tentative} label; /* label state */
} state[MAX NODES];
int   i,   k,   min;
struct state *p;
for (p = &state[0]; p < &state[n]; p++) { /* initialize
state */ p->predecessor = −1; p->length = INFINITY;
p->label = tentative;
}
state[t].length = 0; state[t].label = permanent;
k = t; /* k is the initial working node */ do {
/* Is there a better path from k? */ for (i = 0;
i < n; i++) /* this graph has n nodes */ if
(dist[k][i] != 0 && state[i].label == tentative)
{ if (state[k].length + dist[k][i] <
state[i].length) { state[i].predecessor = k;
state[i].length = state[k].length + dist[k][i];
}
```

```
}
/* Find the tentatively labeled node with the smallest
label. */ k = 0; min = INFINITY; for (i = 0; i < n; i++)
if (state[i].label == tentative && state[i].length <
min) { min = state[i].length; k = i;
}
state[k].label = permanent;
} while (k != s);
/* Copy the path into the output array.
*/ i = 0; k = s;
do {path[i++] = k; k = state[k].predecessor; } while (k >=
0); }
```

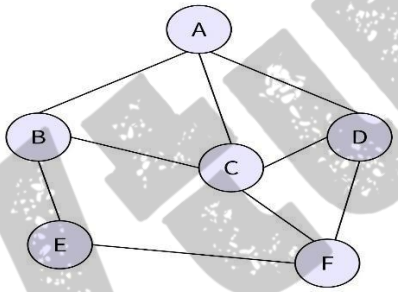
Figure: Dijkstra's algorithm to compute the shortest path through a graph.

3. Flooding

Flooding is a **static routing technique**, based on the following principle:

"When a packet reaches the router, it is transferred to all the outgoing links, except only the link that it has reached the router through.**"**

Flooding is used in routing protocols such as O.S.P.F. (Open Shortest Path First), peer-to-peer file transfers, systems such as Usenet, bridging, etc. Let us have a look at an example for a better understanding. Assume there is a network with **6 routers** connected through transmission lines, as shown in the figure ahead.



Following are the Events That Take Place in Flooding

- Any packet incoming to A is sent to D, C, and B.
- B sends this packet to E and C.
- C sends this packet to F, D, and B.
- D sends this packet to F and C.
- E sends the packet to F.
- F sends the packet to E and C.

Types of Flooding

Flooding in computer networking is usually of the following **three kinds**:

- **Uncontrolled Flooding**

In uncontrolled flooding, every **node distributes the packets unconditionally** to all its neighbors. Broadcast storms become a catastrophe in the absence of conditional logic for the **prohibition of indefinite recirculation** of the same packet.

- **Controlled Flooding**

Controlled flooding is **more reliable**, thanks to the two algorithms that it has:

- **S.N.C.F. (Sequence Number Controlled Flooding)**

As each node possesses **sequence numbers and memory of addresses**, it sticks its very own sequence number and addresses with the packet. If a packet arrives at a node that has the packet in memory already, the node immediately drops the packet.

- **R.P.F. (Reverse Path Forwarding)**

A node transfers a packet only in the **forward direction** to the next node. If the packet came from the next node, it is **returned to the sender**.

- **Selective Flooding**

It is a version of flooding that only **sends packets to routers in the same direction**. Instead of transferring each incoming packet on every line, the routers transmit the packets on only those lines that are approximately going in the **right direction**.

Characteristics of Flooding

Following are some features of flooding:

- Every possible route between the source and the destination for transmission is tried in flooding.
- There always exists a **minimum of one route** which is the shortest.
- Any node that is **connected**, whether **directly or indirectly**, is explored.
- Flooding does not require any information related to the network, such as the costs of various paths, load conditions, topology, etc. This is why it is **non-adaptive**.

Advantages of Flooding

- Flooding always attempts to select the **shortest path**.
- Since any node connected **directly or indirectly** is explored, there is no probability of any node being missed out.
- Flooding is **very robust**. Even if several routers malfunction, the packets still find a path to their destinations.
- Flooding is quite easy to implement and set up, as a router can know only about its neighbors.

Disadvantages of Flooding

- **Repetitive and unauthorized packets** of data can jam a network. This could result in hampering other data packets.
- Flooding becomes **inefficient** in case only one destination requires the packet because flooding unconditionally transmits the data packet to all nodes.
- Flooding aids in generating an **infinite number of duplicate data packets** unless there exists some logic to enforce a limit upon the creation of packets.
- Flooding is expensive in terms of the bandwidth that it wastes. A message may have a single destination, yet it has to be sent to all the hosts.
- When a D.O.S. (Denial Of Service) attack or ping flood occurs, flooding may harm the reliability of the network.

4. Distance Vector Routing Algorithm

Distance vector routing algorithm is also called as **Bellman-Ford algorithm** or **Ford Fulkerson algorithm** as this algorithm is used to find the shortest route from one node to another node in the network.

The routing protocol is used to calculate the best route from source to destination based on the distance or hops as its primary metric to define an optimal path. The distance vector refers to the distance to the neighbor nodes, where routing defines the routes to the established node.

The **Distance Vector routing algorithm**(DVR) shares the information of the routing table with the other routers in the network and keeps the information up-to-date to select an optimal path from source to destination.

A few key points about the distance vector routing protocol:

- **NetworkInformation:**
  Every node in the network should have information about its neighboring node. Each node in the network is designed to share information with all the nodes in the network.
- **RoutingPattern:**
  In DVR the data shared by the nodes are transmitted only to that node that is linked directly to one or more nodes in the network.
- **Datasharing:**
  The nodes share the information with the neighboring node from time to time as there is a change in network topology.

The Bellman-Ford algorithm is defined as:
$d_x(y)=\min_v\{c(x,v)+d_v(y)\}$
where, $d_x(y)$= The least distance from x to y
$c(x,v)$= Node x's cost from each of its neighbor
v $d_v(y)$= Distance to each node from initial
node min_v= selecting shortest distance

Consider a scenario where all the routers are set and run the distant vector routing algorithm. Each router in the network will share the distance information with the neighboring router. All the information is

gathered from the neighbor routers. With each router's information, an optimal distance is calculated and stored in the routing table. This way, the process of calculating the optimal path is done using the distant vector routing protocol.

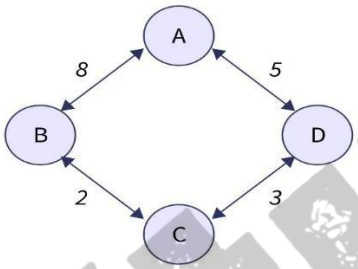
How Does DVR Protocol Work?

The distance vector routing algorithm works by having each router maintain a routing table, giving the bestknown distance from source to destination and which route is used to get there.

These tables are updated by exchanging the information with the neighbor having a direct link. Tables contain one entry for each route, this entry contains two parts, the preferred outgoing line used to reach the destination or an estimate of the time or distance to that destination.

The metric used can be the number of hops required to reach from source to destination. Time delay in milliseconds, the router can measure it with a special echo signal which the receiver can timestamp and send as soon as possible.

The router exchanges the network topology information periodically with its neighboring node and updates the information in the routing table. The cost of reaching the destination is estimated based on the metric, and an optimal path is obtained to send data packets from source to destination. **Example of Distance Vector Routing**



Step - 1

As we can see in the above diagram of the DVR network, the routers in the network start sharing their information with the neighboring routers in the network.

Routing table of A:

Destination	**distant**	**Hop**
A	0	A
B	8	B
C	infinity	-
D	5	D

Routing table of B:

Destination	distant	Hop
A	8	A
B	0	B
C	2	C
Destination	**distant**	**Hop**
D	infinity	D

Routing table of C:

Destination	distant	Hop
A	infinity	-
B	2	B
C	0	C
D	3	D

Routing table of D :

Destination	distant	Hop
A	5	A
B	infinity	B
C	3	C
D	0	D

Step - 2

After creating the separate local table this information is shared with the neighboring node having a direct link.

For Router A:

The router A has a direct connection to neighboring routers B and D.

Destination	Vector B	Vector D
A	8	5
B	0	infinity

C	2	3
D	infinity	0

So based on the vector information of the neighboring router, the value of the router is updated.

Here the optimal distance is been calculated to reach the specific destination. First, the distance from A to B is identified, in our case which is 8, ie *cost*(*A*−>*B*)=8, and *cost*(*A*−>*D*)=5

The distance to reach a destination B from router A is:

A= min{(A->B+B->A),(A->D+D->B}= min{8+0,5+infinity} = min{8,infinity}= 8(B)

Since the cost is min from neighbor B so the router chooses the path from B. It then updates the routing information with entries (8,*B*).

To find a cost to reach destination C from router A we will use a similar approach.

A=min{(A->B+B->C),(A->D+D->C}= min{8+2,5+13} =min{10,8} = 8(C)

Distance to reach destination D from A

A=min{(A->B+B->D),(A->D+D->D}= min{8+infinity,5+0} = min{infinity,5} = 5(D)

Consequently, A's new routing table is:

Destination	**distant**	**Hop**
A	0	A
B	8	B
C	8	D
D	5	D

For router B:

Router B receives information from A and C.

The new routing table for B is calculated as:

Destination	**Vector A**	**Vector C**
A	0	infinity
B	8	2
C	infinity	0

D	5	3

The cost for vector (B->A)=8
The cost for vector (B->C)=2

Distance to reach destination A from router B:

B=min{(B->A+A->A),(B->C+C->A)}=
min { 8+0 , 2+infinity } = min{8, infinity} = 8(A)

Distance to reach destination C from router B:

B=min{(B->A+A->C),(B->C+C->C)}=
min { 8+infinity , 2+0} = min{infinity, 2} = 2(C)

Distance to reach destination D from router B:

B=min{(B->A+A->D),(B->C+C->D)}=
min { 8+5 , 2+3 } = min{13 , 5 } = 5(C)

Consequently, B's new routing table is:

Destination	**distant**	**Hop**
A	8	A
B	0	B
C	2	C
D	5	C

For router C:

The router C receives information from B and D.

The new routing table for C is calculated as:

Destination	**Vector B**	**Vector D**
A	8	5
B	0	infinity
C	2	3
D	infinity	0

The cost(C->B) = 2
The cost(C->D) = 3

Distance to reach destination A from router C:

B=min{(C->B+B->A),(C->D+D->A),}=
min { 2+8 , 3+5 } = min{10,8} = 8(D)

Distance to reach destination B from router C:

B=min{(C->B+B->B),(C->D+D->B),}=
min { 2+0 , 3+infinity } = 2(B)

Distance to reach destination B from router C:

B=min{(C->B+B->D),(C->D+D->D),}=

min { 2+infinity , 3+0 } = 3(D)

Consequently, C's new routing table is:

Destination	**Distant**	**Hop**
A	8	D
B	2	B
C	0	C
D	3	D

For router D:

The router D receives information from A and C.

The new routing table for D is calculated as:

Destination	**Vector A**	**Vector C**
A	0	infinity
B	8	2
C	infinity	0
D	5	3

The cost(D->A) = 5
The cost(D->C) = 3

Distance to reach destination A from router D:

D=min{(D->A+A->A),(D->C+C->A)}= min { 5+0 , 3+infinity } = min{5 + infinity} = 5(A)

Distance to reach destination B from router D:

D=min{(D->A+A->B),(D->C+C->B)}=
min { 5+8 , 3+2, } = min{13,5} = 5(C)

Distance to reach destination C from router D:

D=min{(D->A+A->B),(D->C+C->B)}=
min { 5+infinity , 3+0 } = 3(C)

Consequently, D's new routing table is:

Destination	**distant**	**Hop**
A	5	A
B	5	C
C	3	C
D	0	D

Step - 3

After this, the router again exchanges the distance vector obtained in step 2 with its neighboring router.

After exchanging the distance vector, the router prepares a new routing table.

For router A:

Destination	**Vector B**	**Vector D**
A	8	5
B	0	5
C	2	3
D	5	0

The cost(A->B) = 8
The cost(A->D) = 5

Distance to reach destination B from router A:

B=min{(A->B+B->B),(A->D+D->B)}=
min { 8+0 , 5+5 } = min{8,10} = 8(B)

Distance to reach destination C from router A:

B=min{(A->B+B->C),(A->D+D->C),}=
min { 8+2 , 5+3 } = min{10,8} = 8(D)

Distance to reach destination D from router A:

B=min{(A->B+B->D),(A->D+D->D),}= min { 8+5 , 5+0} = min{13,5} = 5(D)

Consequently, A's new routing table is:

Destination	**distant**	**Hop**
A	0	A
B	8	B
C	8	D
D	5	D

For router B:

The router B receives information from A and C.

The new routing table for B is calculated as:

Destination	**Vector A**	**Vector C**
A	0	8
B	8	2
C	8	0
D	5	3

The cost for vector (B->A)=8
The cost for vector (B->C)=2

Distance to reach destination A From router B:

B=min{(B->A+A->A),(B->C+C->A)}= min { 8+0 , 2+8 } = min{8, 10} = 8(A)

Distance to reach destination C From router B:

B=min{(B->A+A->C),(B->C+C->C)}= min { 8+8 , 2+0 } = 2(C)

Distance to reach destination D From router B:

B=min{(B->A+A->D),(B->C+C->D)}= min { 8+5 , 2+3 } = min{ 13, 5} = 5(C)

Consequently, B's new routing table is:

Destination	Distant	Hop
A	8	A
B	0	B
C	2	C
Destination	**Distant**	**Hop**
D	5	C

For router C:

The router C receives information from B and D.

The new routing table for C is calculated as:

Destination	Vector B	Vector D
A	8	5
B	0	5
C	2	3
D	5	0

The cost(C->B) = 2
The cost(C->D) = 3

Distance to reach destination A from router C:

B=min{(C->B+B->A),(C->D+D->A),}=
min { 2+8 , 3+5 } = min{10,8} = 8(D)

Distance to reach destination B from router C:

B=min{(C->B+B->B),(C->D+D->B),}=
min { 2+0 , 3+5 } = min{2,7} = 2(B)

Distance to reach destination D from router C:

B=min{(C->B+B->D),(C->D+D->D),}=
min { 2+5 , 3+0 } = min(7,3) = 3(D)

Consequently, C's new routing table is:

Destination	distant	Hop

A	8	D
B	2	B
C	0	C
D	3	D

For router D:

The router D receives information from A and C.

The new routing table for D is calculated as:

Destination	**Vector A**	**Vector C**
A	0	8
B	8	2
C	8	0
D	5	3
The cost(D->A) = 5 The cost(D->C) = 3		

Distance to reach destination A from router D:

D=min{(D->A+A->A),(D->C+C->A)}=
min { 5+0 , 8+3 } = min{5,11} = 5(A)

Distance to reach destination B from router D:

D=min{(D->A+A->B),(D->C+C->B)}=
min { 5+8 , 3+2 } = min{13,5} = 5(D)

Distance to reach destination C from router D:

D=min{(D->A+A->B),(D->C+C->B)}=
min { 5+8 , 3+0 } = min{13,3} = 3(C)

Consequently, D's new routing table is:

Destination	**distant**	**Hop**
A	1	A
B	3	C
C	6	C
D	0	D

As you can see in the above network all the link has been used. In the routing table of A link AD and AB is used. In the routing table of B only link BA and BC. In the routing table of C, only llinksCB and CD are used and in D's routing table only llinksDA and DC are used.

In this sense, we can also check which of the links can be used and which cannot. In this way, we find the shortest path using the distance vector routing algorithm.

Link State Routing

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

Link State Routing has two phases:

Reliable Flooding

Initial state: Each node knows the cost of its neighbors.

Final state: Each node knows the entire graph.

The idea behind link state routing is fairly simple and can be stated as five parts. Each router must do the following things to make it work:

1. Discover its neighbors and learn their network addresses.
2. Set the distance or cost metric to each of its neighbors.
3. Construct a packet telling all it has just learned.
4. Send this packet to and receive packets from all other routers.
5. Compute the shortest path to every other router.

Route Calculation

Each node uses Dijkstra's algorithm on the graph to calculate the optimal routes to all nodes.

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

Let's describe some notations:

- **c(i , j):** Link cost from node i to node j. If i and j nodes are not directly linked, then c(i , j) = ∞.
- **D(v):** It defines the cost of the path from source code to destination v that has the least cost currently. o **P(v):** It defines the previous node (neighbor of v) along with current least cost path from source to v. o **N:** It is the total number of nodes available in the network.

Algorithm

Initialization

N = {A} // **A is a root node**.

for all nodes v if
v adjacent to A
then D(v) =
c(A,v) else D(v)
= infinity **loop**
find w not in N such that D(w) is a minimum.

Add w to N

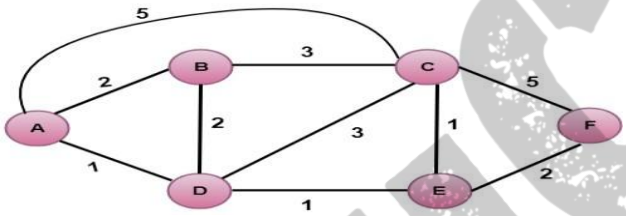
Update D(v) for all v adjacent to w and not in N:

D(v) = min(D(v) , D(w) + c(w,v))

Until all nodes in N

In the above algorithm, an initialization step is followed by the loop. The number of times the loop is executed is equal to the total number of nodes available in the network.

Let's understand through an example:



In the above figure, source vertex is A.

Step 1:

The first step is an initialization step. The currently known least cost path from A to its directly attached neighbors, B, C, D are 2,5,1 respectively. The cost from A to B is set to 2, from A to D is set to 1 and from A to C is set to 5. The cost from A to E and F are set to infinity as they are not directly linked to A.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞

Step 2:

In the above table, we observe that vertex D contains the least cost path in step 1. Therefore, it is added in N. Now, we need to determine a least-cost path through D vertex.

a) **Calculating shortest path from A to B**

1. v = B, w = D
2. D(B) = min(D(B) , D(D) + c(D,B))
3. = min(2, 1+2)>
4. = min(2, 3)
5. The minimum value is 2. Therefore, the currently shortest path from A to B is 2.

b) Calculating shortest path from A to C

1. v = C, w = D
2. D(B) = min(D(C) , D(D) + c(D,C))
3. = min(5, 1+3)
4. = min(5, 4)
5. The minimum value is 4. Therefore, the currently shortest path from A to C is 4.</p>

c) Calculating shortest path from A to E

1. v = E, w = D
2. D(B) = min(D(E) , D(D) + c(D,E))
3. = min(∞, 1+1)
4. = min(∞, 2)
5. The minimum value is 2. Therefore, the currently shortest path from A to E is 2.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞

Note: The vertex D has no direct link to vertex E. Therefore, the value of D(F) is infinity.

Step 3:

In the above table, we observe that both E and B have the least cost path in step 2. Let's consider the E vertex.

Now, we determine the least cost path of remaining vertices through E.

a) Calculating the shortest path from A to B.

1. v = B, w = E
2. D(B) = min(D(B) , D(E) + c(E,B))
3. = min(2 , 2+ ∞)
4. = min(2, ∞)
5. The minimum value is 2. Therefore, the currently shortest path from A to B is 2.

b) Calculating the shortest path from A to C.

1. v = C, w = E

2. D(B) = min(D(C) , D(E) + c(E,C))
3. = min(4 , 2+1)
4. = min(4,3)
5. The minimum value is 3. Therefore, the currently shortest path from A to C is 3.

c) Calculating the shortest path from A to F.

1. v = F, w = E
2. D(B) = min(D(F) , D(E) + c(E,F))
3. = min(∞ , 2+2)
4. = min(∞ ,4)
5. The minimum value is 4. Therefore, the currently shortest path from A to F is 4.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E

Step 4:

In the above table, we observe that B vertex has the least cost path in step 3. Therefore, it is added in N. Now, we determine the least cost path of remaining vertices through B.

a) Calculating the shortest path from A to C.

1. v = C, w = B
2. D(B) = min(D(C) , D(B) + c(B,C))
3. = min(3 , 2+3)
4. = min(3,5)
5. The minimum value is 3. Therefore, the currently shortest path from A to C is 3.

b) Calculating the shortest path from A to F.

1. v = F, w = B
2. D(B) = min(D(F) , D(B) + c(B,F))
3. = min(4, ∞)
4. = min(4, ∞)
5. The minimum value is 4. Therefore, the currently shortest path from A to F is 4.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)

1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E

Step 5:

In the above table, we observe that C vertex has the least cost path in step 4. Therefore, it is added in N. Now, we determine the least cost path of remaining vertices through C.

a) Calculating the shortest path from A to F.

1. v = F, w = C
2. D(B) = min(D(F) , D(C) + c(C,F))
3. = min(4, 3+5)
4. = min(4,8)
5. The minimum value is 4. Therefore, the currently shortest path from A to F is 4.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E
5	ADEBC					4,E

Final table:

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞

2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E
5	ADEBC					4,E
6	ADEBCF					

Disadvantage:

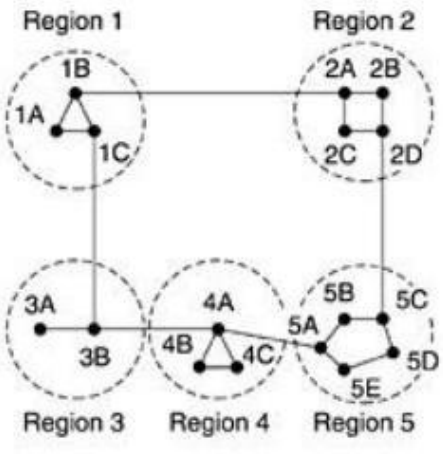
Heavy traffic is created in Line state routing due to Flooding. Flooding can cause an infinite looping, this problem can be solved by using Time-to-leave field

5.Hierarchical routing

Hierarchical routing is an algorithm for routing packets hierarchically. As networks grow in size, the router routing tables grow proportionally. Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them. At a certain point the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network.

When hierarchical routing is used, the routers are divided into regions, with each router knowing all the details about how to route packets to destinations within its own region, but knowing nothing about the internal structure of other regions. When different networks are interconnected, it is natural to regard each one as a separate region in order to free the routers in one network from having to know the topological structure of the other ones.

For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on. Fig. gives a quantitative example of routing in a two-level hierarchy with five regions. The full routing table for router 1A has 17 entries, as shown in Fig. (b). When routing is done hierarchically, as in Fig. (c), there are entries for all the local routers as before, but all other regions have been condensed into a single router, so all traffic for region 2 goes via the 1B -2A line, but the rest of the remote traffic goes via the 1C -3B line. Hierarchical routing has reduced the table from 17 to 7 entries. As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase. Fig.



(a)

Full table for 1A

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

(b)

Hierarchical table for 1A

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

(c)

Fig: Hierarchical routing

Unfortunately, these gains in space are not free. There is a penalty to be paid, and this penalty is in the form of increased path length. For example, the best route from 1A to 5C is via region 2, but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.

When a single network becomes very large, an interesting question is: How many levels should the hierarchy have? For example, consider a subnet with 720 routers. If there is no hierarchy, each router needs 720 routing table entries.

If the subnet is partitioned into 24 regions of 30 routers each, each router needs 30 local entries plus 23 remote entries for a total of 53 entries. If a three-level hierarchy is chosen, with eight clusters, each containing 9 regions of 10 routers, each router needs 10 entries for local routers, 8 entries for routing to other regions within its own cluster, and 7 entries for distant clusters, for a total of 25 entries. Kamoun and Kleinrock (1979) discovered that the optimal number of levels for an N router subnet is ln N, requiring a total of e ln N entries per router.

They have also shown that the increase in effective mean path length caused by hierarchical routing is sufficiently small that it is usually acceptable.

6.Broadcast and Multicast

The world of computer networks has a variety of communication mechanisms and protocols that provide a set of guidelines and rules to be followed while transmitting data from one node to another. These

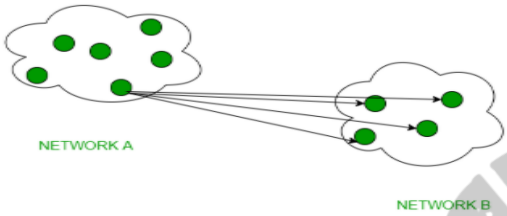
mechanisms and protocols determine the efficiency and reliability of the inter-connected and intra-connected network systems. In this article, we'll learn about the difference between two of the most commonly used message distribution mechanisms – Broadcast and Multicast, shedding light on the unique characteristics of both.

Broadcast and a Multicast are two different communication mechanism in computer networks for transmitting data between the nodes in a network.

Broadcast Routing

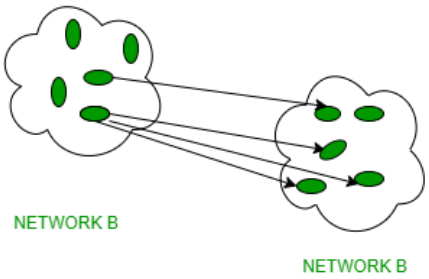
Broadcast transfer (one-to-all) techniques and can be classified into two types : Limited Broadcasting and direct Broadcasting. In broadcasting mode, transmission happens from one host to all the other hosts connected on the LAN. In simple words, broadcasting is a communication mechanism where data is sent to all the nodes in a network. The broadcast address is a special reserved address bits for broadcasting messages in a network, we can calculate the broadcast address given its IP address and the subnet Mask.

Devices such as bridge uses this. A protocol like ARP implements this, in order to know the MAC address for the corresponding IP address of the host machine. ARP does IP address to MAC address translation. RARP does the reverse.



Multicast

Multicasting has one/more senders and one/more recipients participate in data transfer traffic. In multicasting traffic recline between the boundaries of unicast and broadcast. It server's direct single copies of data streams and that are then simulated and routed to hosts that request it. IP multicast requires support of some other protocols such as IGMP (Internet Group Management Protocol), Multicast routing for its working. And also



in Classful IP addressing Class D is reserved for multicast groups.

Difference Between Broadcast and Multicast

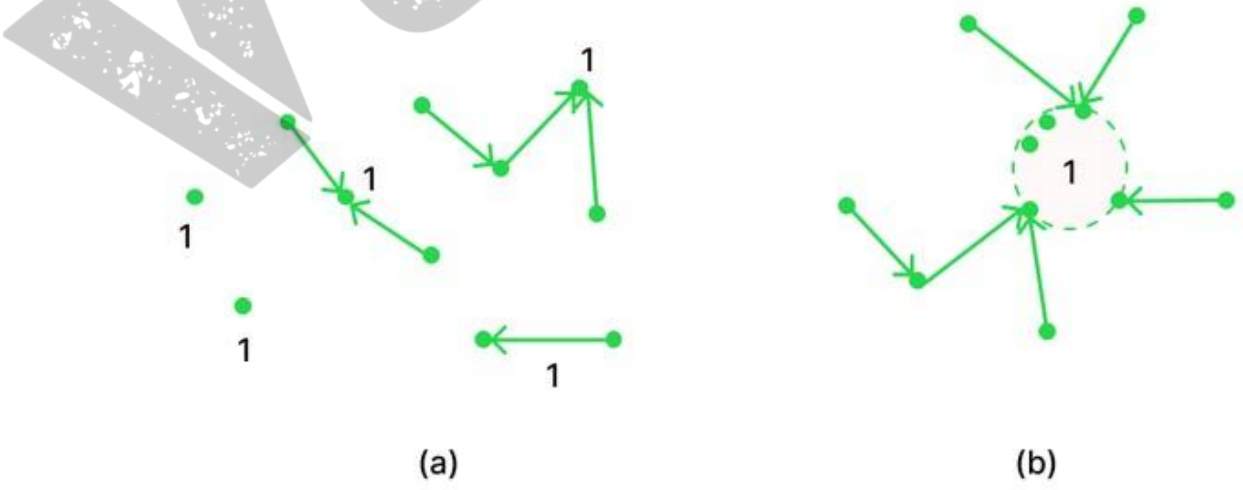
Broadcast	Multicast
It has one sender and multiple receivers.	It has one or more senders and multiple receivers.
It sent data from one device to all the	It sent data from one device to

Broadcast	Multicast
other devices in a network.	multiple devices.
It works on star and bus topology.	It works on star, mesh, tree and hybrid topology.
It scale well across large networks.	It does not scale well across large networks.
Its bandwidth is wasted.	It utilizes bandwidth efficiently.
It has one-to-all mapping.	It has one-to-many mapping.
Hub is an example of a broadcast device.	Switch is an example of a multicast device.
It increases network traffic because the data packets are sent to every other node in the network	It doesn't increase network traffic
The message to be sent should be tripled checked as some sensitive or confidential information shouldn't be distributed to everyone in the network	No such issue, because the message is target to only selected people.

8.Anycast Routing

A packet is delivered to the nearest member of a group, in anycast. **Anycast routing** finds these paths. Sometimes nodes provide a service, such as time of day or content distribution for which it is getting the right information all that matters, not the node that is contacted; any node will do. For example, anycast is used in the internet as part of DNS.

Topology : Regular distance vector and link state routing can produce anycast routes because there is no need to devise new routing schemes for anycast. Suppose we want to anycast to the members of group 1. They will be given the address "1", instead of different addresses. Distance vector routing will distribute vectors as usual, and nodes will choose the shortest path to destination 1. This will result in nodes sending to the nearest instance of destination 1. That is, it believes that all the instances of node 1 are the same node, as in the topology shown in figure below.



(a) Anycast routes to group 1 (b) Topology seen by the routing protocol

This procedure works for link state routing as well, although there is the added consideration that the routing protocol must not find seemingly short paths that pass through node 1. This would result in jumps through hyperspace, since the instances of node 1 are really nodes located in different parts of the network.

Routing for Mobile Hosts

Millions of people use computers while on go, from the truly mobile situations with a wireless device in moving cars, to nomadic situations in which laptop computers are used in a series of a different location. We use the term mobile hosts to mean either category, as distinct from stationary hosts that never move. The mobile hosts introduce a new complication to route packets to the mobile hosts, the network first has to find it.

9. Routing for mobile Hosts

Assumed model :

The model of the world that we will consider is one in which all hosts are assumed to have a permanent home location that never changes. Each host has a permanent home address that can be used to determine home location. Like the telephone number 1-212-5551212 indicates the United States (country code 1) and Manhattan (212).

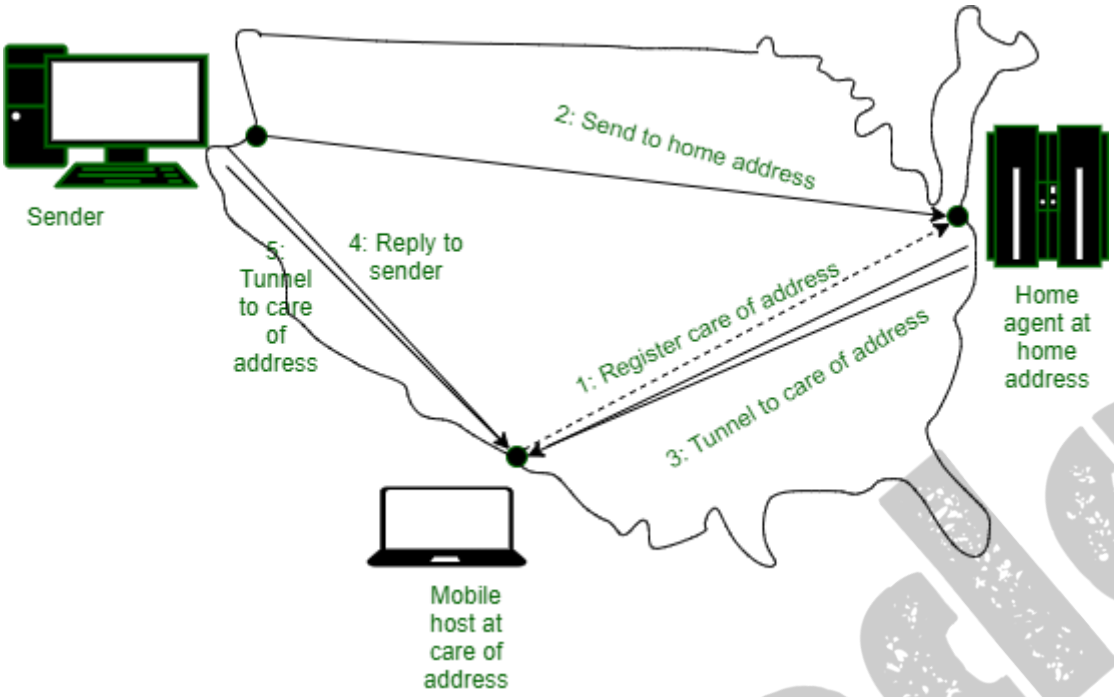
Feature :

- The basic idea used for mobile routing on the internet and cellular network is for the mobile hosts to tell the host at the home location.
- This host, which acts on behalf of the mobile host called a home agent.
- Once it knows where the mobile host currently located, it can forward packets so that they are delivered. The figure shows mobile routing in action.
- The local address called care-of address.
- Once it has this address it can tell its home agent where it is now. It does this by sending registration message to home agent with care of address.

Description of Diagram :

The message is shown with a dashed line in the figure indicate that it is a control message, not a data message. The sender sends a data packet to the mobile host using its permanent address. This packet is routed by the network to the host home location because the home addresses belong there. It encapsulates the packet with a new header and sends this bundle to the care-of address. This mechanism is called tunneling. It is very important on the internet, so we will look at it in more detail later.

Diagram :



Step1: When the encapsulated packet arrives at the care-of address, the mobile host unwraps it and retrieves the packet from the sender.
Step2: The overall route is called triangle routing because its way is circuitous if the remote location is far from the home location.
Step3: As part of the step, 4 senders learn the current care-of address.
Step4: Subsequent packets can be routed directly to the mobile host by tunnelling them to the care-of address (step 5) bypassing the home location.
Step5: If connectivity lost for any reason as the mobile moves, the home address can always be used to reach the mobile.

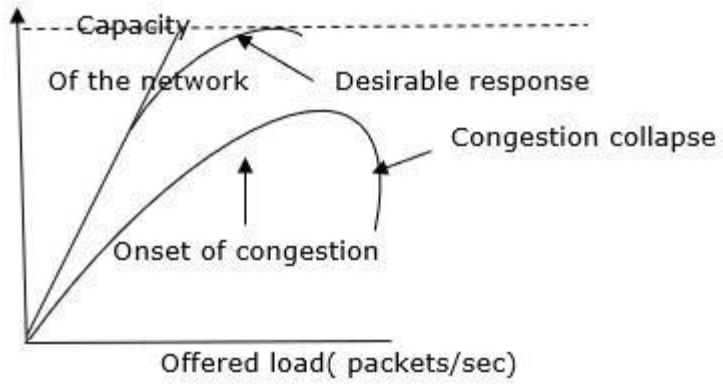
CONGESTION CONTROL ALGORITHM

When too many packets are present in the network it causes packet delay and loss of packet which degrades the performance of the system. This situation is called congestion.

The network layer and transport layer share the responsibility for handling congestions. One of the most effective ways to control congestion is trying to reduce the load that transport layer is placing on the network. To maintain this, network and transport layers have to work together.

When too many packets are present in the network it causes packet delay and loss of packet which degrades the performance of the system. This situation is called congestion.

The network layer and transport layer share the responsibility for handling congestions. One of the most effective ways to control congestion is trying to reduce the load that transport layer is placing on the network. To maintain this, network and transport layers have to work together.



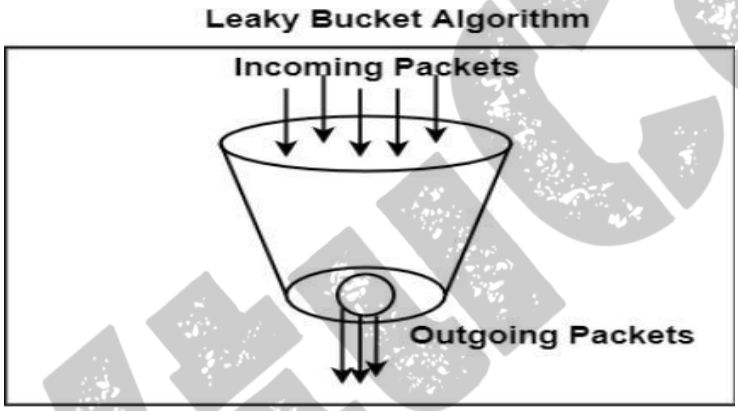
With too much traffic, performance drops sharply.

Types of Congestion Control Algorithms

There are two types of Congestion control algorithms, which are explained below −

Leaky Bucket Algorithm

It mainly controls the total amount and the rate of the traffic sent to the network.



Step 1 − Let us imagine a bucket with a small hole at the bottom where the rate at which water is poured into the bucket is not constant and can vary but it leaks from the bucket at a constant rate.

Step 2 − So (up to water is present in the bucket), the rate at which the water leaks does not depend on the rate at which the water is input to the bucket.

Step 3 − If the bucket is full, additional water that enters into the bucket that spills over the sides and is lost.

Step 4 − Thus the same concept applied to packets in the network. Consider that data is coming from the source at variable speeds. Suppose that a source sends data at 10 Mbps for 4 seconds. Then there is no data for 3 seconds. The source again transmits data at a rate of 8 Mbps for 2 seconds. Thus, in a time span of 8 seconds, 68 Mb data has been transmitted.

That's why a leaky bucket algorithm is used. The data flow would be 8 Mbps for 9 seconds. Thus, the constant flow is maintained.

Token bucket algorithm

Token bucket algorithms are used to overcome the problems that we are facing using leaky bucket algorithms. The leaky bucket algorithm's main problem is that it cannot control burst data, as it only allows average rate i.e. constant rate of data flow and also it does not consider idle time of host.

Step 1 − For example, if the host was idle for 12 seconds and now it is willing to send data at a very high speed for another 12 seconds, the total data transmission will be divided into 24 seconds and average data rate will be maintained.

Step 2 − The host has no advantage of sitting idle for 12 seconds. Thus, we adopted a token bucket algorithm.

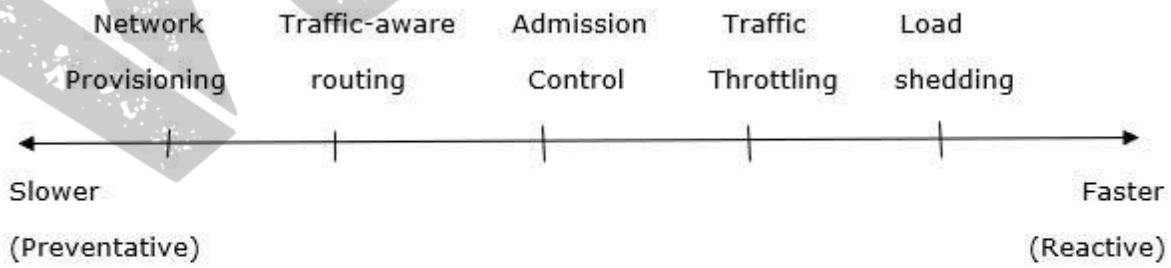
Step 3 − Hence, the token bucket algorithm is modification of leaky bucket in which leaky bucket contains tokens.

Step 4 − In this a token(s) are generated at every clock tick. For every packet to be transmitted, the system must remove token(s) from the bucket. Thus, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens.

For example, if a system generates 10 tokens in one clock tick and the host is idle for 10 ticks. The bucket will contain 10, 00 tokens. Now, suppose the host wants to send burst data, it consumes all 10, 00 tokens at once for sending 10, 00 cells or bytes. Thus, the host can be able to send burst data as long as the bucket is not empty.

1.Approaches to Congestion Control

There are some approaches for congestion control over a network which are usually applied on different time scales to either prevent congestion or react to it once it has occurred.



Time scale of approaches to congestion control

Let us understand these approaches step wise as mentioned below −

Step 1 − The basic way to avoid congestion is to build a network that is well matched to the traffic that it carries. If more traffic is directed but a low-bandwidth link is available, definitely congestion occurs.
Step 2 − Sometimes resources can be added dynamically like routers and links when there is serious congestion. This is called provisioning, and which happens on a timescale of months, driven by long-term trends.

Step 3 − To utilise most existing network capacity, routers can be tailored to traffic patterns making them active during daytime when network users are using more and sleep in different time zones.

Step 4 − Some of local radio stations have helicopters flying around their cities to report on road congestion to make it possible for their mobile listeners to route their packets (cars) around hotspots. This is called traffic aware routing.

Step 5 − Sometimes it is not possible to increase capacity. The only way to reduce the congestion is to decrease the load. In a virtual circuit network, new connections can be refused if they would cause the network to become congested. This is called admission control.

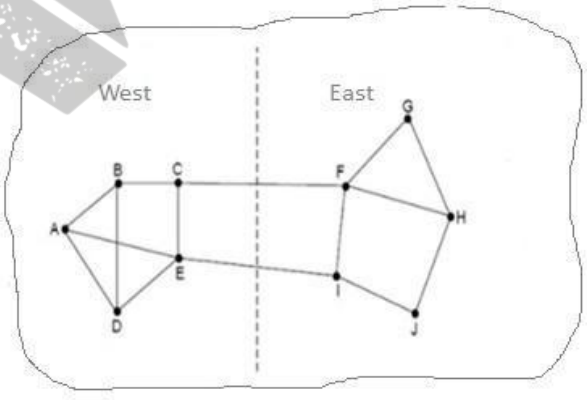
Step 6 − Routers can monitor the average load, queueing delay, or packet loss. In all these cases, the rising number indicates growing congestion. The network is forced to discard packets that it cannot deliver. The general name for this is Load shedding. The better technique for choosing which packets to discard can help to prevent congestion collapse.

2.Traffic Aware Routing

Traffic awareness is one of the approaches for congestion control over the network. The basic way to avoid congestion is to build a network that is well matched to the traffic that it carries. If more traffic is directed but a low-bandwidth link is available, congestion occurs.

The main goal of traffic aware routing is to identify the best routes by considering the load, set the link weight to be a function of fixed link bandwidth and propagation delay and the variable measured load or average queuing delay.

Least-weight paths will then favour paths that are more lightly loaded, remaining all are equal. The traffic aware routing is diagrammatically represented as follows −



Explanation

Step 1 − Consider a network which is divided into two parts, East and West both are connected by links CF and EI.

Step 2 − Suppose most of the traffic in between East and West is using link CF, and as a result CF link is heavily loaded with long delays. Including queueing delay in the weight which is used for shortest path calculation will make EI more attractive.

Step 3 − After installing the new routing tables, most of East-West traffic will now go over the EI link. As a result in the next update CF link will appear to be the shortest path.

Step 4 − As a result the routing tables may oscillate widely, leading to erratic routing and many potential problems.

Step 5 − If we consider only bandwidth and propagation delay by ignoring the load, this problem does not occur. Attempts to include load but change the weights within routing scheme to shift traffic across routes arrow range only to slow down routing oscillations.

Step 6 − Two techniques can contribute for successful solution, which are as follows −

- Multipath routing
- The routing scheme to shift traffic across routes.

Features

The features of traffic aware routing are as follows −

- It is one of the congestion control techniques.
- To utilise most existing network capacity, routers can be tailored to traffic patterns making them active during daytime when network users are using more and sleep in different time zones.
- Roots can be changed to shift traffic away because of heavily used paths.
- Network Traffic can be split across multiple paths.

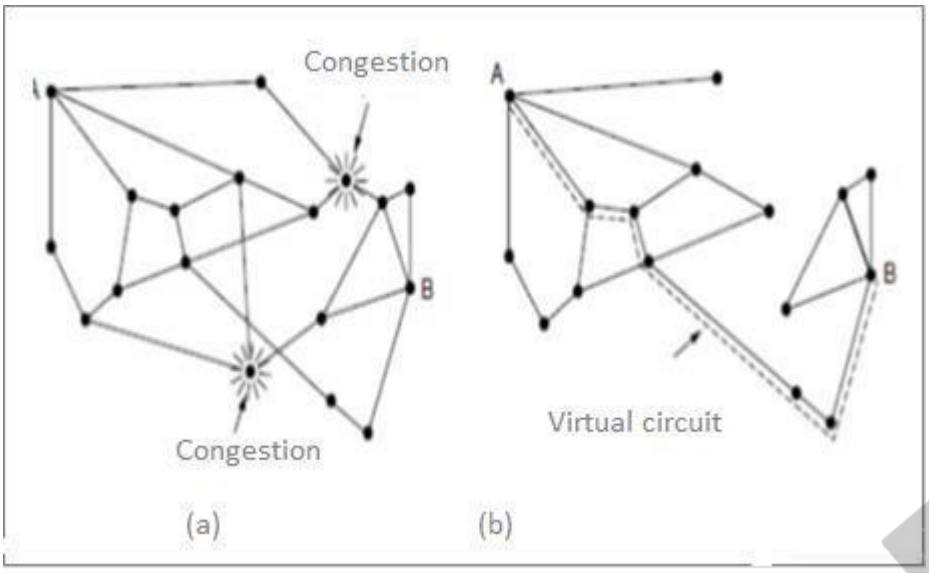
3.Admission Control

It is one of techniques that is widely used in virtual-circuit networks to keep congestion at bay. The idea is do not set up a new virtual circuit unless the network can carry the added traffic without becoming congested.

Admission control can also be combined with traffic aware routing by considering routes around traffic hotspots as part of the setup procedure.

Example

Take two networks (a) A congestion network and (b) The portion of the network that is not congested. A virtual circuit A to B is also shown below −



Explanation

Step 1 − Suppose a host attached to router A wants to set up a connection to a host attached to router B. Normally this connection passes through one of the congested routers.

Step 2 − To avoid this situation, we can redraw the network as shown in figure (b), removing the congested routers and all of their lines.

Step 3 − The dashed line indicates a possible route for the virtual circuit that avoids the congested routers.

4.Traffic Throttling

Traffic Throttling is an approach used to avoid congestion. In networks and the internet, the senders try to send as much traffic as possible as the network can readily deliver. In a network when congestion is approaching it should tell the senders of packets to slow down them. Traffic Throttling can be used in virtual circuit networks and datagram networks. Various approaches are used for throttling traffic. Each used approach must solve two problems. They are:

Problem 1. The router must be able to determine when the congestion is approaching. It must identify the congestion before it has arrived. For this, each router used in the network must continuously check for all the resources and their activities in the network. Router can continuously monitor using three possibilities. They are:

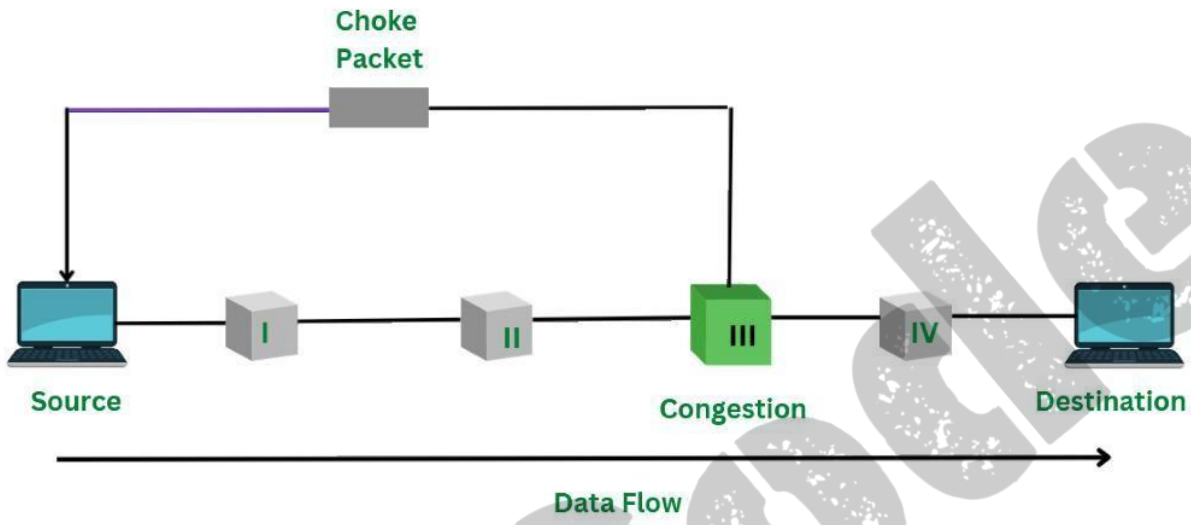
- Using output links
- To buffer the most useful and priority-based packets inside the router
- The total number of packets that are lost because of insufficient buffering

Problem 2. The second problem is that the router must send the feedback on time to the senders that are creating congestion. To deliver this feedback, the router must identify the senders properly. The router needs to send them warning efficiently without sending more packets in an already congested network. Different feedback mechanisms are used for solving this problem.

Feedback Mechanisms

1. **Choke packets**

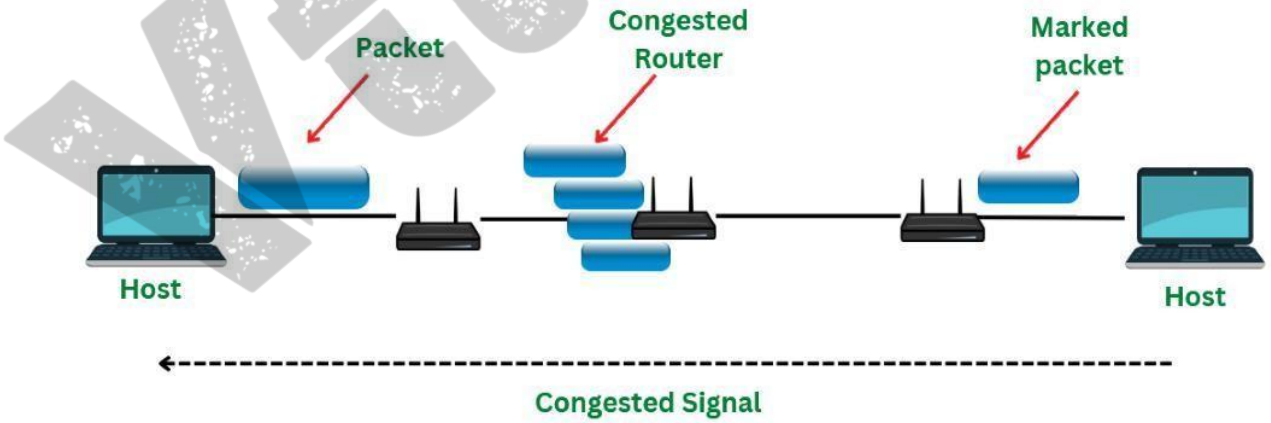
Choke packets are a mechanism where the router directly sends the choked packet back to its sender or host. The header bit of the original packet is turned on so that it will not be able to generate any choke packet. At the time of congestion to decrease the load router will send back only the choked packets at a lower rate. In the case of datagram networks randomly, the packets are selected therefore it leads to more choked packets. The below diagram describes the choke packets approach.



Choke Packets Mechanism

2. **Explicit Congestion Notification**

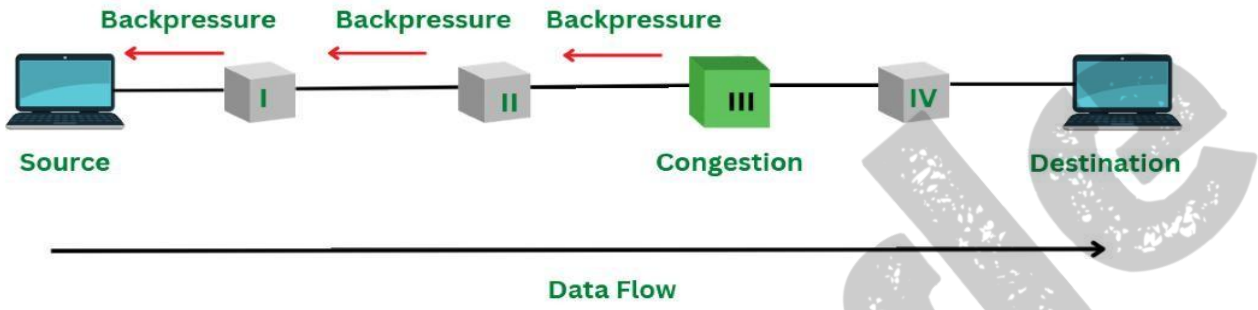
In the explicit congestion notification approach the router does not send extra packets to the host but sets a bit of any one of the packet headers to inform that the network has approached with congestion. When any packet is delivered in the network the destination sends a reply packet to the sender informing that congestion has occurred. In the case of choke packets, the sender then throttles its transmission. The below diagram describes the explicit congestion notification.



Explicit Congestion Notification

3. Hop-by-Hop Backpressure

After the congestion has been signaled still due to a slow signal many packets are received from the long distances. The choke packets have an effect at every step and each router requires more buffers. The main aim of this Hop-by-Hop Backpressure technique is to provide faster relief at the point of congestion in the network. This technique propagates in the opposite direction of the data and is majorly used in virtual circuits. The below diagram describes Hop-by-Hop Backpressure in detail.



Hop-by-Hop Backpressure

5.Load shedding

Load shedding is one of the techniques used for congestion control. A network router consists of a buffer. This buffer is used to store the packets and then route them to their destination. Load shedding is defined as an approach of discarding the packets when the buffer is full according to the strategy implemented in the data link layer. The selection of packets to discard is an important task. Many times packets with less importance and old packets are discarded.

Selection of Packets to be Discarded

In the process of load shedding the packets need to be discarded in order to avoid congestion. Therefore which packet needs to be discarded is a question. Below are the approaches used to discard the packets.

1. Random Selection of packets

When the router is filled with more packets, the packets are selected randomly for discarding. Discarding the packets it can include old, new, important, priority-based, or less important packets. Random selection of packets can lead to various disadvantages and problems.

2. Selection of packets based on applications

According to the application, the new packets will be discarded or old packets can be discarded by the router. When the application is regarding file transfer new packets are discarded and when the application is regarding multimedia the old packets are discarded.

3. Selection of packets based on priority

The source of packets can mark the priority stating how much important the packet is. Depending upon the priority provided by the sender the packet can either be selected or discarded. The priority can be given according to price, algorithm, and methods used, the functions that it will perform, and its effect on another task upon selecting and discarding the packets.

4. Random early detection

Randomly early detection is an approach in which packets are discarded before the buffer space becomes full. Therefore the situation of congestion is controlled earlier. In this approach, the router initially

maintains a specific queue length for the outgoing lines. When this average set line is exceeded it warns for congestion and discards the packets.

Advantages of Load Shedding

- Using the load shedding technique can help to recover from congestion.
- Load shedding technique reduces the flow of network traffic.
- It discards the packet from the network before congestion occurs
- Load shedding maintains a synchronized flow of packets in the network.

Disadvantages of Load Shedding

- If the size of the buffer is very less it discards more packets
- It is an overhead task for the router to continuously check if it has becomes full.
- Load shedding can sometimes discard important packets also considered as old packets.
- Load shedding cannot completely guarantee the avoidance of congestion.

QUALITY OF SERVICE

Quality of service (QoS) is the use of mechanisms or technologies that work on a network to control traffic and ensure the performance of critical applications with limited network capacity. It enables organizations to adjust their overall network traffic by prioritizing specific high-performance applications.

QoS is typically applied to networks that carry traffic for resource-intensive systems. Common services for which it is required include internet protocol television (IPTV), online gaming, streaming media, videoconferencing, video on demand (VOD), and Voice over IP (VoIP).

1. **Application Requirements**

A stream of packets from a source to a destination is called a flow (Clark,1988). A flow might be all the packets of a connection in a connection-oriented network, or all the packets sent from one process to another process in a connectionless network. The needs of each flow can be characterized by four primary parameters: bandwidth, delay, jitter, and loss. Together, these determine the QoS (Quality of Service) the flow requires.

Several common applications and the stringency of their network requirements are listed in Fig.1. Note that network requirements are less demanding than application requirements in those cases that the application can improve on the service provided by the network. In particular, networks do not need to be lossless for reliable file transfer, and they do not need to deliver packets with identical delays for audio and video playout. Some amount of loss can be repaired with retransmissions, and some amount of jitter can be smoothed by buffering packets at the receiver. However, there is nothing applications can do to remedy the situation if the network provides too little bandwidth or too much delay.

Understanding how QoS network software works is reliant on defining the various types of traffic that it measures. These are:

1. Bandwidth: The speed of a link. QoS can tell a router how to use bandwidth. For example, assigning a certain amount of bandwidth to different queues for different traffic types.
2. Delay: The time it takes for a packet to go from its source to its end destination. This can often be affected by queuing delay, which occurs during times of congestion and a packet waits in a

queue before being transmitted. QoS enables organizations to avoid this by creating a priority queue for certain types of traffic.

3. Loss: The amount of data lost as a result of packet loss, which typically occurs due to network congestion. QoS enables organizations to decide which packets to drop in this event.
4. Jitter: The irregular speed of packets on a network as a result of congestion, which can result in packets arriving late and out of sequence. This can cause distortion or gaps in audio and video being delivere

Application	Bandwidth	Delay	Jitter	Loss
E-mail	Low	Low	Low	Medium
File Sharing	High	Low	Low	Medium
Web Access	Medium	Medium	Low	Medium
Remote login	Low	Medium	Medium	Medium
Audio on demand	Low	Low	High	Low
Video on demand	High	Low	High	Low
Telephony	Low	High	High	Low
Video conferencing	High	High	High	Low

Fig.1.Stringency of applications' quality-of-service requirements.

2. Traffic Shaping

Traffic shaping enables organizations to increase network performance by controlling the amount of data that flows into and out of the network. Traffic is categorized, queued, and directed according to network policies. Essentially, traffic shaping regulates the network by slowing the transmission of packets classified as less important so that priority applications are delivered without delay.

Why is Traffic Shaping Important?

Any network has a finite amount of bandwidth, which makes traffic shaping through bandwidth management a key tool to ensure the performance of critical applications and the delivery of time-sensitive data. Traffic shaping is a powerful and flexible way to ensure quality of service and defend against bandwidth-abusing distributed denial-of- service (DDoS) attacks. It protects networks and applications from traffic spikes, regulates abusive users, and prevents network attacks from overwhelming network resources.

Traffic shaping reduces congestion and thus helps the network live up to its promise. However, to make it work, there is also the issue of how the provider can tell if the customer is following the agreement and what to do if the customer is not. Packets in excess of the agreed pattern might be dropped by the network, or they might be marked as having lower priority. Monitoring a traffic flow is called traffic policing.

Leaky and Token Buckets

Try to imagine a bucket with a small hole in the bottom, as illustrated in Fig.2(b). No matter the rate at which water enters the bucket, the outflow is at a constant rate, R, when there is any water in the bucket and zero when the bucket is empty. Also, once the bucket is full to capacity B, any additional water entering It spills over the sides and is lost.

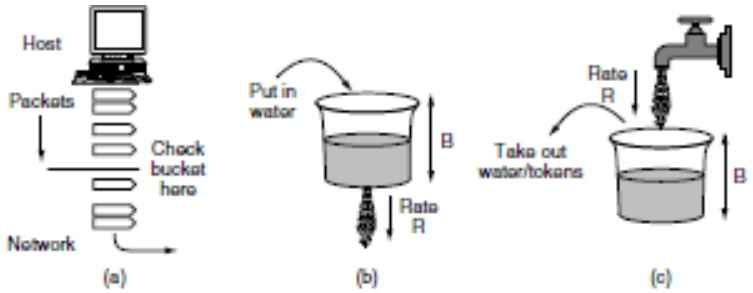


Fig.2(a) Shaping packets. (b) A leaky bucket. (c) A token bucket.

shown in Fig.2 (a). Conceptually, each host is connected to the network by an interface containing a leaky bucket. To send a packet into the network, it must be possible to put more water into the bucket. If a packet arrives when the bucket is full, the packet must either be queued until enough water leaks out to hold it or be discarded. The former might happen at a host shaping its traffic for the network as part of the operating system. The latter might happen in hardware at a provider network interface that is policing traffic entering the network. This technique was proposed by Turner (1986) and is called the leaky bucket algorithm. A different but equivalent formulatio1n is to imagine the network interface as a bucket that is being filled, as shown in Fig.2 (c). The tap is running at rate Rand the bucket has a capacity of B, as before. Now, to send a packet we must be able to take water, or tokens, as the contents are commonly called, out of the bucket (rather than bucket is empty, we must wait until more tokens arrive before we can send another packet. This algorithm is called the token bucket algorithm.

Leaky and token buckets limit the long-term rate of a flow but allow short term bursts up to a maximum regulated length to pass through unaltered and without suffering any artificial delays. Large bursts will be smoothed by a leaky bucket traffic shaper to reduce congestion in the network. As an example, imagine that a computer can produce data at up to 1000 Mbps (125 million bytes/sec) and that the first link of the network also runs at this speed. The pattern of traffic the host generates is shown in Fig. 1(a). This pattern is bursty. The average rate over one second is 200 Mbps, even though the host sends a burst of 16,000KB at the top speed of 1000 Mbps (for 1/8 of the second).

3 Packet Scheduling

Algorithms that allocate router resources among the packets of a flow and between competing flows are called packet scheduling algorithms. Three different kinds of resources can potentially be reserved for different flows:

1. Bandwidth.
2. Buffer space.
3. CPU cycles.

The first one, bandwidth, is the most obvious. If a flow requires 1 Mbps and the outgoing line has a capacity of 2 Mbps, trying to direct three flows through that line is not going to work. Thus, reserving bandwidth means not oversubscribing any output line.

A second resource that is often in short supply is buffer space. When a packet arrives, it is buffered inside the router until it can be transmitted on the chosen outgoing line. The purpose of the buffer is to absorb small bursts of traffic as the flows contend with each other. If no buffer is available, the packet has to be discarded since there is no place to put it. For good quality of service, some buffers might be reserved for a specific flow so that flow does not have to compete for buffers with other flows. Up to

some maximum value, there will always a buffer available when the flow needs one. Finally, CPU cycles may also be a scarce resource. It takes router CPU time to process a packet, so a router can process only a certain number of packets per second. While modern routers are able to process most packets quickly, some kinds of packets require greater CPU processing, such as the ICMP packets.

Packet scheduling algorithms allocate bandwidth and other router resources by determining which of the buffered packets to send on the output line next. We already described the most straightforward scheduler when explaining how routers work. Each router buffers packets in a queue for each output line until they be sent, and they are sent in the same order that they arrived. This algorithm is known as FIFO (First-In First-Out), or equivalently FCFS (First-Come First-Serve).FIFO routers usually drop newly arriving packets when the queue is full. Since the newly arrived packet would have been placed at the end of the queue, this behavior is called tail drop.The other scheduling algorithms that we will describe also create other opportunities for deciding which packet to drop when the buffers are full.

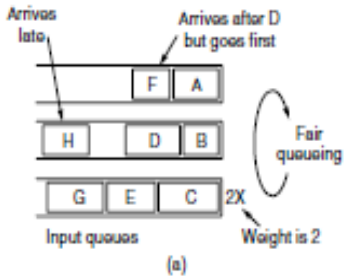
FIFO scheduling is simple to implement, but it is not suited to providing good quality of service because when there are multiple flows, one flow can easily affect the performance of the other flows. If the first flow is aggressive and sends large bursts of packets, they will lodge in the queue. Processing packets in the order of their arrival means that the aggressive sender can hog most of the capacity of the routers its packets traverse, starving the other flows and reducing their quality of service. To add insult to injury, the packets of the other flows that do get through are likely to be delayed because they had to sit in the queue behind many packets from the aggressive sender.

Many packet scheduling algorithms have been devised that provide stronger isolation between flows and thwart attempts at interference (Bhatti and Crowcroft, 2000). One of the first ones was the fair queueing algorithm devised by Nagle (1987). The essence of this algorithm is that routers have separate queues, one for each flow for a given output line. When the line becomes idle, the router scans the queues round-robin, as shown in Fig 3. It then takes the first packet on the next queue. In this way, with n hosts competing for the output line, each host gets to send one out of every n packets. It is fair in the sense that all flows get to send packets at the same rate. Sending more packets will not improve this rate.

Fig.3.Round-robin fair queueing.

Although a start, the algorithm has a flaw: it gives more bandwidth to hosts that use large packets than to hosts that use small packets. Demers et al. (1990) suggested an improvement in which the round-robin is done in such a way as to simulate a byte-by-byte round-robin, instead of a packet-by-packet round-robin. The trick is to compute a virtual time that is the number of the round at which each packet would finish being sent. Each round drains a byte from all of the queues that have data to send. The packets are then sorted in order of their finishing times and sent in that order.

This algorithm and an example of finish times for packets arriving in three flows are illustrated in Fig.4 If a packet has length L, the round at which it will finish is simply L rounds after the start time. The start time is either the finish time of the previous packet, or the arrival time of the packet, if the queue is empty when it arrives.



Packet	Arrival time	Length	Finish time	Output order
A	0	8	8	1
B	5	6	11	3
C	5	10	10	2
D	8	9	20	7
E	8	8	14	4
F	10	6	16	5
G	11	10	19	6
H	20	8	28	8

(b)

Fig.4. (a) Weighted Fair Queueing. (b) Finishing times for the packets

From the table in Fig. 4(b), and looking only at the first two packets in the top two queues, packets arrive in the order A, B, D, and F. Packet A arrives at round 0 and is 8 bytes long, so its finish time is round 8. Similarly the finish time for packet B is 11. Packet D arrives while B is being sent. Its finish time is 9 byte-rounds after it starts when B finishes, or 20. Similarly, the finish time for F is 16. In the absence of new arrivals, the relative sending order is A, B, F, D, even though F arrived after D. It is possible that another small packet will arrive on the top flow and obtain a finish time before D. It will only jump ahead of D if the transmission of that packet has not started. Fair queueing does not preempt packets that are currently being transmitted. Because packets are sent in their entirety, fair queueing is only an approximation of the ideal byte-by-byte scheme. But it is a very good approximation, always staying within one packet transmission of the ideal scheme.

One shortcoming of this algorithm in practice is that it gives all hosts the same priority. In many situations, it is desirable to give, for example, video servers more bandwidth than, say, file servers. This is easily possible by giving the video server two or more bytes per round. This modified algorithm is called WFQ (Weighted Fair Queueing). Letting the number of bytes per round be the weight of a flow, W, we can now give the formula for computing the finish time:

$$F_i = \max(A_i, F_i - 1) + L_i / W$$

where Ai is the arrival time, Fi is the finish time, and Li is the length of packet i. The bottom queue of Fig. 4(a) has a weight of 2, so its packets are sent more quickly as you can see in the finish times given in Fig. 4(b).

4 Integrated Services or IntServ:

The bandwidth over a specified network path is reserved by this QoS approach. Applications request a reservation of network resources for themselves, while network devices simultaneously watch the packet flow to ensure that network resources are open to receiving packets. Keep in mind that the IntServ-capable routers and resource reservation protocol are required while implementing the Integrated Services Model.

RSVP—The Resource reSerVation Protocol

RSVP is a transport layer protocol that is used to reserve resources in a computer network to get different quality of services (QoS) while accessing Internet applications. It operates over Internet protocol (IP) and initiates resource reservations from the receiver's end.

Features

- RSVP is a receiver oriented signalling protocol. The receiver initiates and maintains resource reservation.
- It is used both for unicasting (sending data from one source to one destination) and multicasting (sending data simultaneously to a group of destination computers).

- RSVP supports dynamic automatic adaptation to changes in network.
- It provides a number of reservation styles. It also provides support for addition of future styles.

RSVP Messages

There are two types of RSVP messages −

- Path Messages (path): A path message is sent by the sender to all receivers by multicasting storing the path state at each node in its path. It stores the necessary information so that the receivers can make the reservation.
- Reservation messages (resv): The resv message is sent by the receiver to the sender along the reverse path of the path message. It identifies the resources that is requires by the data flow.

As an example, consider the network of Fig. 5(a). Hosts 1 and 2 are multicast senders, and hosts 3, 4, and 5 are multicast receivers. In this example, the senders and receivers are disjoint, but in general, the two sets may overlap. The multicast trees for hosts 1 and 2 are shown in Fig. 5(b) and Fig. 5(c), respectively.

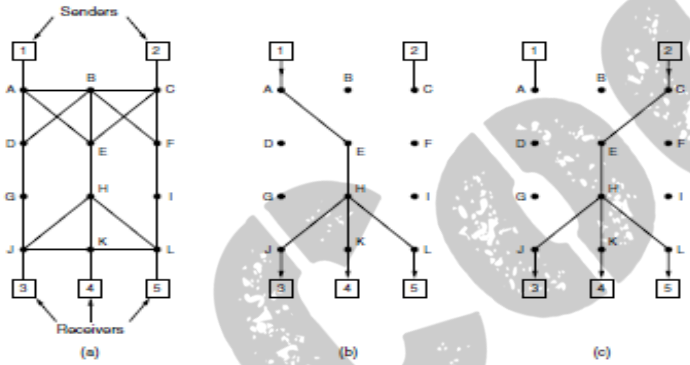


Fig.5.(a) A network. (b) The multicast spanning tree for host 1. (c) The multicast spanning tree for host 2.

To get better reception and eliminate congestion, any of the receivers in a group can send a reservation message up the tree to the sender. The message is propagated using the reverse path forwarding algorithm discussed earlier. At each hop, the router notes the reservation and reserves the necessary bandwidth. We saw in the previous section how a weighted fair queueing scheduler can be used to make this reservation. If insufficient bandwidth is available, it reports back failure. By the time the message gets back to the source, bandwidth has been reserved all the way from the sender to the receiver making the reservation request along the spanning tree.

An example of such a reservation is shown in Fig. 6(a). Here host 3 has requested a channel to host 1. Once it has been established, packets can flow from 1 to 3 without congestion. Now consider what happens if host 3 next reserves a channel to the other sender, host 2, so the user can watch two television programs at once. A second path is reserved, as illustrated in Fig. 6(b). Note that two separate channels are needed from host 3 to router E because two independent streams are being transmitted.

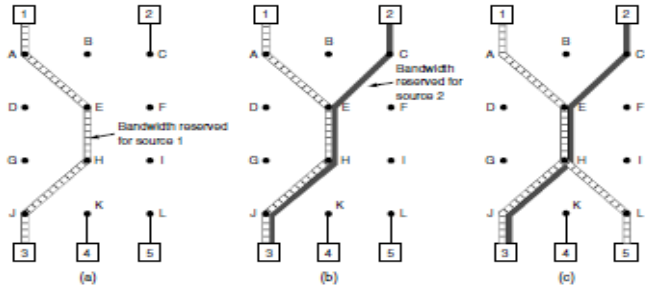


Fig.6(a) Host 3 requests a channel to host 1. (b) Host 3 then requests a second channel, to host 2. (c) Host 5 requests a channel to host 1.

Finally, in Fig. 6(c), host 5 decides to watch the program being transmitted by host 1 and also makes a reservation. First, dedicated bandwidth is reserved as far as router H. However, this router sees that it already has a feed from host 1, so if the necessary bandwidth has already been reserved, it does not have to reserve any more. Note that hosts 3 and 5 might have asked for different amounts of bandwidth (e.g., if host 3 is playing on a small screen and only wants the lowresolution information), so the capacity reserved must be large enough to satisfy the greediest receiver.

5 Differentiated Services

Differentiated Services: in this QoS model, the network elements such as routers and switches are configured to serve multiple categories of traffic with different priority orders. A company can categorize the network traffic based on its requirements. Eg. Assigning higher priority to audio traffic etc.

The difference between Integrated Services and Differentiated Services:

Integrated Services	**Differentiated Services**
This Architecture mainly specifies the elements to guarantee Quality of Service (QoS) on the network.	This Architecture mainly specifies a simple and scalable mechanism for classifying and managing the traffic of the network and also provides QoS on the modern IP networks.
These services mainly involve the prior reservation of the resources before sending in order to achieve Quality of Service.	These services mark the packets with the priority and then sends it to the network and there is no concept of prior reservation.
It is also known as IntServ.	It is also known as DiffSer.
These are not Scalable	These are Scalable.
These involve per flow Setup	These involve long term Setup
In this end to end service scope is available.	In this domain service scope is involved

Expedited Forwarding

The idea behind expedited forwarding is very simple. Two classes of service are available: regular and expedited. The vast majority of the traffic is expected to be regular, but a limited fraction of the packets are expedited. The expedited packets should be able to transit the network as though no other packets were present. In this way they will get low loss, low delay and low jitter service—just what is needed for VoIP. A symbolic representation of this ''two-tube'' system is given in Fig. 7. Note that there is still just one physical line. The two logical pipes shown in the figure represent a way to reserve bandwidth for different classes of service, not a second physical line.

One way to implement this strategy is as follows. Packets are classified as expedited or regular and marked accordingly. This step might be done on the sending host or in the ingress (first) router. The advantage of doing classification on the sending host is that more information is available about which packets belong to which flows. This task may be performed by networking software or even the operating system, to avoid having to change existing applications. For example, it is becoming common for VoIP packets to be marked for expedited service by hosts. If the packets pass through a corporate network or ISP that supports expedited service, they will receive preferential treatment. If the network does not support expedited service, no harm is done.

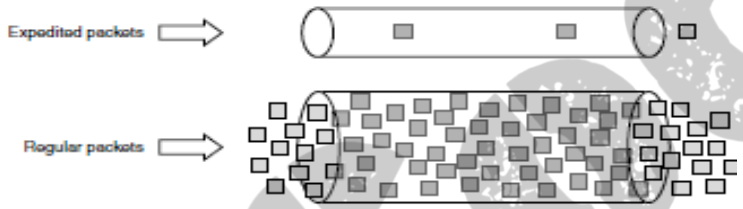


Fig.7. Expedited packets experience a traffic-free network.

Assured Forwarding

Figure.8. shows one way packets might be processed under assured forwarding.The first step is to classify the packets into one of the four priority classes. As before, this step might be done on the sending host (as shown in the figure) or in the ingress router, and the rate of higher-priority packets may be limited by the operator as part of the service offering.

The next step is to determine the discard class for each packet. This is done by passing the packets of each priority class through a traffic policer such as a token bucket. The policer lets all of the traffic through, but it identifies packets that fit within small bursts as low discard, packets that exceed small bursts as medium discard, and packets that exceed large bursts as high discard. The combination of priority and discard class is then encoded in each packet.

Finally, the packets are processed by routers in the network with a packetscheduler that distinguishes the different classes. A common choice is to use weighted fair queueing for the four priority classes, with higher classes given higher weights. In this way, the higher classes will get most of the bandwidth, but the lower classes will not be starved of bandwidth entirely.

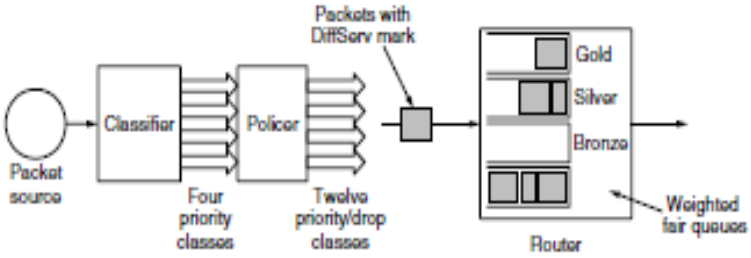


Fig.8.A possible implementation of assured forwarding.