(3)

UNIT - III

The Network layer

Network layer Design Issues

- store - and - forward packet switching :-

(1)

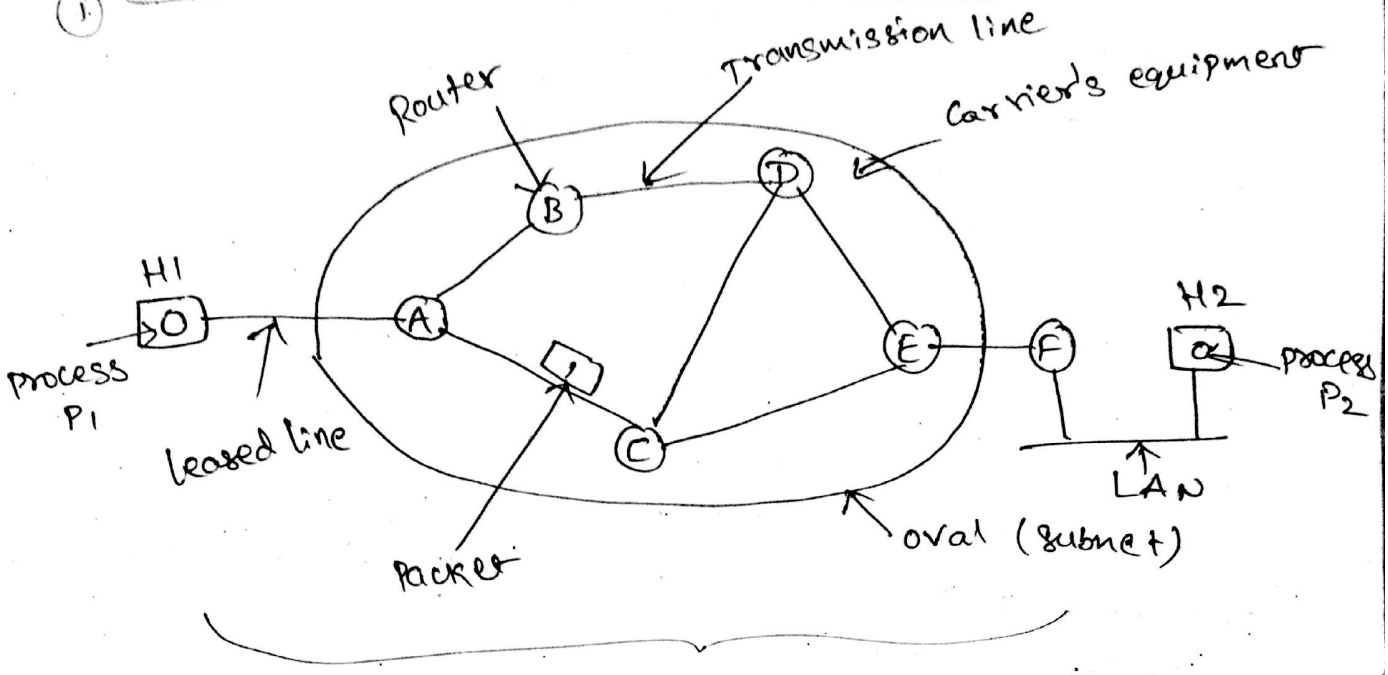


Fig:- The environment of the network layer protocols.

The major components of the system are the carrier's equipment (routers connected by transmission lines) shown inside the oval, and the customer's equipment shown outside the oval.

Host H1 is directly connected to one of the carrier's routers A, by a leased line.

In contrast, H2 is on a LAN with a router F, owned and operated by the customer. This router also has a leased line to the carrier's equipment. F router is outside the oval because it does not belong to the carrier, but in terms of constru-

ction, s/w and protocols it is probable no diff. exerv from the carrier's routers.

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

- services provided to the transport layer :-

(8) what kind of services the network layer provides to the transport layer.
The network layer services have been designed with the following goals :

1) The services should be independent of the router technology.
2) The TL should be shielded from the number, type, and topology of the routers
3) The n/w addresses made available to the TL should use a uniform numbering plan, even across LANs & WANs.

~~network~~ layer should provide connection-oriented service or connection-less service.

one camp (represented by the Internet community) argues that the routers job is moving packets around and nothing else. In their view, the subnet is inherently unreliable, no matter how it is designed. Therefore, The hosts should accept the fact that the network is unreliable

and do error control and flow control.

The other camp (represented by the telephone companies) argues that the subnet should provide a reliable, connection-oriented service. In this view, quality of service is the dominant factor, and without connections in the subnet, quality of service is very difficult to achieve.

These two camps are best exemplified by the Internet and ATM.

The Internet offers connectionless network-layer service.

ATM networks offer connection-oriented network-layer service.

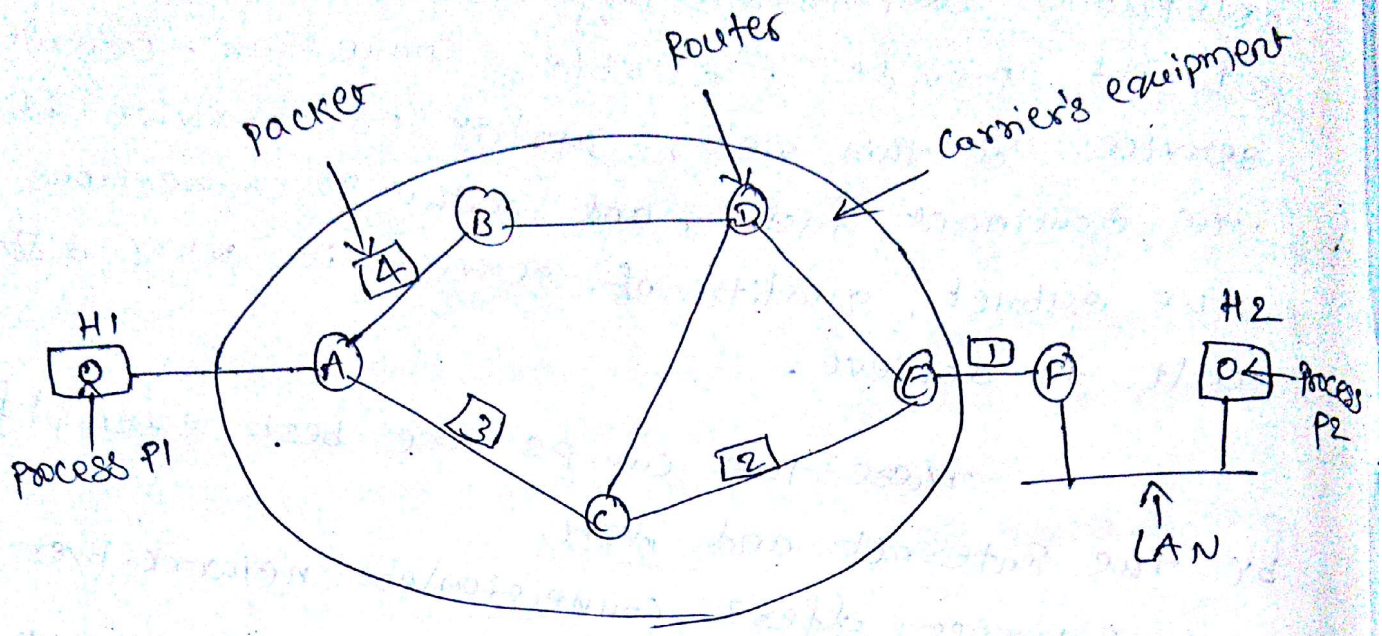
- Implementation of connectionless service :-

(3) Two different organizations are possible, depending on the type of service offered:

If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed. In this context, the packets are frequently called datagrams and the subnet is called a datagram subnet.

If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent. This connection is called a VC (virtual circuit), in analogy with the physical circuits set up by the

telephone system, and the subnet is called a virtual circuit subnet.



A's table

Initially	later
A -	A -
B B	B B
C C	C C
D B	D B
E C	E B
F C	F B

dest. line

C's table

A	A
B	A
C	-
D	D
E	E
F	E

E's table

A	C
B	D
C	C
D	D
E	-
F	F

Fig:- Routing within a datagram subnet

How a datagram subnet works :-

Suppose that the process P1 has a long message for P2. It hands the message to the transport layer with instructions to deliver it to process P2 on host H2.

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

At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination. only directly-connected lines can be used. for ex - in that fig - A has only two outgoing lines - to B and C - so every incoming packet must be sent to one of these routers, even if the destination is some other router. A's initial routing table is shown in that fig.

As they arrived at A, packets 1, 2 and 3 were stored briefly. Then each was forwarded to C according to A's table. Packet 1 was then forwarded to E and then to F. When it got to F, it was encapsulated in a DL frame and sent to H2 over the LAN. Packets 2 and 3 follow the same route.

However, something different happened to packet 4. When it got to A it was sent to router B, even though it is also destined for F. For some reason, A decided to send packet 4 via a different route than that of the first three. It learned of a traffic jam somewhere along the ACE path and updated its routing table as shown in A's table as in later.

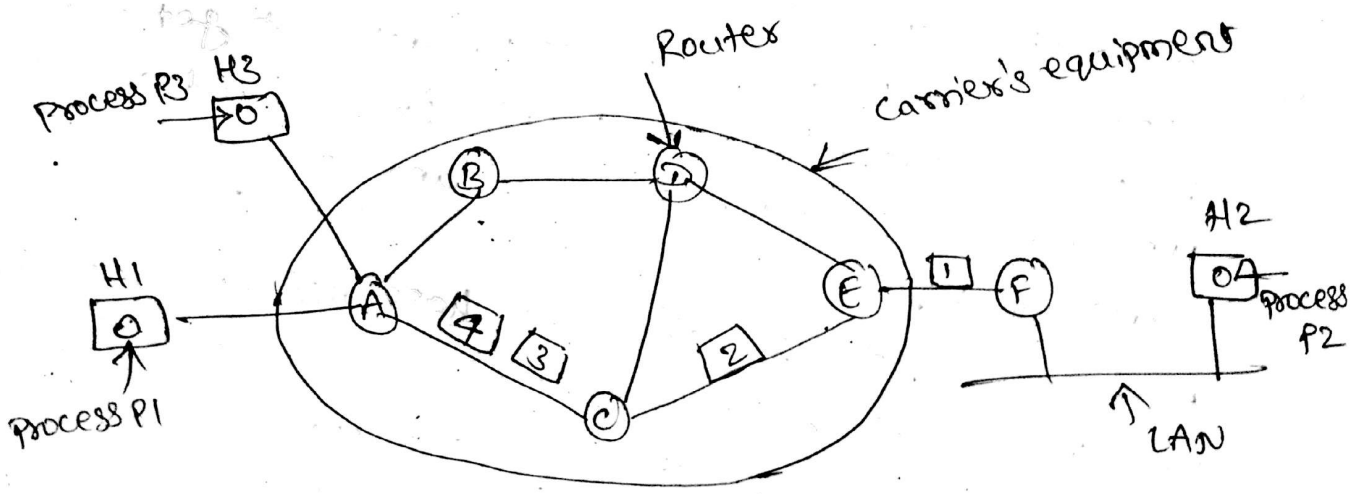
The algorithm manages the table and makes the routing decisions is called routing algorithm.

- Implementation of connection-oriented service:

(A) For connection-oriented service, we need a virtual circuit subnet.

How a virtual-circuit subnet works:-

The idea behind virtual circuits is to avoid having to choose a new route for every packet sent as shown in the previous fig. Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers. That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works. When the connection is released, the virtual circuit is also terminated. With CO service, each packet carries an identifier telling which VC it belongs to.



A's table

H1	1
H3	1

In

C	1
C	2

out

c's table

A	1
A	2

E	1
E	2

e's table

C	1
C	2

f	1
f	2

Fig:- Routing within a VC subnet

As an example, consider the situation of fig. Here, host H1 has established connection 1 with host H2. It is remembered as the first entry in each of the routing tables. The first line of A's table says that if a packet bearing connection identifier 1 comes in from H1, it is to be sent to router C and given connection identifier 1. Similarly the first entry at C routes the packet to E, also with connection identifier 1.

Now let us consider what happens if H3 also wants to establish a connection to H2. It chooses connection identifier 1 and tells the subnet to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here bcz although A can easily distinguish connection 1 packets from H1 from connection 1 packets from H3, C cannot do this. For this reason, A assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets. In some context, this is called label switching.

- Comparison of virtual-Circuit and Datagram subnets :-

Issue	Datagram subnet	VC subnet
Circuit set up	Not needed	Required
Addressing	Each packet contains the full source & destination address	Each packet contains a short VC number

Issue	Datagram subnet	VC subnet
State information	Routers do not hold state info about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is setup, 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
Congestion control	Difficult	Easy

Routing Algorithms

The main function of the NL is routing packets from the source machine to the destination mk. In most subnets, packets will require multiple hops to make the journey. The only notable exception is for broadcast networks, but even here routing is an issue if the source and destination are not on the same n/w. The algorithms that choose the routes and the data structures that they use are a major area of NL design.

The routing algorithm is that part of the NL s/w responsible for deciding which o/p line an incoming packet should be transmitted on. If the subnet uses datagrams internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time.
If the subnet uses virtual circuit internally, routing decisions are made only when a new VC is being set up. Thereafter, data packets just follow the previously established route. It is sometimes called session routing.

stability is an important goal for the routing algorithm.

Routing algorithms can be grouped into two major classes: nonadaptive or static
adaptive or dynamic

Nonadaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. It is sometimes called static routing.

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get their information (e.g. locally, from adjacent routers, or from all routers), when they change 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, no. of hops, or estimated transit time). It is sometimes called dynamic routing.

Nonadaptive or static

Adaptive or dynamic

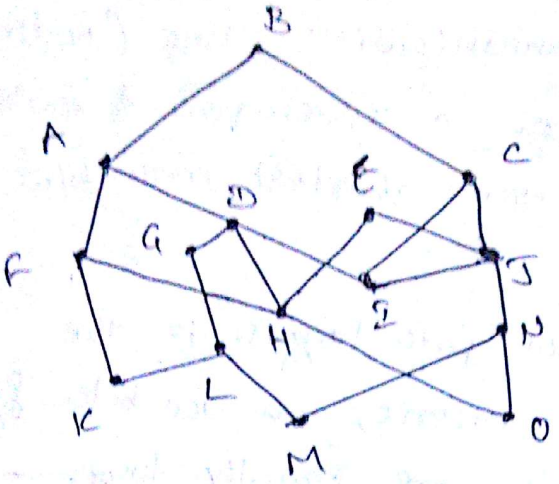
- The optimality principle :-

Before we get into specific algorithms, it may be helpful to note that one can make a general statement about optimal routes without regard to n/w topology or traffic. This statement is known as the optimality principle.

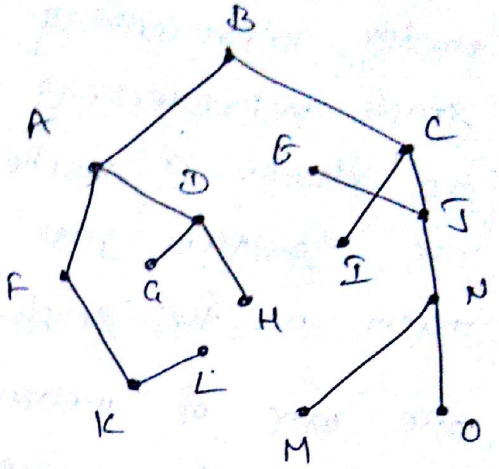
From the figure, if router J is on the optimal path from router I to router K then the optimal path from J to K also

falls along the same route.

To see this, call the part of the route from I to J r_1, and the rest of the route r_2. If a route better than r_2 existed from J to K, it could be concatenated with r_1 to improve the route from I to K, contradicting our statement that $r_1 r_2$ is optimal.



(a) A subnet



(b) A sink tree for router B

As a direct consequence of the optimality principle, we can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree as shown in fig(b), where the distance metric is the no. of hops. Note that a sink tree is not necessarily unique, other trees with the same path lengths may exist. The goal of all routing algorithms is to discover and use the sink trees for all routers. Sink tree does not contain any loops, so each packet will be delivered within a finite and bounded no. of hops.

the optimality principle and the sink tree provide a benchmark against which other routing algorithms can be measured.

Shortest path routing :- (static) Dijkstra

(1)

The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (or link). To choose a route between a given pair of routers, the algorithm just finds the shortest path b/w them on the graph.

One way of measuring the path length is the no. of hops. Using this metric, in the below fig. the paths ABC and ABE are equally long.

metrics

Another metric is the geographic distance in kilometers, in which case ABC is clearly much longer than ABE.

several algorithms for computing the shortest path b/w nodes of a graph are known. This one is due to Dijkstra (1959).

Each node is labeled (in parentheses) with its distance from the source node along the best known path.

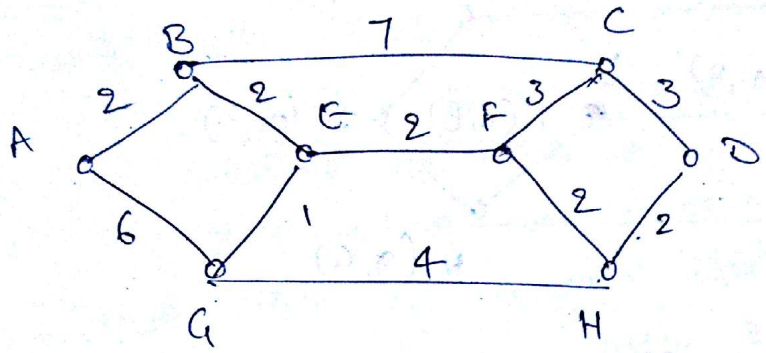
Initially, no paths are known, so all nodes are labeled with infinity.

As the algorithm proceeds and paths are found, the labels may change, reflecting better paths.

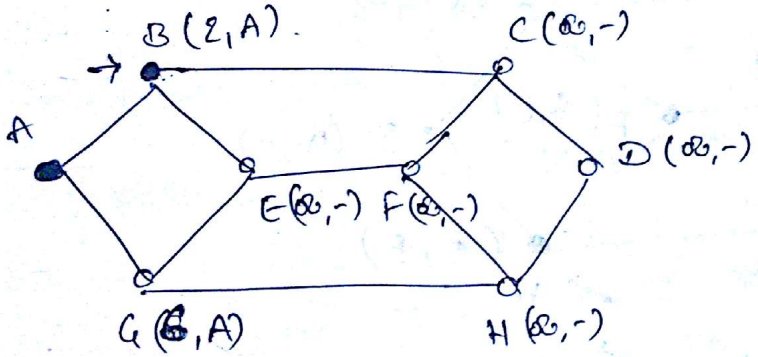
A label may be either tentative or permanent.

Initially, all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter

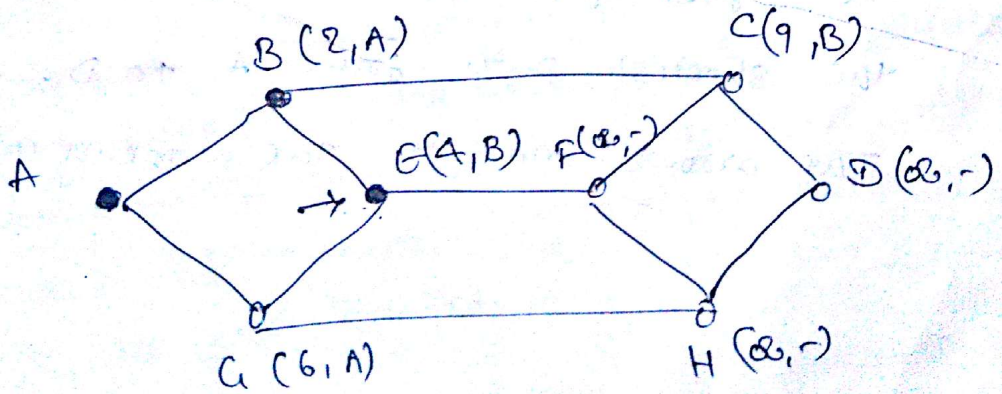
To illustrate how the labeling algorithm works, look at the weighted, undirected graph of fig (a).



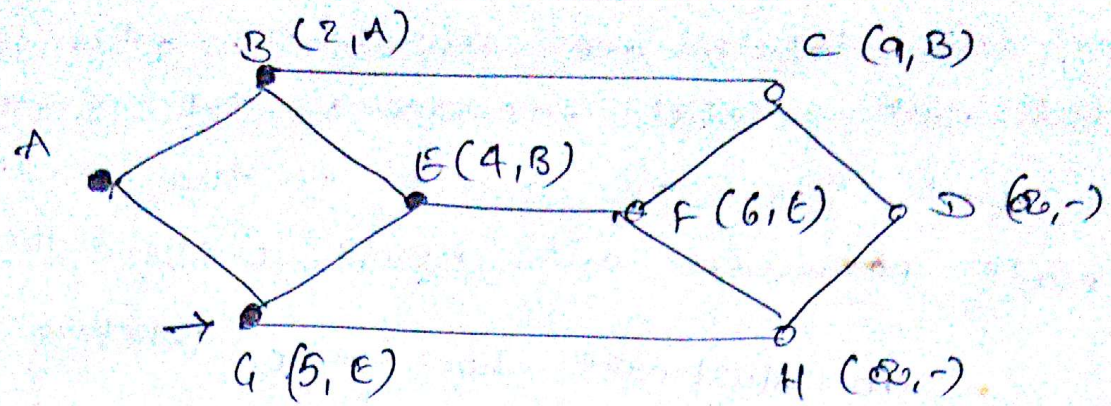
(a)



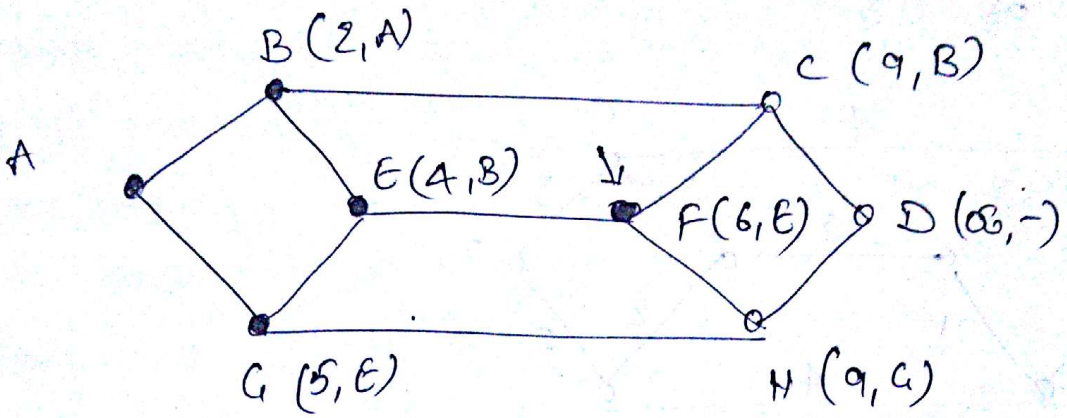
(b)



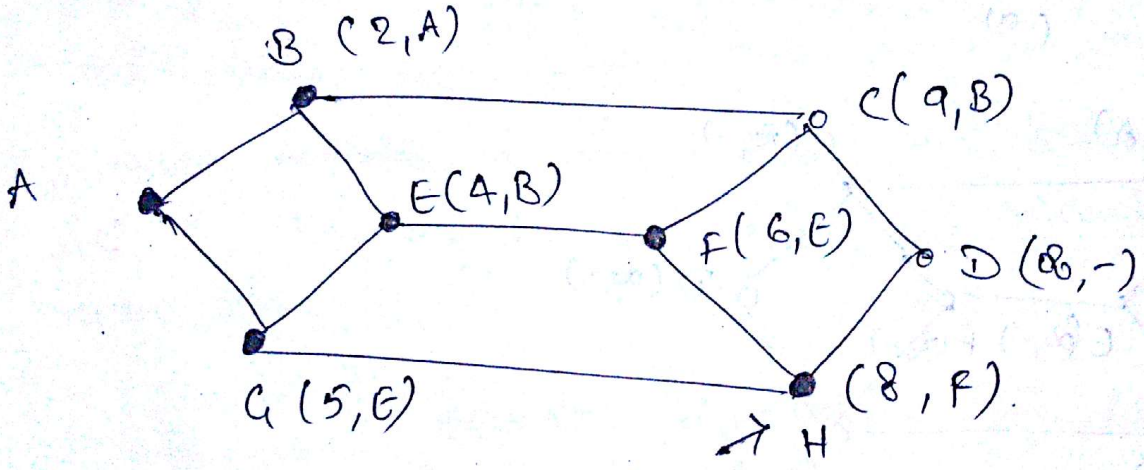
(c)



(d)



(e)



(f)

Fig :- The first five steps used in computing the shortest path from A to D.
The arrows indicate the working node.

— Flooding :- Static

(2)

Another static algorithm is flooding. in which every incoming packet is sent out on every outgoing line except the one it arrived on. Flooding obviously generates vast numbers of duplicate packets, some measures are taken to damp the process.

one such measure is to have a hop counter contained in the header of each packet, which is decremented at each hop. with the packet being discarded when the counter reaches zero. Ideally, the hop counter should be initialized to the length of the path from source to destination. If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the diameter of the subnet.

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

A variation of flooding that is slightly more practical is selective flooding. In this algorithm the routers do not send every incoming packet out on every line,

only on those lines that are going approximately in the right direction. There is usually little point in sending a westbound packet on an eastbound line unless the topology is extremely peculiar and the router is sure of this fact.

Flooding is not practical in most applications, but it does have some uses.

Applications :-

1) In military applications, where large nos of routers may be blown to bits at any instant, the tremendous robustness of flooding is highly desirable.

2) In distributed database applications, it is sometimes necessary to update all the databases concurrently, in which case flooding can be useful.

3) In wireless networks, all messages transmitted by a station can be received by all other stations within its radio range, which is, in fact, flooding can be useful.

A fourth possible use of flooding is as a metric against which other routing algorithms can be compared.

Flooding is always chooses the shortest path bcz it chooses every possible path in parallel.

(3)

- Distance vector Routing :-

Modern computer networks generally use dynamic routing algorithms rather than the static routing algorithm because static algorithms do not take the current network load into account. two dynamic algorithms - Distance vector routing
link state routing

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

The distance vector routing algorithm is sometimes called Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm. It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP. (Routing Information Protocol)

In distance vector routing, each router maintains a routing table indexed by, and containing one entry for, each router in the subnet. This entry contains two parts : the preferred outgoing line to use for that destination and an estimate of the time or distance to that destination.

The metric used might be number of hops, time delay in milliseconds, total number of packets queued along the path.

metrics

The router is assumed to know the "distance" to each of its neighbors.

If the metric is hops, the distance is just one hop.

If the metric is queue length, the router simply examines ~~each~~ each queue.

If the metric is delay, the router can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can.

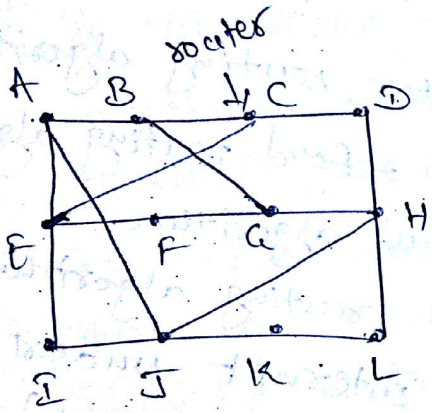


Fig (a) Subnet

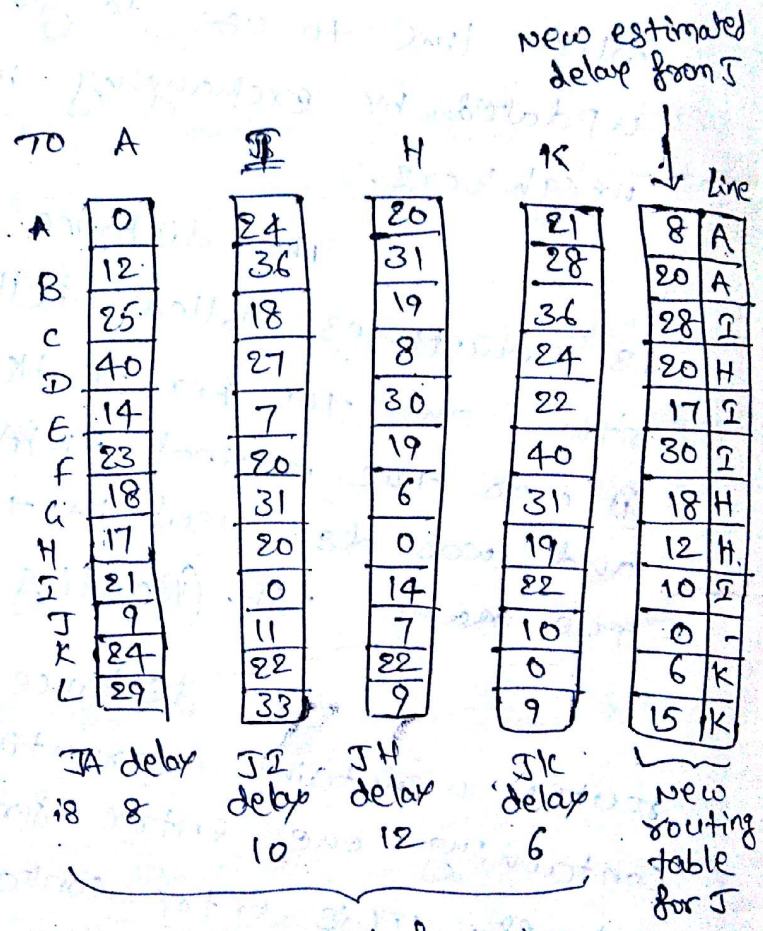


Fig (b) – Input from A, I, H, K and the new routing table for J.

fig (a) shows a subnet.
The first four columns of fig (b) show the delay vectors received from the neighbors of router J.

A claims to have a 12 msec delay to B,
a 25-msec delay to C,
a 40-msec delay to D etc.

I claims to have a 24-msec delay to A
36-msec delay to B
18-msec delay to C etc.

same as H and K claims to have a distances delay to every router.

suppose that J has measured or estimated its delay to its neighbors, A, I, H and K as 8, 10, 12 and 6 msec respectively.

consider how J computes its new route to router G. It knows that it can get to A in 8 msec, and A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec to G if it forwards packets bound for G to A. (18+8 = 26 msec)

J A delay is 8 msec
A claims 18 msec delay to G
so from J to G delay is 18+8 = 26 msec.

similarly, it computes the delay to G via I,

J I delay is 10 msec
I claims 31 msec delay to G
so from J to G via I - (31+10 = 41)

via H → (J H delay + H delay to G)
12 + 6 = 18 msec

via K → J K delay + K delay to G = 6 + 31 = 37 msec

The best of these values is 18, so it makes an entry in its routing table that the delay to G is 18 msec and the route to use is via H.

The same calculation is performed for all the other destinations, with the new routing table shown in the last column of the figure.

From J to A → JA delay is 8 via A (0+8) = 8

II delay + I to A = 10+24 = 34

JH delay + H to A = 12+20 = 32

JK delay + K to A = 6+21 = 27

From these values (8, 34, 32, 27) select minimum delay it happens via A so it indicates line.

From J to B → via A — JA delay + A to B = 8+12 = 20

via I — JI delay + I to B = 10+36 = 46

via H — JH delay + H to B = 12+31 = 43

via K — JK delay + K to B = 6+28 = 34

From these values (20, 46, 43, 34) select minimum delay i.e 20 it happens via A so, it indicates line. and so on, up to K node. so prepare new routing table from J to every node via neighbors nodes.

The Count-to-Infinity Problem:-

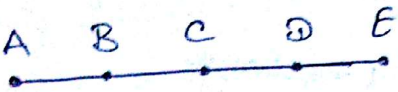
Distance vector routing works in theory but has a serious drawback in practice.

In particular, it reacts rapidly to good news, but leisurely to bad news.

Consider a router whose best route to destination X is large, If on the next

exchange neighbor A suddenly reports a short delay to X, the router just switches over to using the line to A to send traffic to X. In one vector exchange, the good news is processed.

To see how fast good news propagates, consider the five-node (linear) subnet as shown in below figure, where the delay metric is the number of hops.



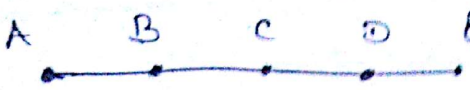
.	.	.	.	Initially
1	.	.	.	After 1 exchange
1	2	.	.	After 2 exchange
1	2	3	.	After 3 exchange
1	2	3	4	After 4 exchange

Suppose A is down initially and all the other routers know this. In other words, they have all recorded the delay to A as infinity.

When A comes up, the other routers learn about it via the vector exchanges. At the time of the first exchange, B learns that its left neighbor has zero delay to A. B now makes an entry in its routing table that A is one hop away to the left. All the other routers still think that A is down. At this point, the routing table entries for A are as shown in the second row as shown in the fig.

On the next exchange, C learns that B has a path of length 1 to A, so it updates

its routing table to indicate a path of length 2, but D and E do not hear the good news until later. Clearly, the good news is spreading at the rate of one hop per exchange. For a subnet whose longest path is of length N hops, within N exchanges everyone will know about newly revived lines and routers.



1	2	3	4	Initially
3	2	3	4	After 1 exchange
3	4	3	4	After 2 exchange
5	4	5	4	After 3 exchange
5	6	5	6	After 4 exchange
7	6	7	6	After 5 exchange
7	8	7	8	
.	.	.	.	

Now let us consider the situation of above figure, in which all the lines and routers are initially up. Routers B, C, D and E have distances to A of 1, 2, 3 and 4 respectively.

Suddenly A goes down, or alternatively, the line b/w A and B is cut, which is effectively the same thing from B's point of view.

At the first packet exchange, B does not hear anything from A, fortunately, C says: Do not worry; I have a path to A of length 2. B thinks it can reach A via C, with a path

length of 3. D and E do not update their entries for A on the first exchange. It shows in that figure at after 1 exchange.

on the second exchange, C notices that each of its neighbors claims to have a path to A of length 3. It picks one of them at random and makes its new distance to A 4, as shown in that figure at third row. Subsequent exchanges produce the history shown in the rest of the figure.

From this figure, it should be clear how bad news travels slowly: no router ever has a value more than one higher than the minimum of all its neighbors. Gradually, all routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinite. For this reason, it is wise to set infinity to the longest path plus 1. If the metric is time delay, there is no well-defined upper bound, so a high value is needed to prevent a path with a long delay from being treated down. This problem is known as the count-to-infinity problem.