MODULE III

Network layer – Routing – Shortest path routing, Flooding, Distance Vector Routing, Link State Routing, RIP, OSPF, Routing for mobile hosts.

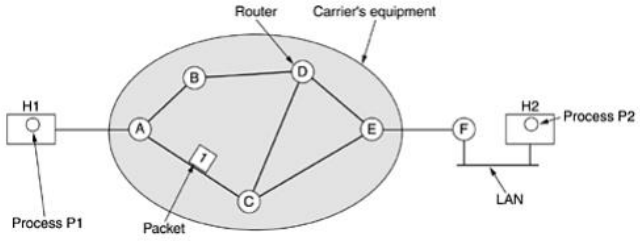
THE NETWORK LAYER

The network layer is concerned with getting packets from the source all the way to the destination. Getting to the destination may require making many hops at intermediate routers along the way. This function clearly contrasts with that of the data link layer, which has the more modest goal of just moving frames from one end of a wire to the other. Thus, the network layer is the lowest layer that deals with end-to-end transmission. To achieve its goals, the network layer must know about the topology of the communication subnet (i.e., the set of all routers) and choose appropriate paths through it. It must also take care to choose routes to avoid overloading some of the communication lines and routers while leaving others idle. Finally, when the source and destination are in different networks, new problems occur. It is up to the network layer to deal with them.

Network Layer Design Issues

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

Figure 5-1. 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 shaded oval, and
- the customers' equipment, shown outside the oval.
- Host H1 is directly connected to one of the carrier's routers, A, by a leased line.
- 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.

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 nextrouter along the path until it reaches the destination host, where it is delivered.

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

The network layer provides services to the transport layer at the network layer/transport layer interface. The network layer services have been designed with the following goals in mind:

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

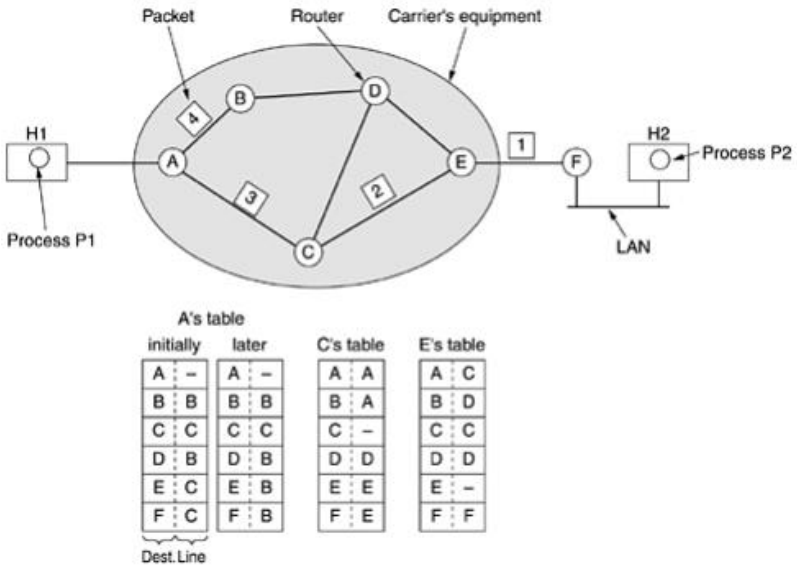
The two classes of service the network layer can provide to its users:

- The network service should be connectionless, with primitives SEND PACKET and RECEIVE PACKET. No packet ordering and flow control should be done, because the hosts are going to do that. Each packet must carry the full destination address, because each packet sent is carried independently of its predecessors.
- The subnet should provide a reliable, connection-oriented service. Quality of service is the dominant factor, and without connections in the subnet, quality of service is very difficult to achieve, especially for real-time traffic such as voice and video.

3. Implementation of Connectionless Service.

If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed. The packets are frequently called **datagrams** (in analogy with telegrams) and the subnet is called a **datagram subnet**. If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent. This connection is called a **VC (virtual circuit).**

Figure 5-2. Routing within a datagram subnet.



A's table initially		A's table later		C's table		E's table	
A	–	A	–	A	A	A	C
B	B	B	B	B	A	B	D
C	C	C	C	C	–	C	C
D	B	D	B	D	D	D	D
E	C	E	B	E	E	E	–
F	C	F	B	F	E	F	F

Suppose that the process P1 in Fig. 5-2 has a long message for P2. It hands the message to the transport layer with instructions to deliver it to process P2 on host H2. The transport layer code runs on H1, typically within the operating system. It prepends a transport header to the front of the message and hands the result to the network layer.

the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4 and sends each of them in turn to router A using some point-to-point protocol, for example, PPP. At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination. Only directly-connected lines can be used. For example, in Fig. 5-2, A has only two outgoing lines—to B and C—so every incoming

packet must be sent to one of these routers, even if the ultimate destination is some other router. A's initial routing table is shown in the figure under the label "initially."

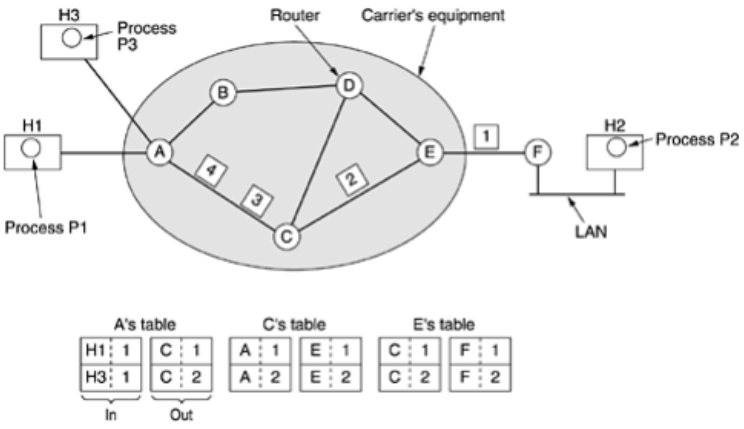
As they arrived at A, packets 1, 2, and 3 were stored briefly (to verify their checksums). Then each was forwarded to C according to A's table. Packet 1 was then forwarded to E and then to F. When it got to F, it was encapsulated in a data link layer frame and sent to H2 over the LAN. Packets 2 and 3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

Packet 4, When it got to A it was sent to router B, even though it is also destined for F. For some reason, A decided to send packet 4 via a different route than that of the first three. Perhaps it learned of a traffic jam somewhere along the ACE path and updated its routing table, as shown under the label "later." The algorithm that manages the tables and makes the routing decisions is called **the routing algorithm**.

4. **Implementation of Connection-Oriented Service.**

For connection-oriented service, we need a virtual-circuit subnet.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. When the connection is released, the virtual circuit is also terminated.

Figure 5-3. Routing within a virtual-circuit subnet.



With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to. As an example, consider the situation of Fig. 5-3. 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.

If H3 also wants to establish a connection to H2. It chooses connection identifier 1 (because it is initiating the connection and this is its only connection) and tells the subnet to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here because although A can easily distinguish connection 1 packets from H1 from connection 1 packets from H3, C cannot do this. For this reason, A assigns a different connection identifier to the outgoing traffic for the second

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

5. **Comparison of Virtual-Circuit and Datagram Subnets:**

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

ROUTING:

The main function of the network layer is routing packets from the source machine to the destination machine. In most subnets, packets will require multiple hops to make the journey.

The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on.

- If the subnet uses **datagrams** internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time.
- If the subnet uses **virtual circuits** internally, routing decisions are made only when a new virtual circuit is being set up. Thereafter, data packets just follow the previously-established route. The latter case is sometimes called **session routing** because a route remains in force for an entire user session.

Routing - is making the decision which routes to use.
Forwarding - is what happens when a packet arrives.
Router - having two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing tables. This process is **forwarding**. The other process is responsible for **filling in and updating the routing tables**. That is where the routing algorithm comes into play.
Properties are desirable in a routing algorithm: correctness, simplicity, robustness, stability, fairness, and optimality.
Routing algorithms can be grouped into two major classes: **nonadaptive and adaptive.**

- **Nonadaptive algorithms** do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J (for all I and J) is computed in advance, off-line, and downloaded to the routers when the network is booted. This procedure is sometimes called **static routing.**

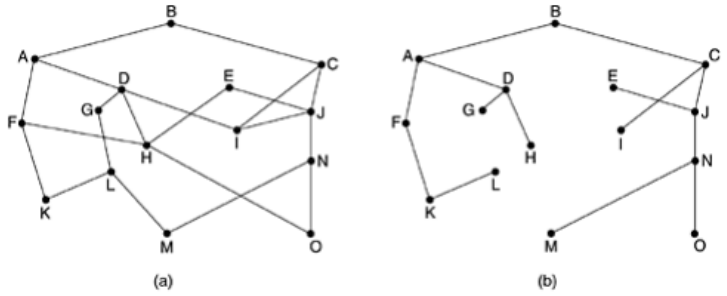
- **Adaptive algorithms,** change their routing decisions to reflect changes in the topology, and the traffic. Adaptive algorithms differ in where they get their information, when they change the routes, and what metric is used for optimization.

The Optimality Principle

A general statement about optimal routes without regard to network topology or traffic is known as the **optimality principle**. It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.

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** and is illustrated in Fig. 5-6, where the distance metric is the number 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.

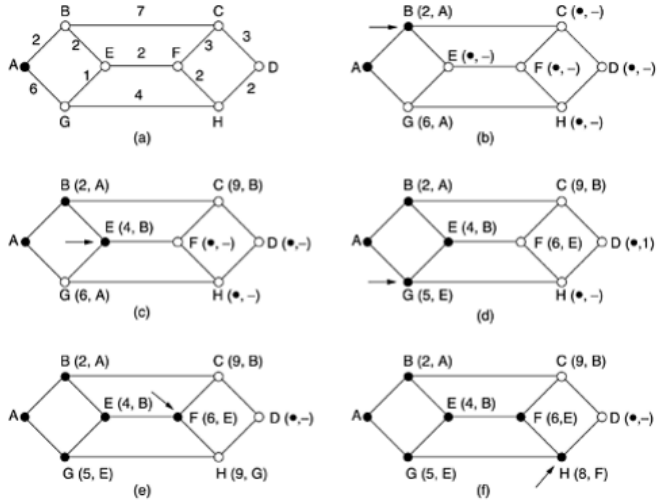
Figure 5-6. (a) A subnet. (b) A sink tree for router B



Shortest Path Routing

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

Figure 5-7. The first five steps used in computing the shortest path from A to D. The arrows indicate the working node.



- Measuring path length is the number of hops → the paths ABC and ABE in Fig. 5-7 are equally long.
- Another metric is the geographic distance → in kilometers, in which case ABC is clearly much longer than ABE.

Dijkstra's shortest path algorithm:

Compute the shortest path starting at the terminal node, t, rather than at the source node, s. Since the shortest path from t to s in an undirected graph is the same as the shortest path from s to t. The reason for searching backward is that each node is labeled with its predecessor rather than its successor. When the final path is copied into the output variable, path, the path is thus reversed.

FLOODING:

Flooding is a static algorithm, in which every incoming packet is sent out on every outgoing line except the one it arrived on. Flooding generates vast numbers of duplicate packets. One measure avoid this 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. 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 the full diameter of the subnet.
- keep track of which **packets have been flooded**, to avoid sending them out a second time. 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. To **prevent the list from growing** without bound, each list should be augmented by **a counter, k**, meaning that all sequence numbers through k have been seen. When a packet comes in, it is easy to check if the packet is a duplicate; if so, it is discarded. Furthermore, the full list below k is not needed, since k effectively summarizes it.
- **selective flooding -** the routers do not send every incoming packet out on every line, only on those lines that are going approximately in the right direction.

Uses of flooding:

- in military applications, where large numbers of routers may be blown to bits at any instant, the tremendous robustness of flooding is highly desirable.
- In distributed database applications, it is sometimes necessary to update all the databases concurrently, in which case flooding can be useful.
- 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, and some algorithms utilize this property.
- It is as a metric against which other routing algorithms can be compared. Flooding always chooses the shortest path because it chooses every possible path in parallel.

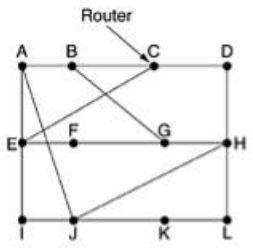
Dynamic algorithms:
Take the current network load into account. Two dynamic algorithms in particular, distance vector routing and link state routing, are the most popular.

DISTANCE VECTOR ROUTING / BELLMAN-FORD ROUTING ALGORITHM / FORD-FULKERSON ALGORITHM:

Distance vector routing algorithms operate by having each router maintain a table (i.e, a vector) 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.
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 etc.

Imagine that one of these tables has just come in from neighbor X, with Xi being X's estimate of how long it takes to get to router i. If the router knows that the delay to X is m msec, it also knows that it can reach router i via X in Xi + m msec. By performing this calculation for each neighbor, a router can find out which estimate seems the best and use that estimate and the corresponding line in its new routing table. Note that the old routing table is not used in the calculation.
This updating process is illustrated in Fig. 5-9. Part (a) shows a subnet. The first four columns of part (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. 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.

Figure 5-9. (a) A subnet. (b) Input from A, I, H, K, and the new routing table for J.



To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	—
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

(a) (b)

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. Similarly, it computes the delay to G via I, H, and K as 41 (31 + 10), 18 (6 + 12), and 37 (31 + 6) msec, respectively. 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 that 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.

Drawback : The Count-to-Infinity Problem:
It converges to the correct answer, it may do so slowly. Consider the five-node (linear) subnet of Fig. 5-10, where the delay metric is the number of hops. 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.

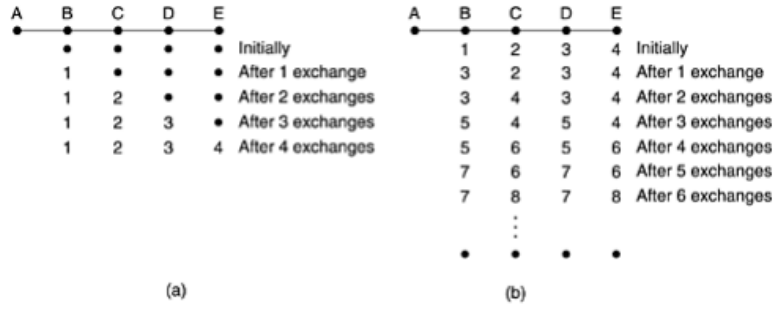


fig 5-10

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 of Fig. 5-10(a). 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. In a subnet whose longest path is of length N hops, within N exchanges everyone will know about newly-revived lines and routers.

Fig. 5-10(b), 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 between 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. Little does B know that C's path runs through B itself. For all B knows, C might have ten lines all with separate paths to A of length 2. As a result, 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.

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 the them at random and makes its new distance to A 4, as shown in the third row of Fig. 5-10(b). Subsequent exchanges produce the history shown in the rest of Fig. 5-10(b).

From this figure, it should be clear why 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 infinity. 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 as down. This problem is known as the **count-to-infinity problem**. The core of the problem is that when X tells Y that it has a path somewhere, Y has no way of knowing whether it itself is on the path

The delay metric was queue length, it did not take line bandwidth into account when choosing routes. The second problem the algorithm often took too long to converge (the count-to-infinity problem)

LINK STATE ROUTING

The idea behind link state routing is simple and can be stated as five parts. Each router must do the following:
1. Discover its neighbors and learn their network addresses.
2. Measure the delay or cost to each of its neighbors.
3. Construct a packet telling all it has just learned.
4. Send this packet to all other routers.
5. Compute the shortest path to every other router.

1. Discover its neighbors and learn their network addresses.

- Send a special HELLO packet on each point-to-point line.
- The router on the other end is expected to send back a reply telling who it is.
- When two or more routers are connected by a LAN.
- model the LAN as a node itself

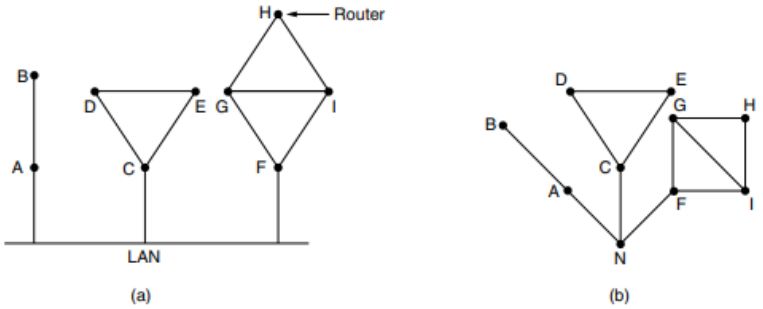


Figure 5-11. (a) Nine routers and a broadcast LAN. (b) A graph model of (a).

2. Measure the delay or cost to each of its neighbors - Setting Link Costs

- each link to have a distance or cost metric for finding shortest paths.
- make the cost inversely proportional to the bandwidth of the link.
- send over the line a special **ECHO packet** that the other side is required to send back immediately. By measuring the round-trip time and dividing it by two, the sending router can get a reasonable estimate of the delay.

3. Building Link State Packets

- each router to build a packet containing all the data.
- The packet starts with the identity of the sender, followed by a sequence number and age and a list of neighbors.

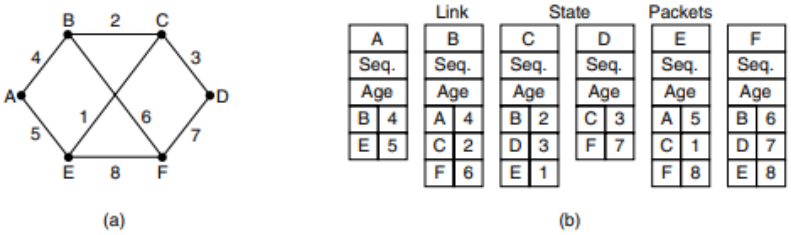


Figure 5-12. (a) A network. (b) The link state packets for this network.

determining when to build them:

- build them periodically, that is, at regular intervals.
- build them when some significant event occurs, such as a line or neighbor going down or coming back up again or changing its properties appreciably

4. **Distributing the Link State Packets**

basic distribution algorithm - use flooding to distribute the link state packets. To keep the flood in check, each packet contains a sequence number that is incremented for each new packet sent. Routers keep track of all the (source router, sequence) pairs they see. When a new link state packet comes in, it is checked against the list of packets already seen. If it is new, it is forwarded on all lines except the one it arrived on. If it is a duplicate, it is discarded. If a packet with a sequence number lower than the highest one seen so far ever arrives, it is rejected as being obsolete since the router has more recent data.

When a link state packet comes in to a router for flooding, it is not queued for transmission immediately. Instead, it is put in a holding area to wait a short while in case more links are coming up or going down. If another link state packet from the same source comes in before the first packet is transmitted, their sequence numbers are compared. If they are equal, the duplicate is discarded. If they are different, the older one is thrown out. To guard against errors on the links, all link state packets are acknowledged.

Drawbacks of algorithm:

- if the sequence numbers wrap around,- The solution here is to use a 32-bit sequence number. With one link state packet per second.
- if a router ever crashes, it will lose track of its sequence number. If it starts again at 0, the next packet will be rejected as a duplicate.
- if a sequence number is ever corrupted.

The solution to all these problems is to include **the age** of each packet after the sequence number and decrement it once per second. When the age hits zero, the information from that router is discarded.

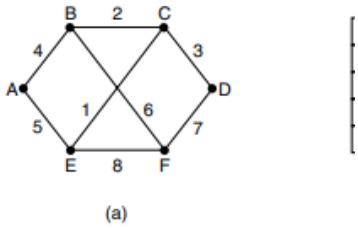


Figure 5-12. (a) A network.

Source	Seq.	Age	Send flags A	Send flags C	Send flags F	ACK flags A	ACK flags C	ACK flags F	Data
A	21	60	0	1	1	1	0	0	
F	21	60	1	1	0	0	0	1	
E	21	59	0	1	0	1	0	1	
C	20	60	1	0	1	0	1	0	
D	21	59	1	0	0	0	1	1	

Figure 5-13. The packet buffer for router *B* in Fig. 5-12(a).

The data structure used by router B for the network shown in Fig. 5-12(a) is depicted in Fig. 5-13. Each row here corresponds to a recently arrived, but as yet not fully processed, link state packet. The table records where the packet originated, its sequence number and age, and the data. In addition, there are send and acknowledgement flags for each of B's three links (to A, C, and F, respectively). The send flags mean that the packet must be sent on the indicated link. The acknowledgement flags mean that it must be acknowledged there.

The link state packet from A arrives directly, so it must be sent to C and F and acknowledged to A, as indicated by the flag bits. Similarly, the packet from F has to be forwarded to A and C and acknowledged to F. However, the situation with the third packet, from E, is different. It arrives twice, once via EAB and once via EFB.

Consequently, it has to be sent only to C but must be acknowledged to both A and F, as indicated by the bits. If a duplicate arrives while the original is still in the buffer, bits have to be changed. For example, if a copy of C's state arrives from F before the fourth entry in the table has been forwarded, the six bits will be changed to 100011 to indicate that the packet must be acknowledged to F but not sent there.

5. **Computing the New Routes**

Once a router has accumulated a full set of link state packets, it can construct the entire network graph because every link is represented. Now Dijkstra's algorithm can be run locally to construct the shortest paths to all possible destinations. This information is installed in the routing tables, and normal operation is resumed.

For a network with n routers, each of which has k neighbors, the memory required to store the input data is proportional to kn.

Any cast:

Delivery models in which a source sends to a single destination called **unicast,** to all destinations called **broadcast,** and to a group of destinations called **multicast**. Another delivery model, called **anycast** is sometimes also useful. In anycast, a packet is delivered to the nearest member of a group. Schemes that find these paths are called **anycast** routing.

Within each network, an **intradomain or interior gateway protocol** is used for routing. Across the networks that make up the internet, an **interdomain or exterior gateway protocol** is used. The networks may all use different intradomain protocols, but they must use the same interdomain protocol. In the Internet, the interdomain routing protocol is called **BGP (Border Gateway Protocol)**.
Each network is operated independently of all the others, it is often referred to as an **AS (Autonomous System).** Eg: an ISP network. Routing inside an autonomous system is called **Interior Routing (**RIP, OSPF**).** Routing between AS is called as **Exterior Routing (BGP).**

ROUTING INFORMATION PROTOCOL:

It is an interior router protocol used inside an autonomous system. It is based on distance vector routing

Points to be remembered in RIP:

- in autonomous system we consider routers and networks. Routers have routing table while networks do not have.
- Destination of a routing table in a network which is the first column of routing table.
- Metric used is the distance which is the number of links to reach the destination called hop count.
- Infinity is defined as 16 means the maximum hop permitted is 15.
- Next hop column defines address of the router to which packet to be sent to reach the destination.

Destination	Hop count	Next hop	Other information

Routing table updation is done using RIP Response message.

RIP Routing Table Updation algorithm:

- RIP response message arrived
- Increment hop count by one for each advertised destination
- Repeat the following steps for each advertised destination
    - Add advertised information to the table if destination Is not present in the routing table.
    - Replace entry in the table with advertised one in next hop field is same
    - Replace entry in routing table if advertising hop count is smaller than the one in table.

RIP message format:

RIP message are of two types:

1. Message that deliver routing information
2. Message that request routing information

Command	version	All zeros
Address family identifier		All zeros
IP address		
All zeros		
All zeros		
Metric PC		
Repeat of last 20 bytes		

Command – indicate if it is request or response.
Version – version used
Zeros –provide backup compatibility with previous standards of RIP
AFI – specifies the address
IP address – ip address for the entry
Metric – number of internetwork hops encountered. Maximum of 15.

Disavantages:

- It only understands the shortest route to destination based on the count of hops
- Depends on routers for computing routing update
- Routing table need to be broadcasted every 30secnds
- Distance are based on hops not on cost

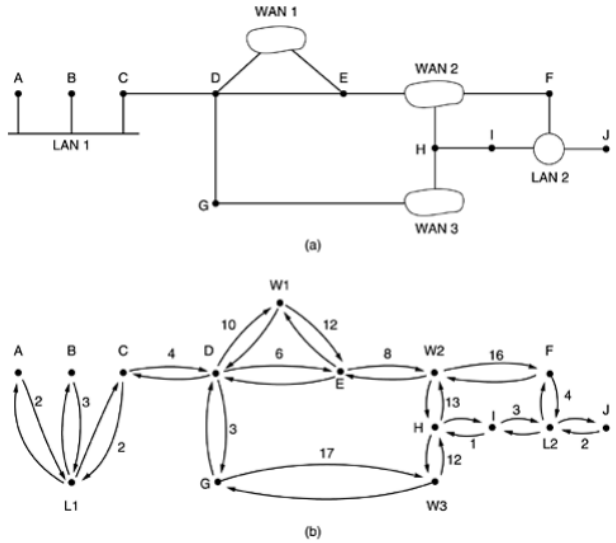
OSPF (OPEN SHORTEST PATH FIRST) - The Interior Gateway Routing Protocol

The Internet is made up of a large number of autonomous systems. Each AS is operated by a different organization and can use its own routing algorithm inside. For example, the internal networks of companies X, Y, and Z are usually seen as three ASes if all three are on the Internet. All three may use different routing algorithms internally. A routing algorithm within an AS is called an **interior gateway protocol**; an algorithm for routing between ASes is called an **exterior gateway protocol.**

OSPF supports three kinds of connections and networks:
1. Point-to-point lines between exactly two routers.
2. Multiaccess networks with broadcasting (e.g., most LANs).
3. Multiaccess networks without broadcasting (e.g., most packet-switched WANs).

A multiaccess network is one that can have multiple routers on it, each of which can directly communicate with all the others. All LANs and WANs have this property. Figure 5-64(a) shows an AS containing all three kinds of networks. Note that hosts do not generally play a role in OSPF.

Figure 5-64. (a) An autonomous system. (b) A graph representation of (a).



OSPF operates by abstracting the collection of actual networks, routers, and lines into a directed graph in which each arc is assigned a cost (distance, delay, etc.). It then computes the shortest path based on the weights on the arcs. A serial connection between two routers is represented by a pair of arcs, one in each direction. Their weights may be different. A multiaccess network is represented by a node for the network itself plus a node for each router. The arcs from the network node to the routers have weight 0 and are omitted from the graph.

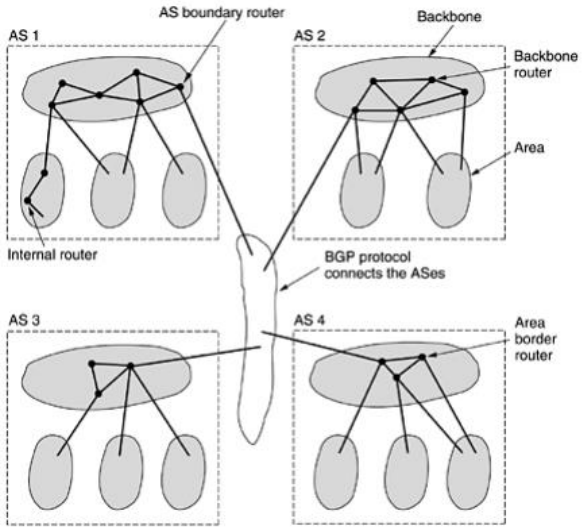
OSPF allows ASes to be divided into numbered **areas**, where an area is a network or a set of contiguous networks. Areas do not overlap but need not be exhaustive, that is, some routers may belong to no area. An area is a generalization of a subnet. Outside an area, its topology and details are not visible.

Every AS has a **backbone area,** called **area 0**. All areas are connected to the backbone, possibly by tunnels, so it is possible to go from any area in the AS to any other area in the AS via the backbone.

During normal operation, three kinds of routes may be needed:

1. intra-area - Intra-area routes are the easiest, since the source router already knows the shortest path to the destination router
2. interarea - Interarea routing always proceeds in three steps:
    a. go from the source to the backbone;
    b. go across the backbone to the destination area;
    c. go to the destination
3. inter-AS.

Fig 5-65 :The relation between ASes, backbones, and areas in OSPF.



OSPF distinguishes four classes of routers:

1. Internal routers are wholly within one area.
2. Area border routers connect two or more areas.
3. Backbone routers are on the backbone.
4. AS boundary routers talk to routers in other ASes.

Figure 5-66. The five types of OSPF messages.

Message type	Description
Hello	Used to discover who the neighbors are
Link state update	Provides the sender's costs to its neighbors
Link state ack	Acknowledges link state update
Database description	Announces which updates the sender has
Link state request	Requests information from the partner

When a router boots, it sends **HELLO** messages on all of its point-to-point lines and multicasts them on LANs to the group consisting of all the other routers.

One router is elected as the designated router. It is said to be adjacent to all the other routers on its LAN, and exchanges information with them. Neighboring routers that are not adjacent do not exchange information with each other. A backup designated router is always kept up to date to ease the transition should the primary designated router crash and need to replaced immediately.

Each router periodically floods **LINK STATE UPDATE** messages to each of its adjacent routers. This message gives its state and provides the costs used in the topological database. The flooding messages are acknowledged, to make them reliable. Each message has a sequence number, so a router can see whether an incoming LINK STATE UPDATE is older or newer than what it currently has. Routers also send these messages when a line goes up or down or its cost changes.

DATABASE DESCRIPTION messages give the sequence numbers of all the link state entries currently held by the sender. By comparing its own values with those of the sender, the receiver can determine who has the most recent values. These messages are used when a line is brought up.

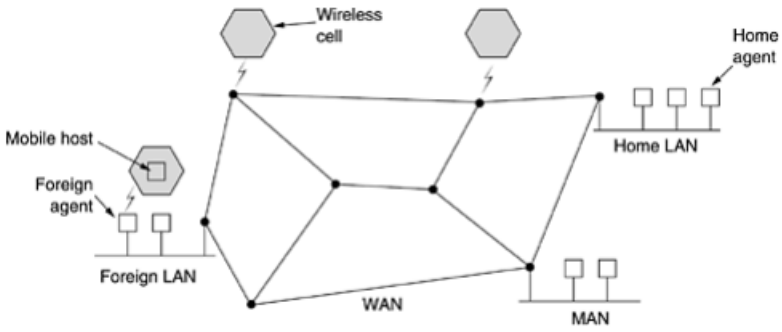
Either partner can request link state information from the other one by using **LINK STATE REQUEST** messages. The result of this algorithm is that each pair of adjacent routers checks to see who has the most recent data, and new information is spread throughout the area this way. All these messages are sent as raw IP packets.

Using flooding, each router informs all the other routers in its area of its neighbors and costs. This information allows each router and The backbone area to construct the graph for its area(s) and compute the shortest path. In addition, the backbone routers accept information from the area border routers in order to compute the best route from each backbone router to every other router. This information is propagated back to the area border routers, which advertise it within their areas. Using this information, a router about to send an interarea packet can select the best exit router to the backbone.

ROUTING FOR MOBILE HOSTS

The model of the world that network designers typically use is shown in Fig. 5-18.

Figure 5-18. A WAN to which LANs, MANs, and wireless cells are attached.



Hosts that never move are said to be **stationary.** They are connected to the network by copper wires or fiber optics. **Migratory hosts** are basically stationary hosts who move from one fixed site to another from time to time but use the network only when they are physically connected to it. **Roaming hosts** actually compute on the run and want to maintain their connections as they move around. We will use the term **mobile hosts** to mean either of the latter two categories, that is, all hosts that are away from home and still want to be connected.

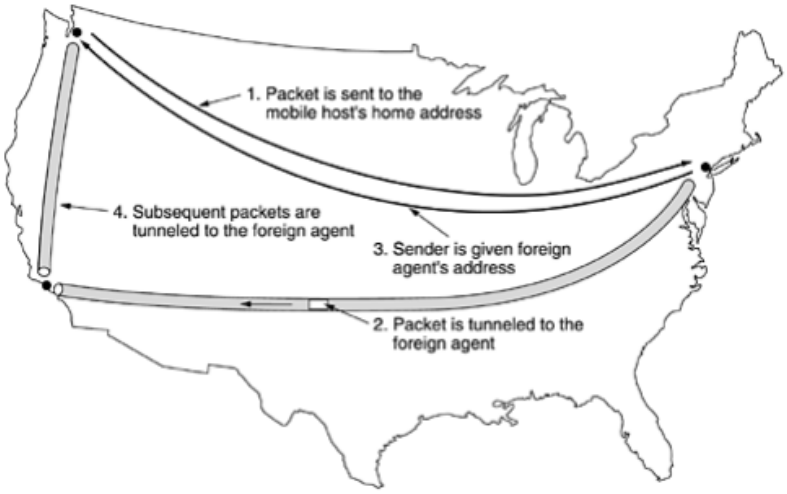
All hosts are assumed to have a permanent home location that never changes. Hosts also have a permanent home address that can be used to determine their home locations, analogous to the way the telephone number 1-212-5551212 indicates the United States (country code 1) and Manhattan (212). The routing goal in systems with mobile hosts is to make it possible to send packets to mobile hosts using their home addresses and have the packets efficiently reach them wherever they may be.

the world is divided up (geographically) into small units. Called **areas**, where an area is typically a LAN or wireless cell. Each area has one or more foreign agents, which are processes that keep track of all mobile hosts visiting the area. In addition, each area has a home agent, which keeps track of hosts whose home is in the area, but who are currently visiting another area.

The registration procedure typically works like this:

1. Periodically, each foreign agent broadcasts a packet announcing its existence and address. A newly-arrived mobile host may wait for one of these messages, but if none arrives quickly enough, the mobile host can broadcast a packet.
2. The mobile host registers with the foreign agent, giving its home address, current data link layer address, and some security information.
3. The foreign agent contacts the mobile host's home agent. It also includes the security information to convince the home agent that the mobile host is really there.
4. The home agent examines the security information, which contains a timestamp, to prove that it was generated within the past few seconds letting he foreign agent to proceed.
5. When the foreign agent gets the acknowledgement from the home agent, it makes an entry in its tables and informs the mobile host that it is now registered.

Figure 5-19. Packet routing for mobile hosts.



The home agent then does two things.

1. it encapsulates the packet in the payload field of an outer packet and sends the latter to the foreign agent (step 2 in Fig. 5-19). This mechanism is called **tunneling;** After getting the encapsulated packet, the foreign agent removes the original packet from the payload field and sends it to the mobile host as a data link frame.
2. the home agent tells the sender to henceforth send packets to the mobile host by encapsulating them in the payload of packets explicitly addressed to the foreign agent instead of just sending them to the mobile host's home address (step 3). Subsequent packets can now be routed directly to the host via the foreign agent (step 4), bypassing the home location entirely.