5

THE NETWORK LAYER

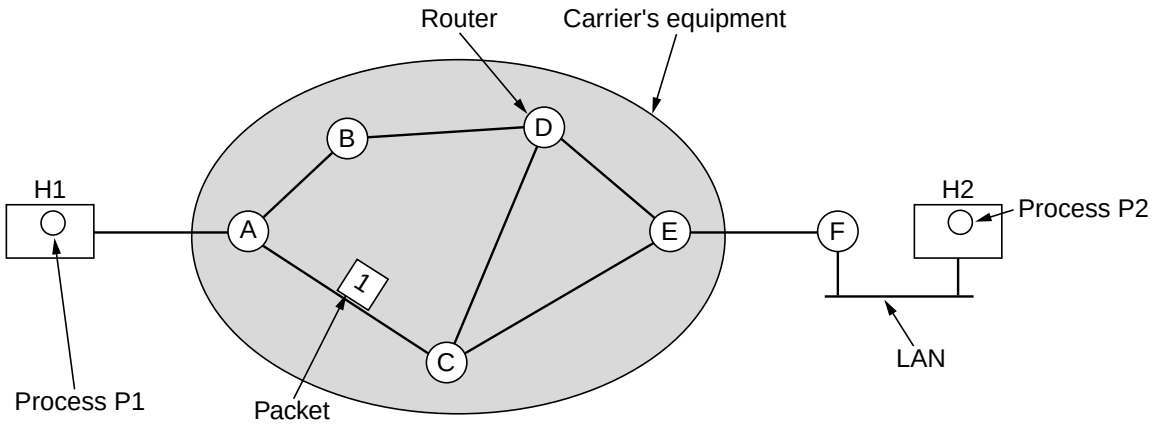


Fig. 5-1. The environment of the network layer protocols.

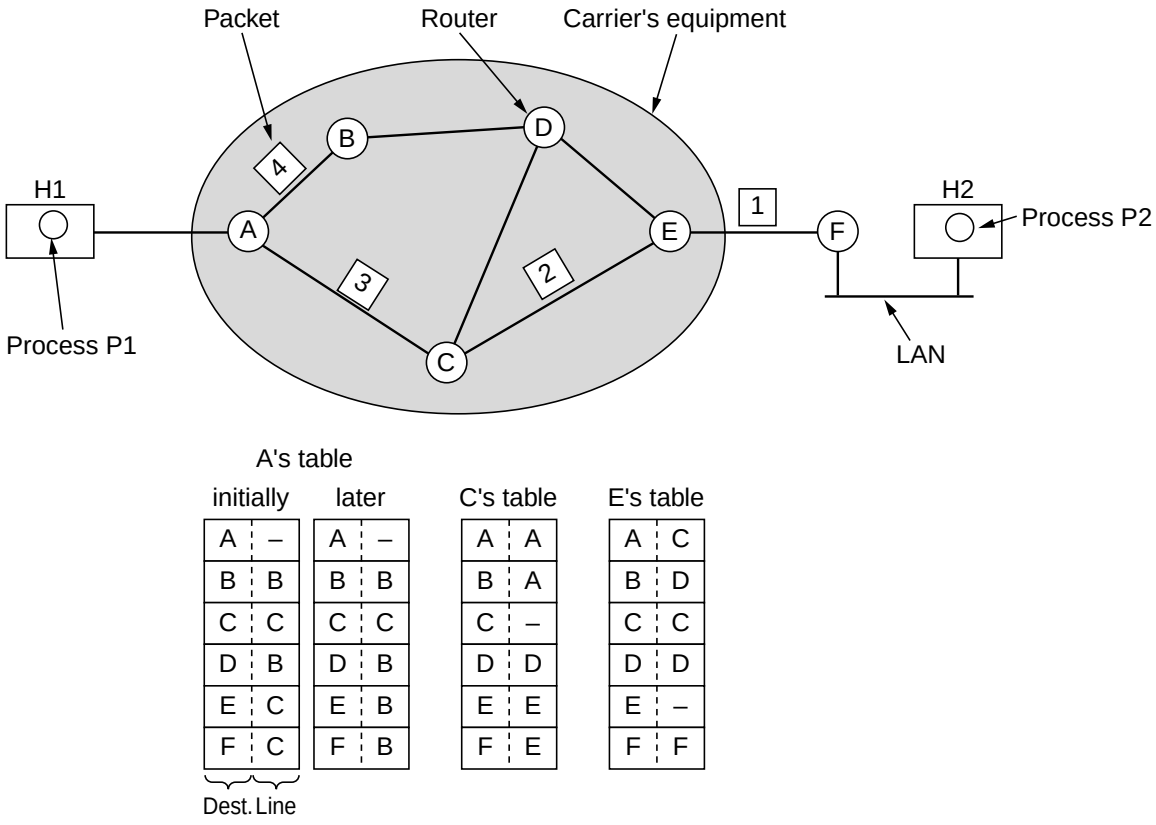


Fig. 5-2. Routing within a datagram subnet.

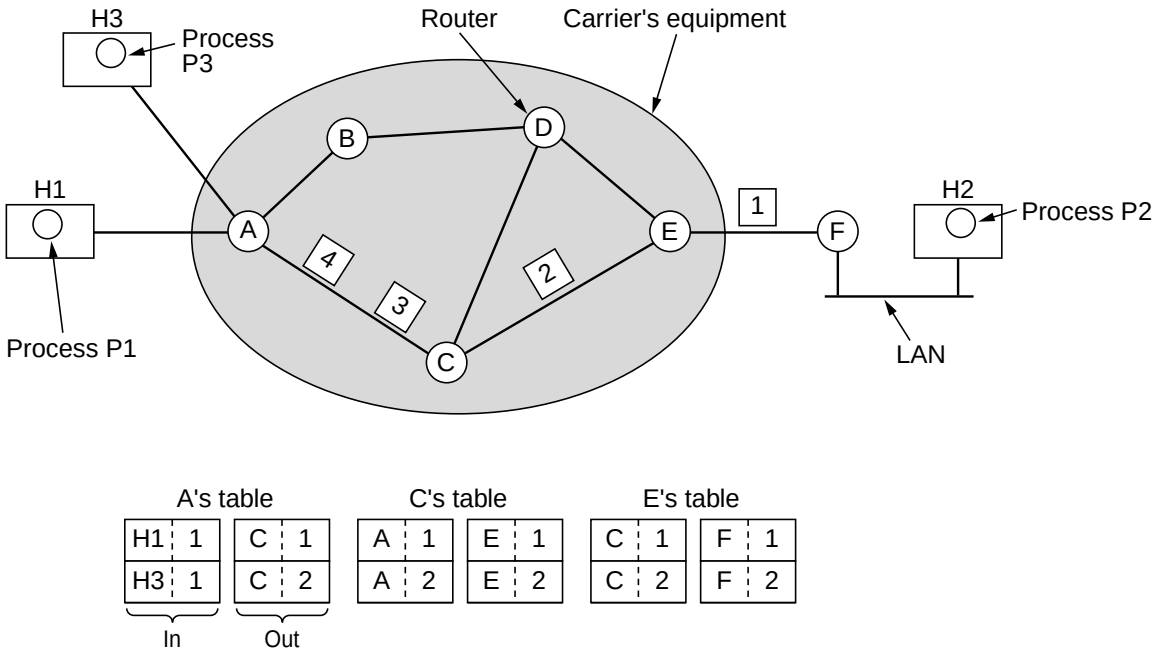


Fig. 5-3. Routing within a virtual-circuit subnet.

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

Fig. 5-4. Comparison of datagram and virtual-circuit subnets.

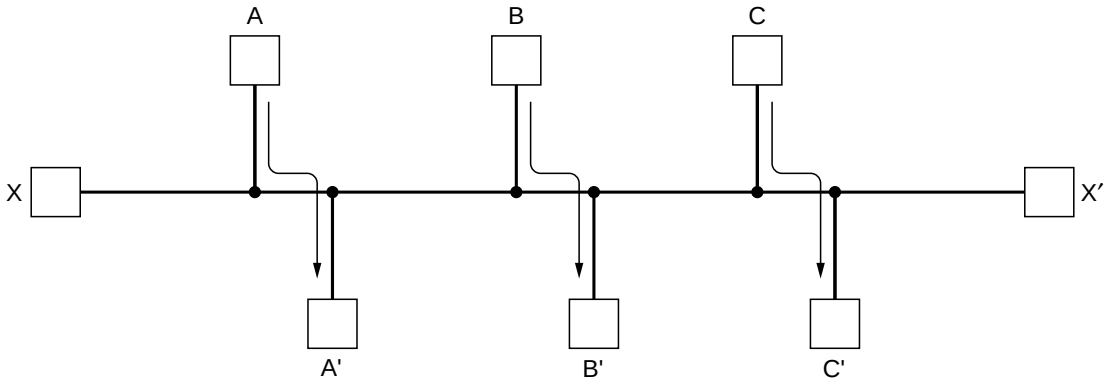


Fig. 5-5. Conflict between fairness and optimality.

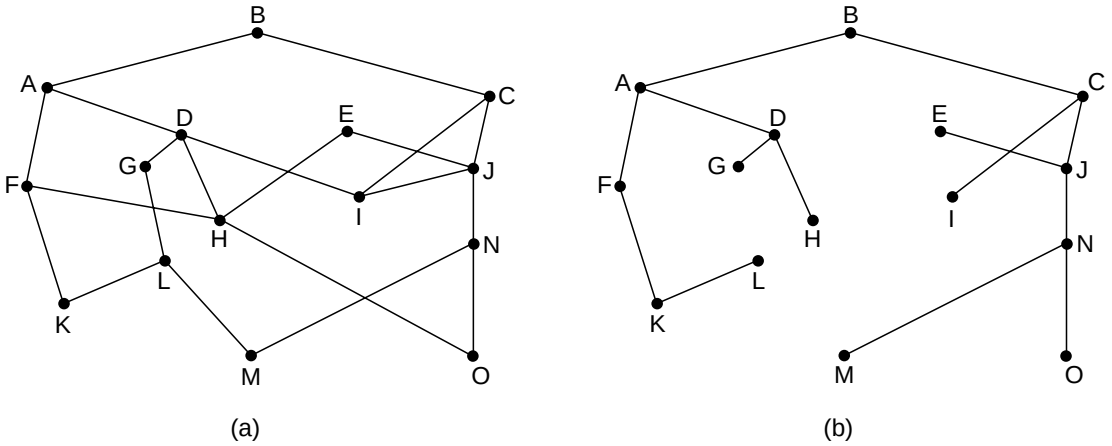


Fig. 5-6. (a) A subnet. (b) A sink tree for router *B*.

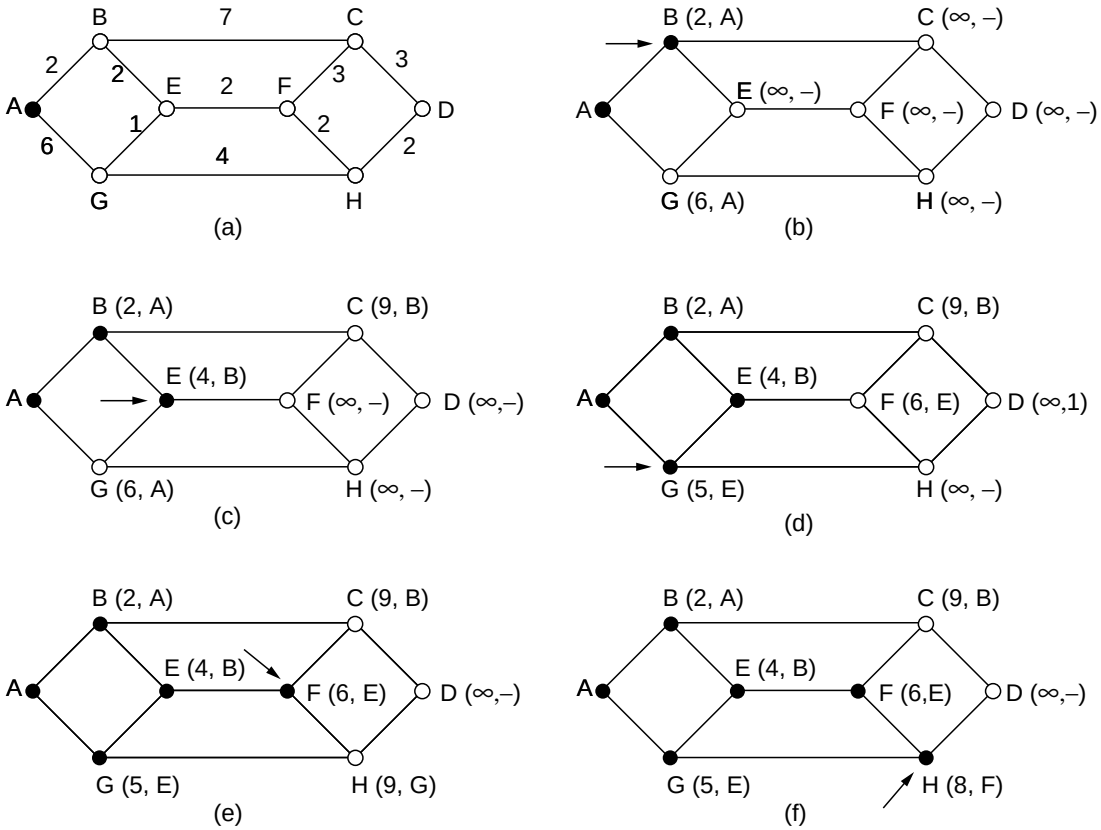


Fig. 5-7. The first five steps used in computing the shortest path from *A* to *D*. The arrows indicate the working node.

```
#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);
}
```

Fig. 5-8. Dijkstra's algorithm to compute the shortest path through a graph.

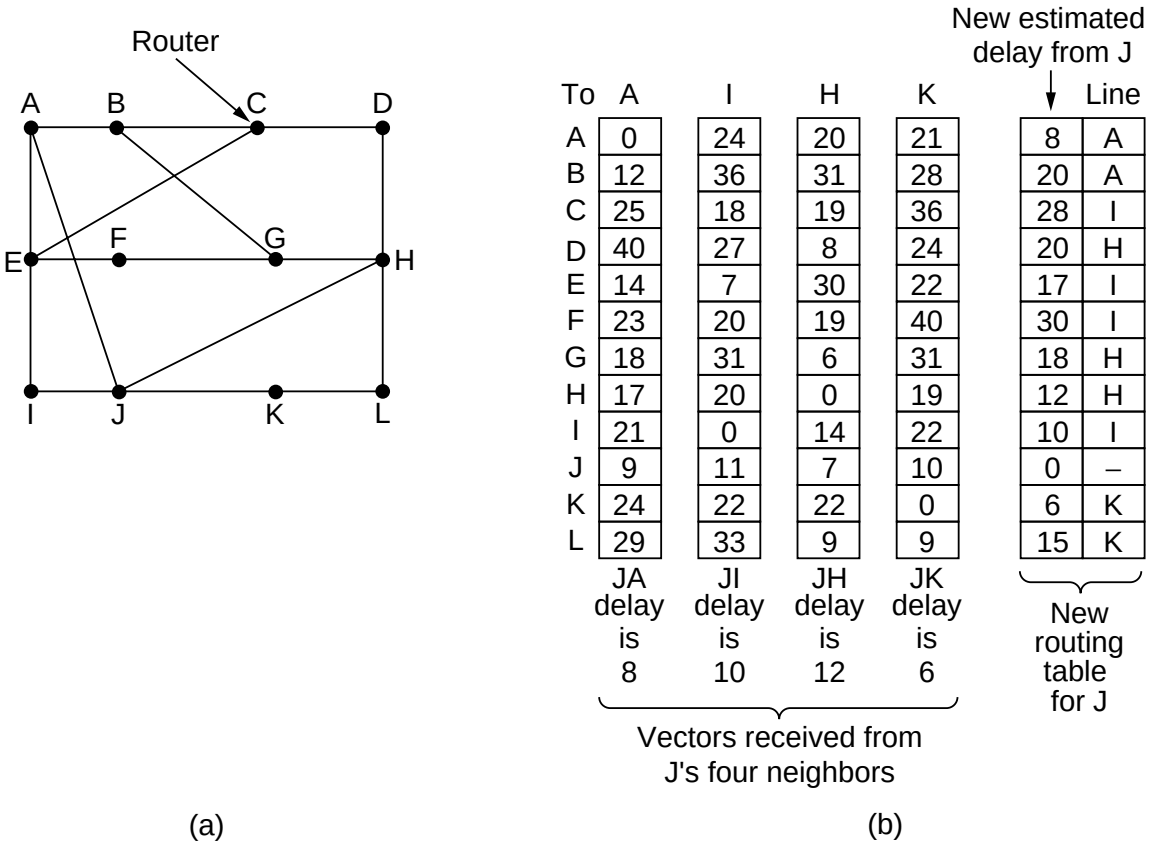


Fig. 5-9. (a) A subnet. (b) Input from *A*, *I*, *H*, *K*, and the new routing table for *J*.

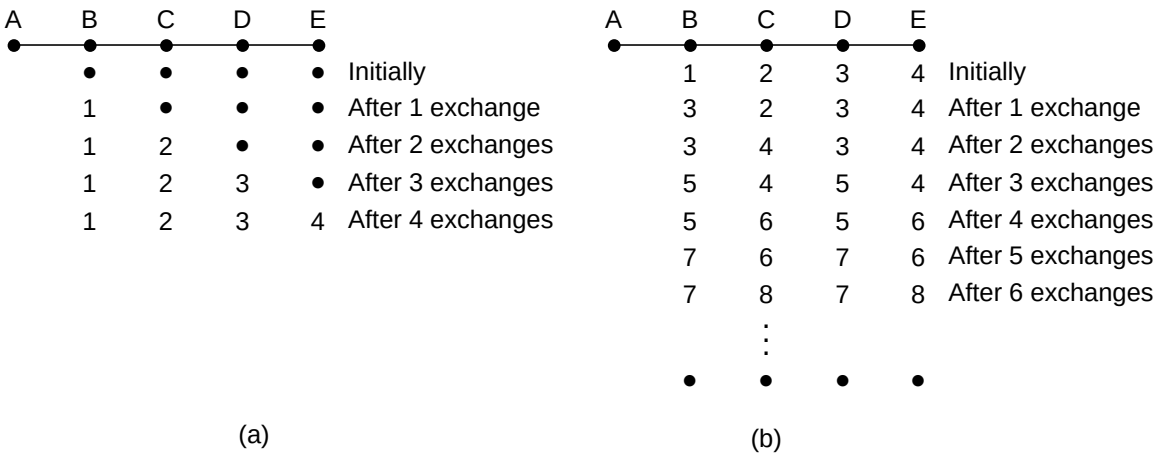


Fig. 5-10. The count-to-infinity problem.

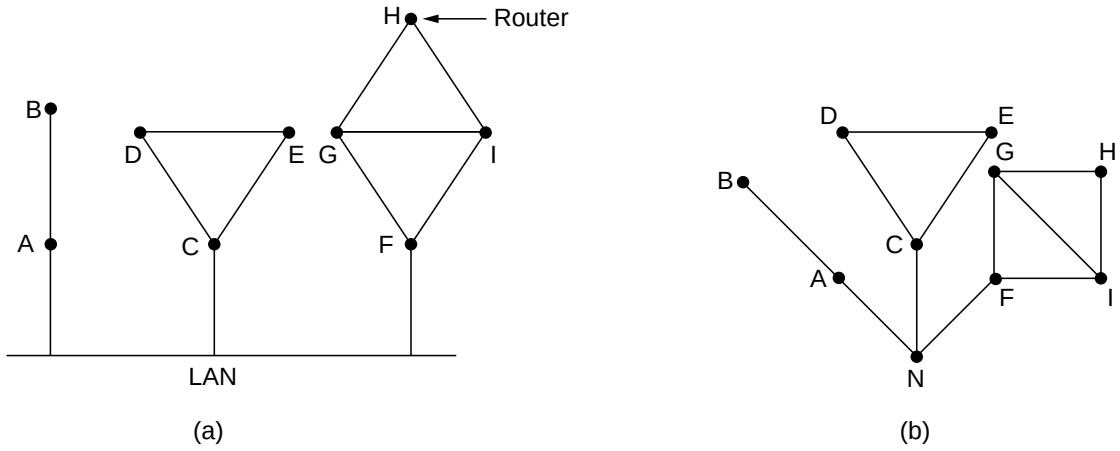


Fig. 5-11. (a) Nine routers and a LAN. (b) A graph model of (a).

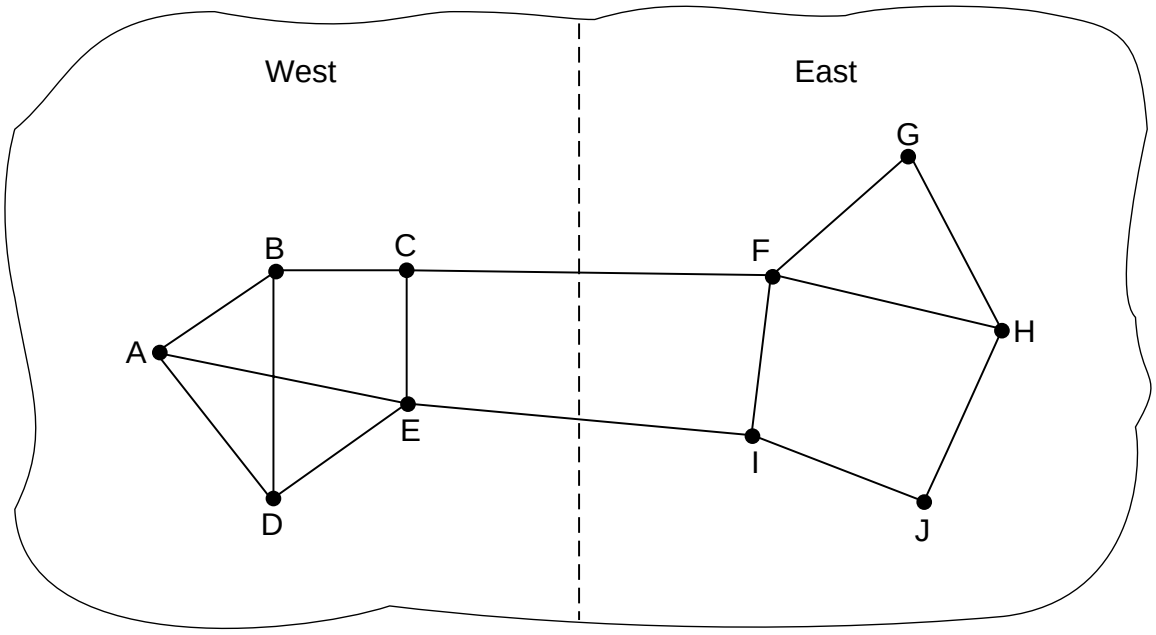
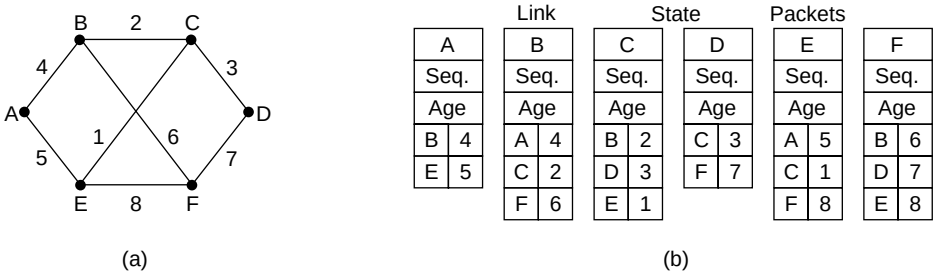


Fig. 5-12. A subnet in which the East and West parts are connected by two lines.

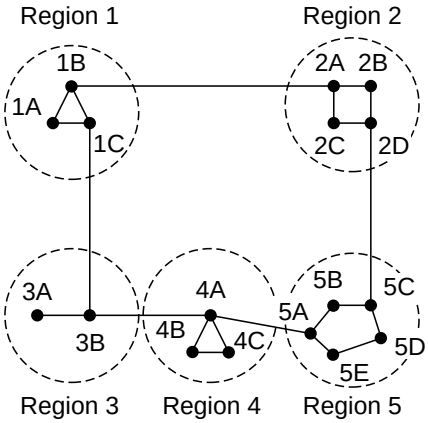


(a) (b)

Fig. 5-13. (a) A subnet. (b) The link state packets for this subnet.

			Send flags			ACK flags			
Source	Seq.	Age	A	C	F	A	C	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	

Fig. 5-14. The packet buffer for router *B* in Fig. 5-0.



(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. 5-15. Hierarchical routing.

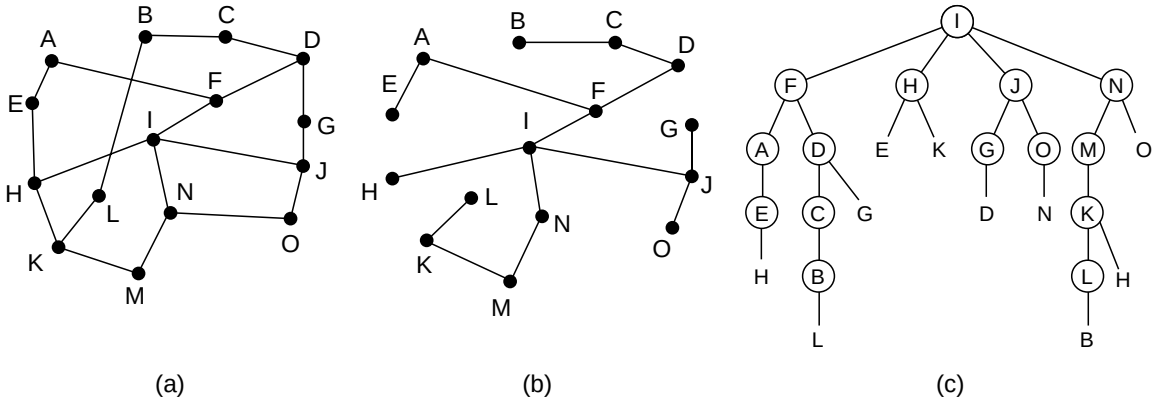


Fig. 5-16. Reverse path forwarding. (a) A subnet. (b) A sink tree. (c) The tree built by reverse path forwarding.

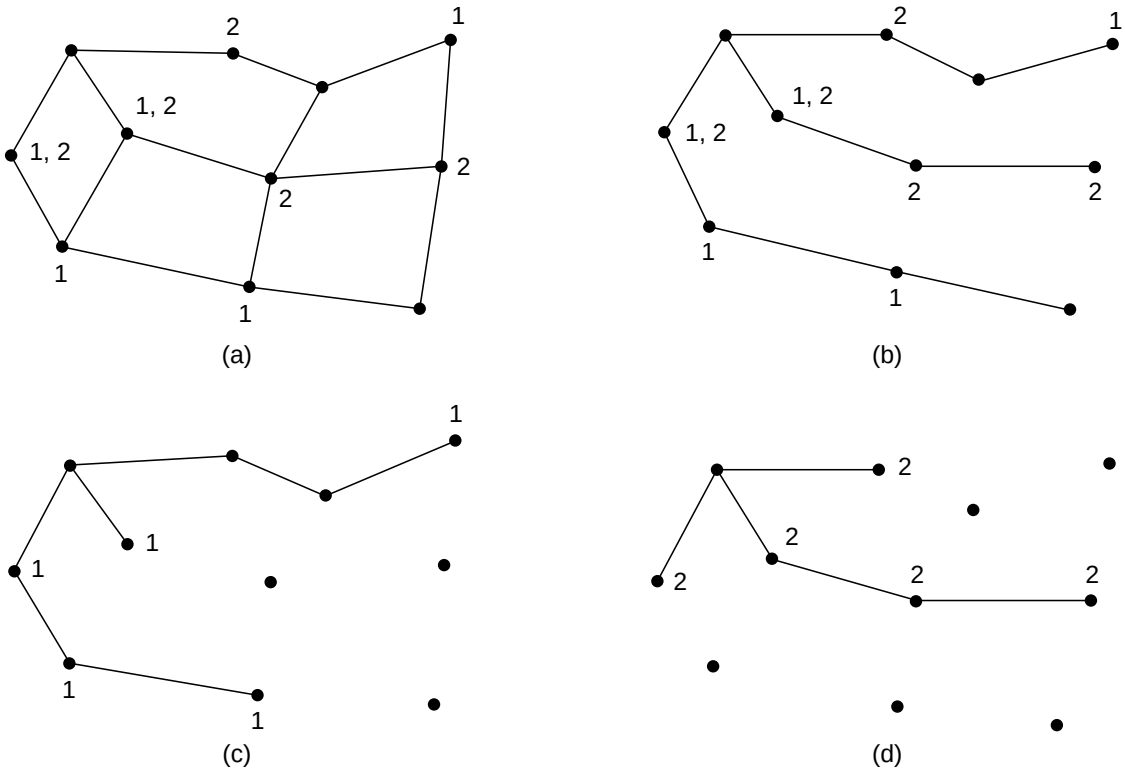


Fig. 5-17. (a) A network. (b) A spanning tree for the leftmost router. (c) A multicast tree for group 1. (d) A multicast tree for group 2.

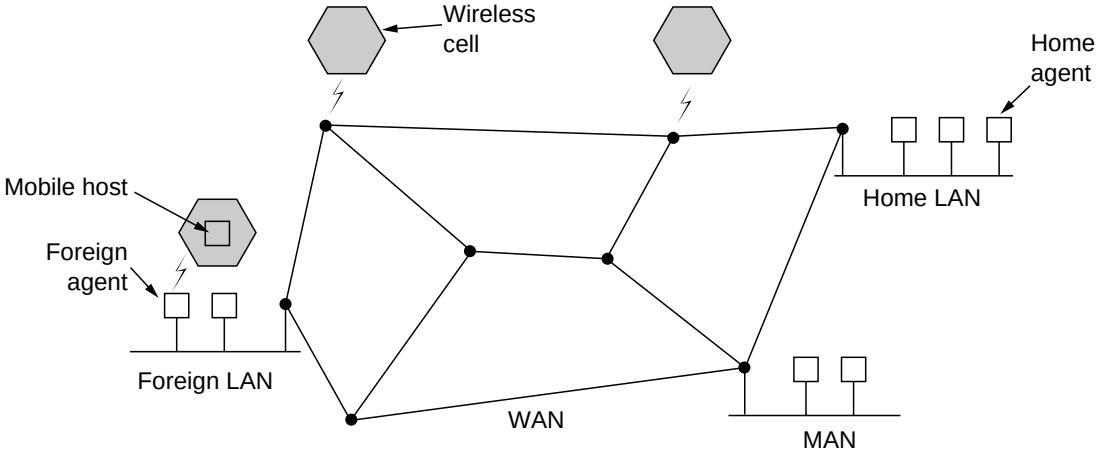


Fig. 5-18. A WAN to which LANs, MANs, and wireless cells are attached.

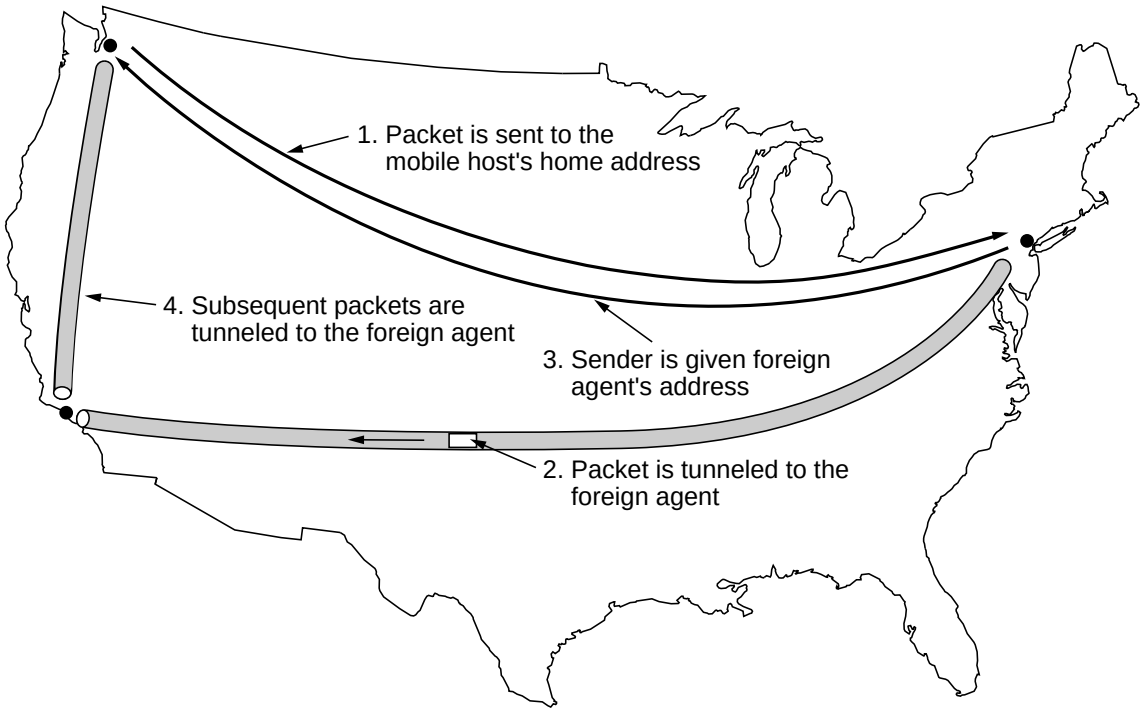


Fig. 5-19. Packet routing for mobile hosts.

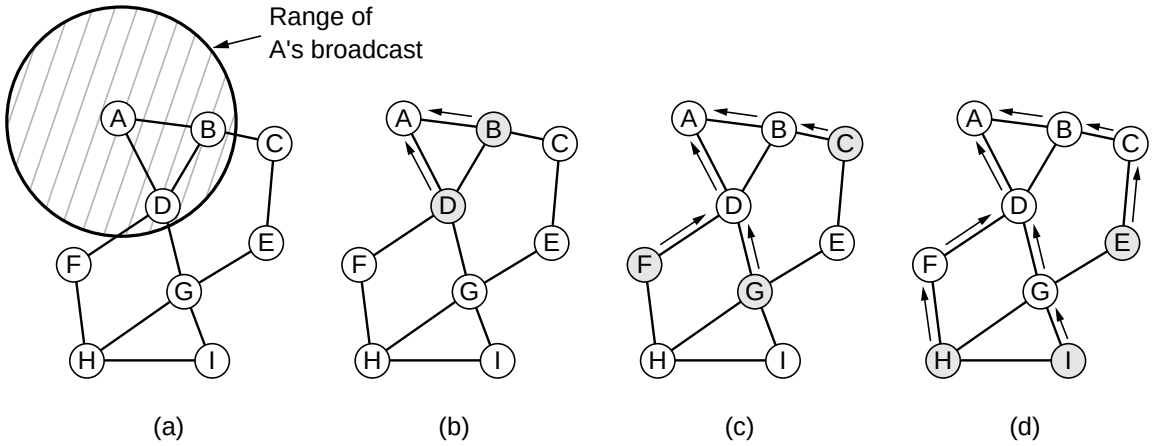


Fig. 5-20. (a) Range of *A*'s broadcast. (b) After *B* and *D* have received *A*'s broadcast. (c) After *C*, *F*, and *G* have received *A*'s broadcast. (d) After *E*, *H*, and *I* have received *A*'s broadcast. The shaded nodes are new recipients. The arrows show the possible reverse routes.

Source address	Request ID	Destination address	Source sequence #	Dest. sequence #	Hop count

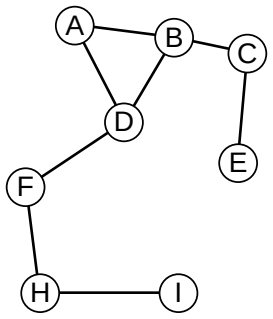
Fig. 5-21. Format of a ROUTE REQUEST packet.

Source address	Destination address	Destination sequence #	Hop count	Lifetime

Fig. 5-22. Format of a ROUTE REPLY packet.

Dest.	Next hop	Distance	Active neighbors	Other fields
A	A	1	F, G	
B	B	1	F, G	
C	B	2	F	
E	G	2		
F	F	1	A, B	
G	G	1	A, B	
H	F	2	A, B	
I	G	2	A, B	

(a)



(b)

Fig. 5-23. (a) *D*'s routing table before *G* goes down. (b) The graph after *G* has gone down.

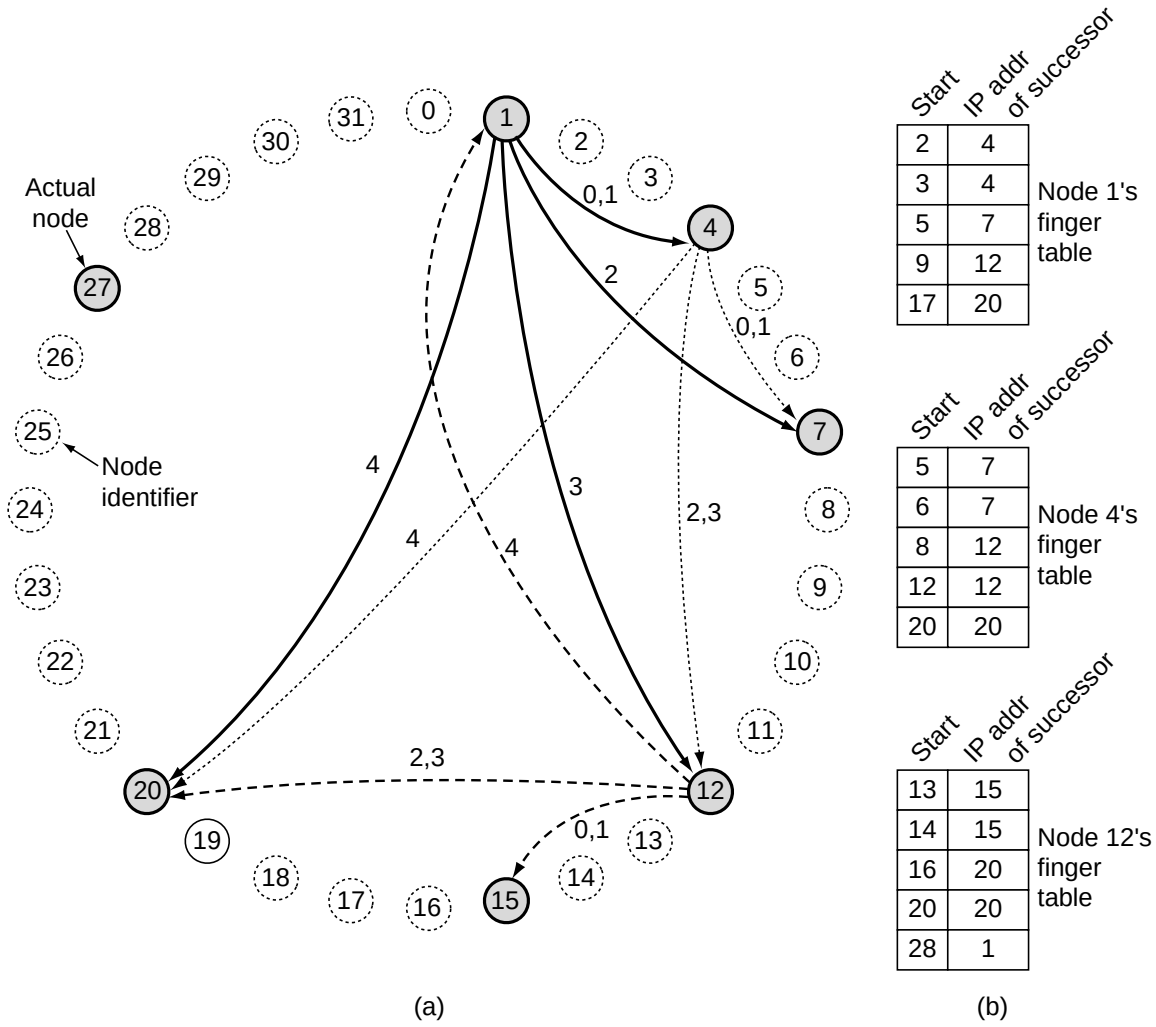


Fig. 5-24. (a) A set of 32 node identifiers arranged in a circle. The shaded ones correspond to actual machines. The arcs show the fingers from nodes 1, 4, and 12. The labels on the arcs are the table indices. (b) Examples of the finger tables.

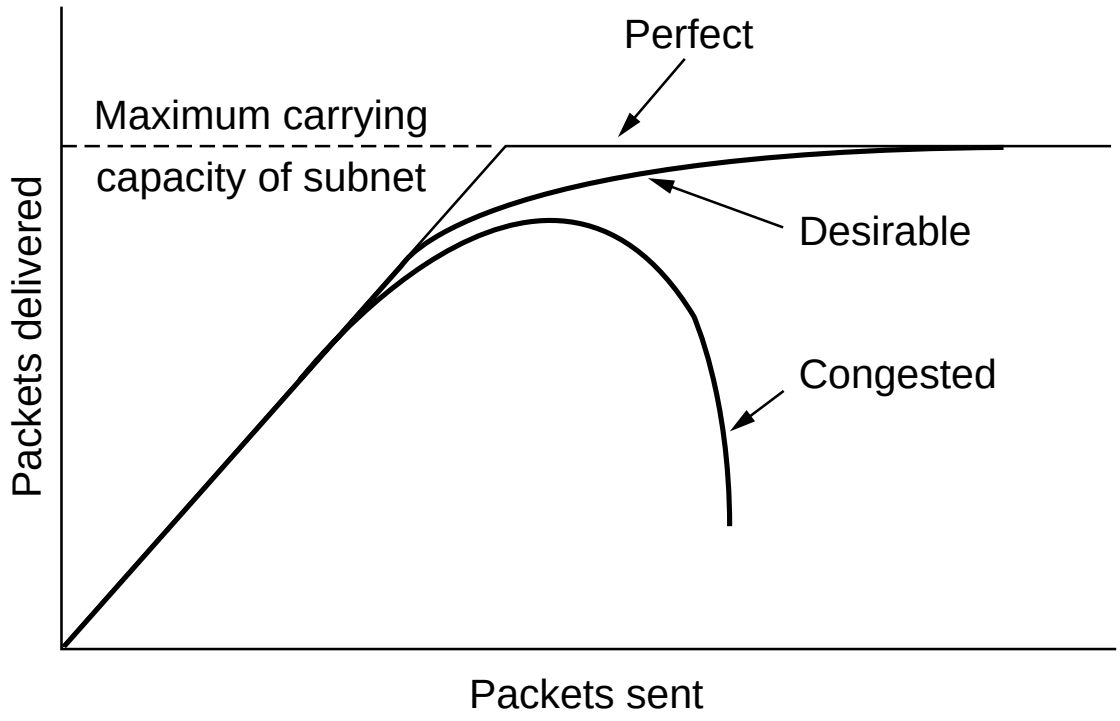


Fig. 5-25. When too much traffic is offered, congestion sets in and performance degrades sharply.

Layer	Policies
Transport	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy • Timeout determination
Network	• Virtual circuits versus datagram inside the subnet • Packet queueing and service policy • Packet discard policy • Routing algorithm • Packet lifetime management
Data link	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy

Fig. 5-26. Policies that affect congestion.

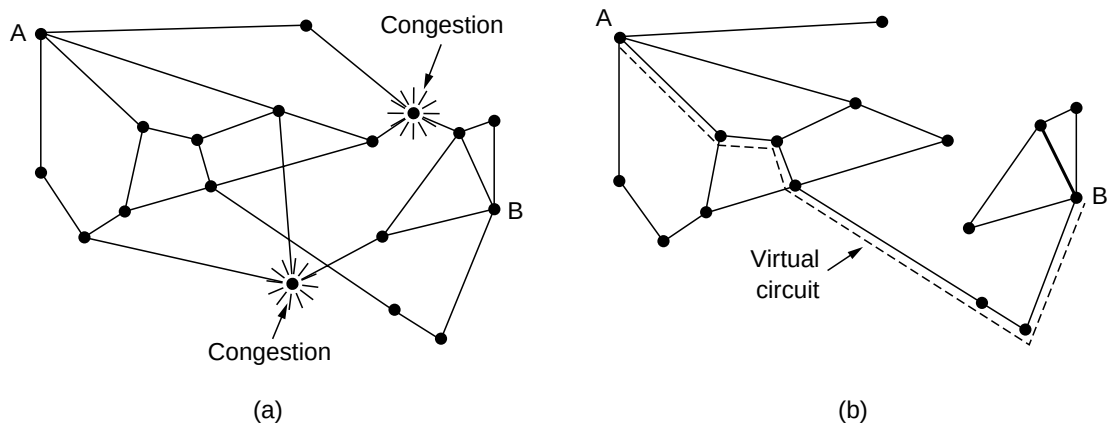


Fig. 5-27. (a) A congested subnet. (b) A redrawn subnet that eliminates the congestion. A virtual circuit from *A* to *B* is also shown.

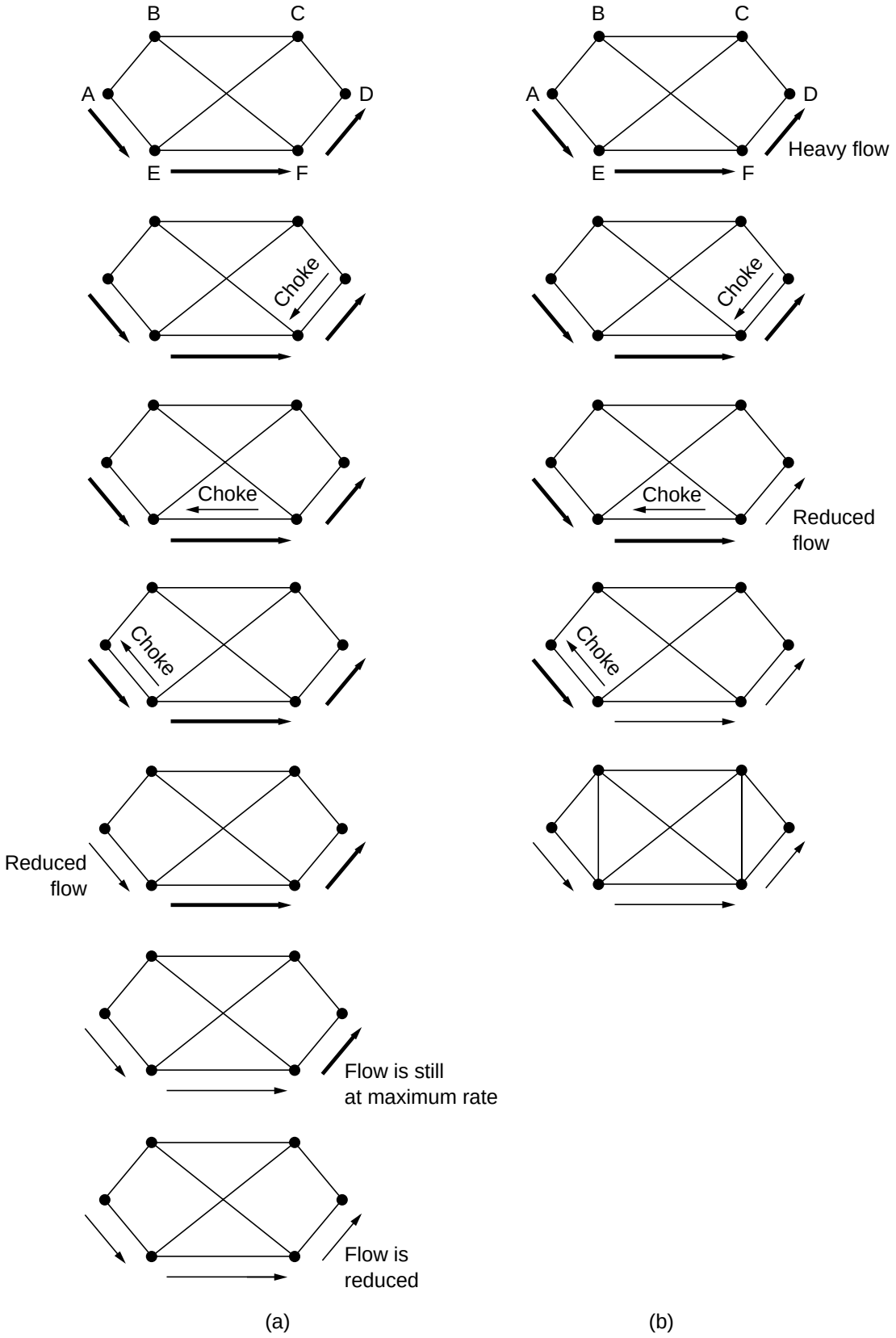


Fig. 5-28. (a) A choke packet that affects only the source. (b) A choke packet that affects each hop it passes through.

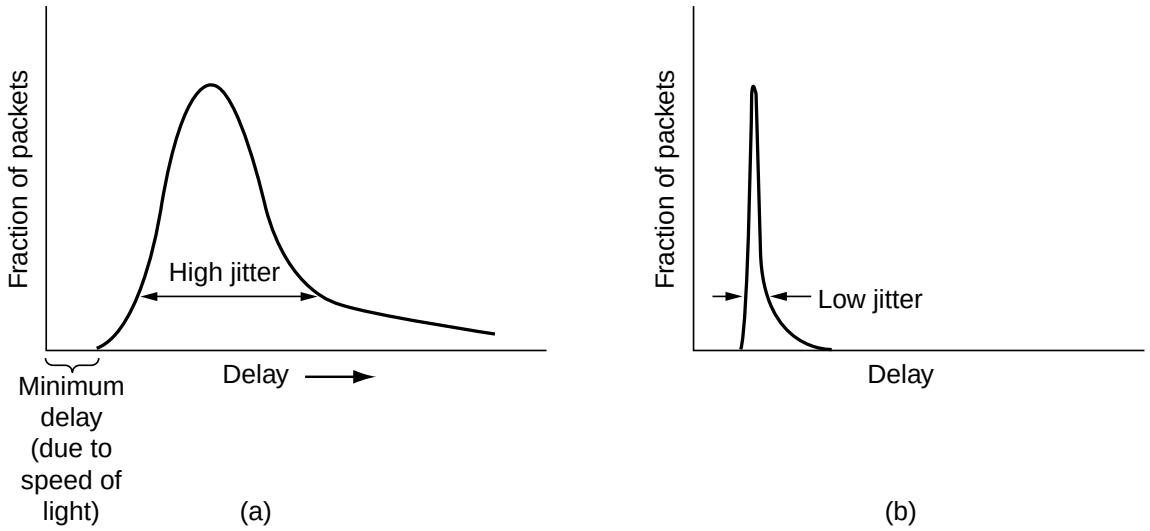


Fig. 5-29. (a) High jitter. (b) Low jitter.

Application	Reliability	Delay	Jitter	Bandwidth
E-mail	High	Low	Low	Low
File transfer	High	Low	Low	Medium
Web access	High	Medium	Low	Medium
Remote login	High	Medium	Medium	Low
Audio on demand	Low	Low	High	Medium
Video on demand	Low	Low	High	High
Telephony	Low	High	High	Low
Videoconferencing	Low	High	High	High

Fig. 5-30. How stringent the quality-of-service requirements are.

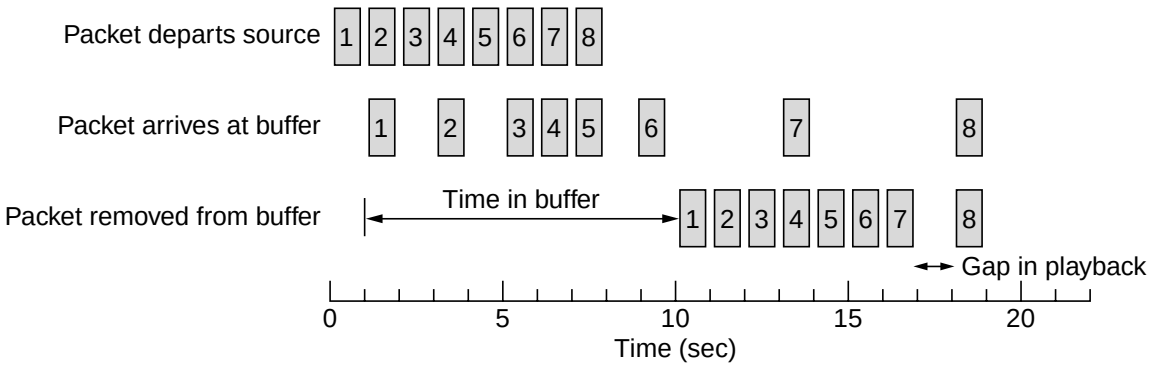


Fig. 5-31. Smoothing the output stream by buffering packets.

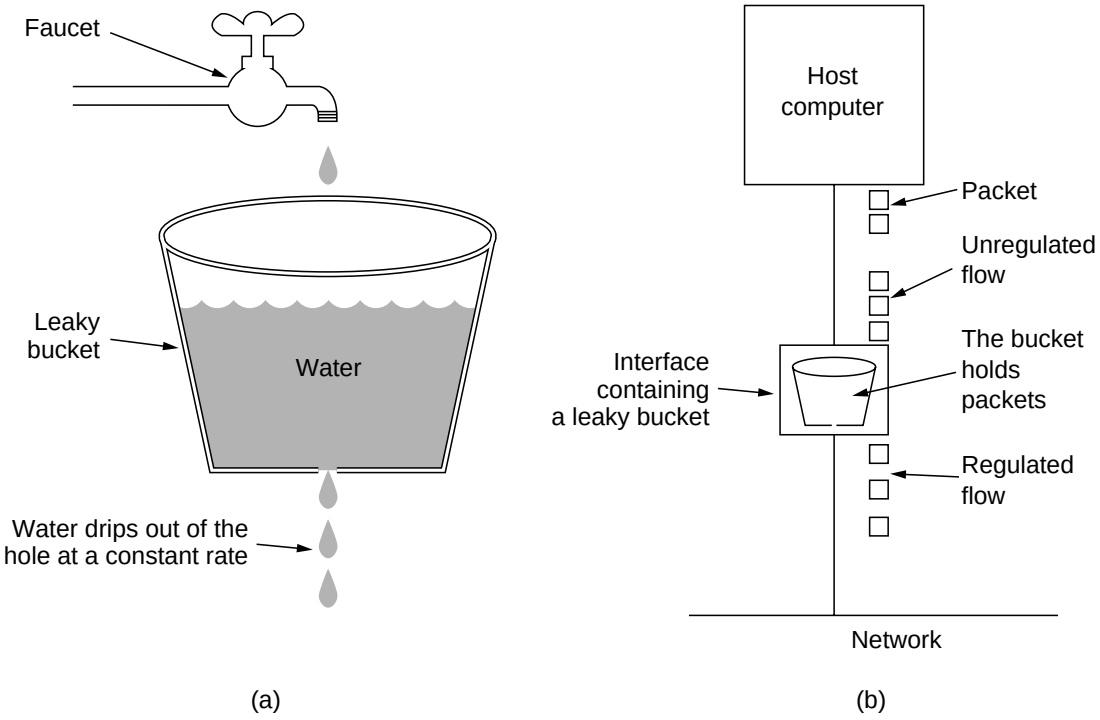


Fig. 5-32. (a) A leaky bucket with water. (b) A leaky bucket with packets.

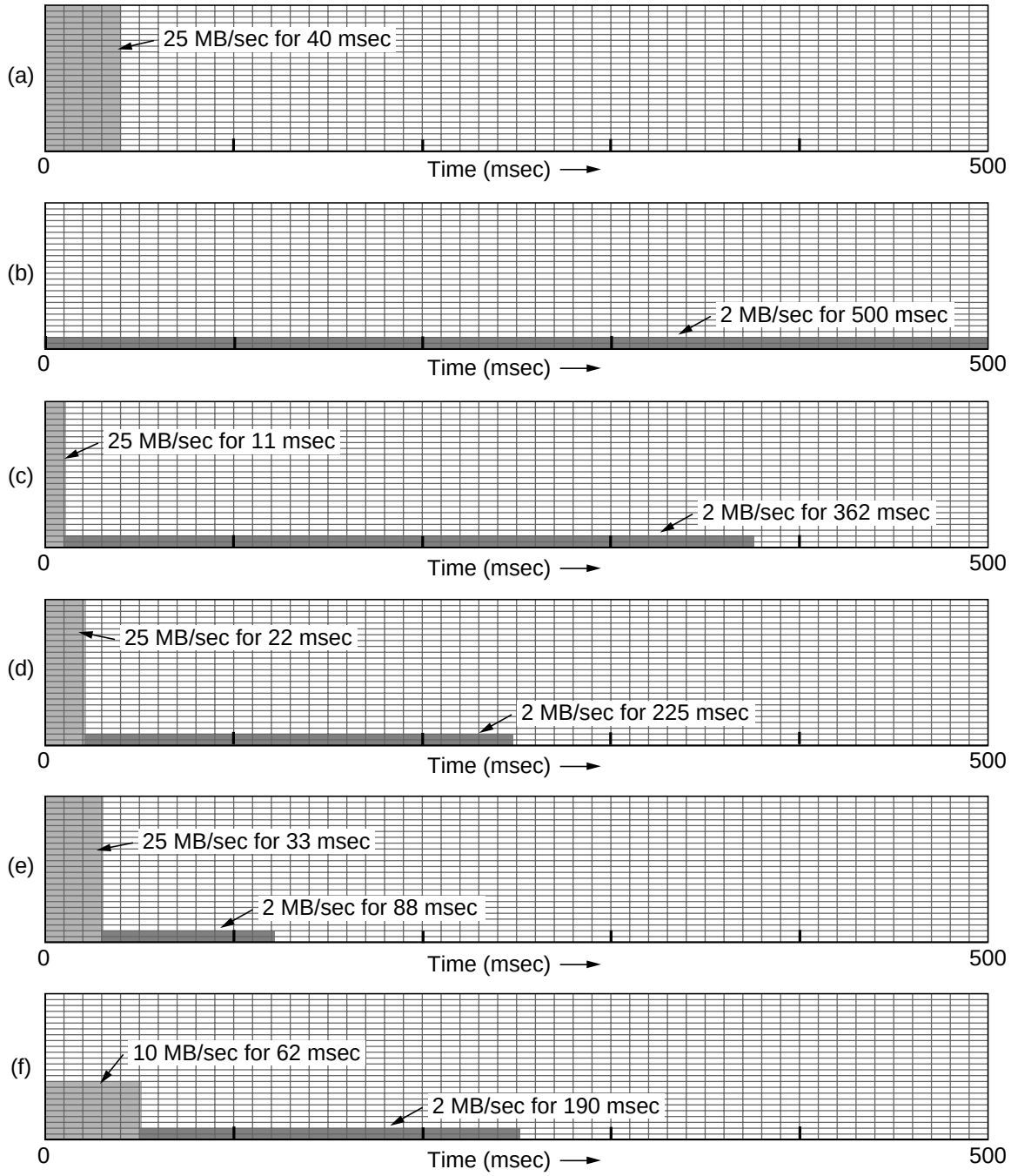


Fig. 5-33. (a) Input to a leaky bucket. (b) Output from a leaky bucket. Output from a token bucket with capacities of (c) 250 KB, (d) 500 KB, and (e) 750 KB. (f) Output from a 500KB token bucket feeding a 10-MB/sec leaky bucket.

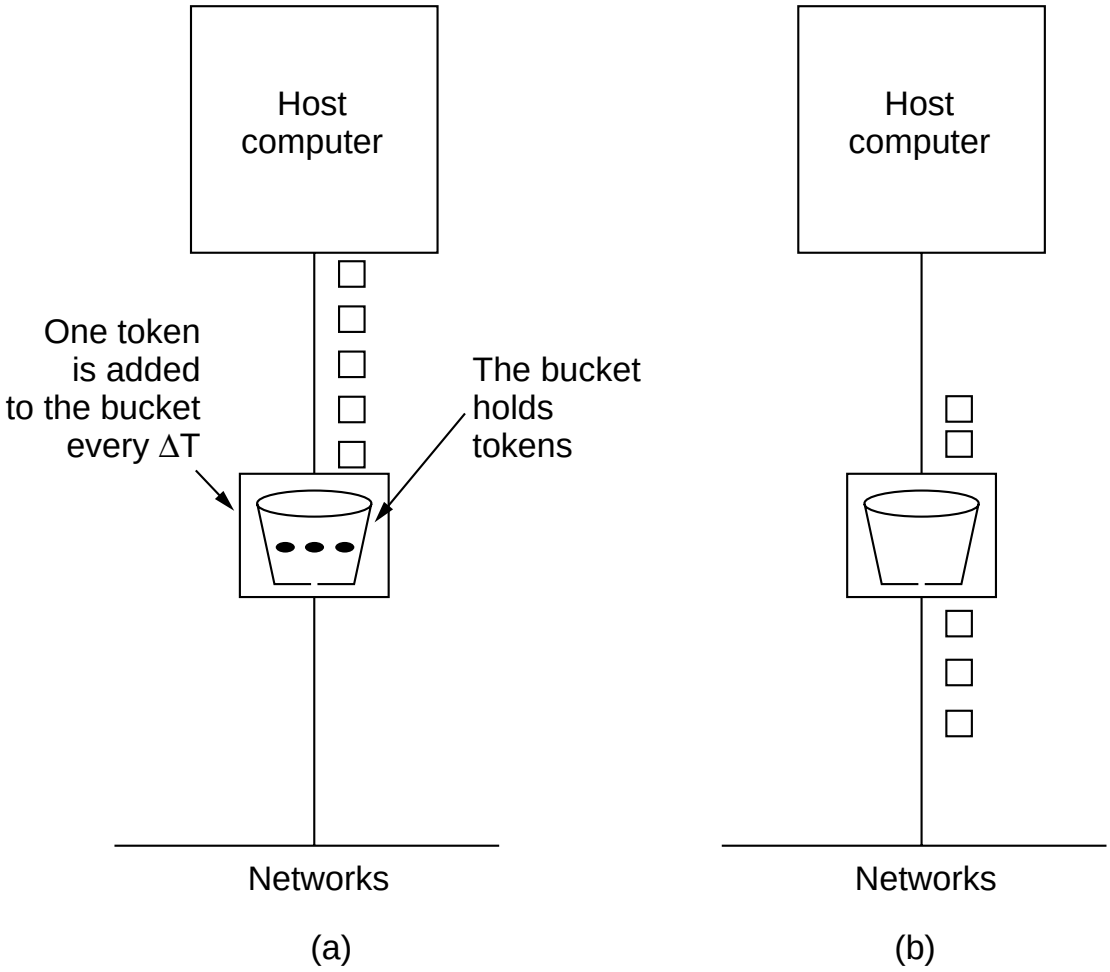
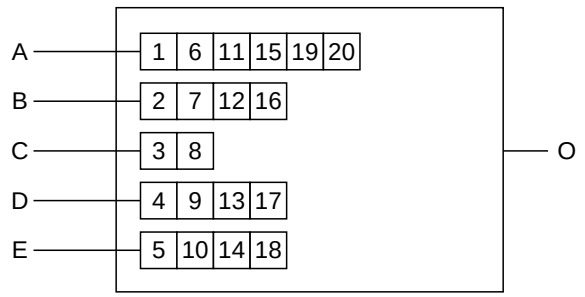


Fig. 5-34. The token bucket algorithm. (a) Before. (b) After.

Parameter	Unit
Token bucket rate	Bytes/sec
Token bucket size	Bytes
Peak data rate	Bytes/sec
Minimum packet size	Bytes
Maximum packet size	Bytes

Fig. 5-35. An example flow specification.



(a)

Packet	Finishing time
C	8
B	16
D	17
E	18
A	20

(b)

Fig. 5-36. (a) A router with five packets queued for line *O*. (b) Finishing times for the five packets.

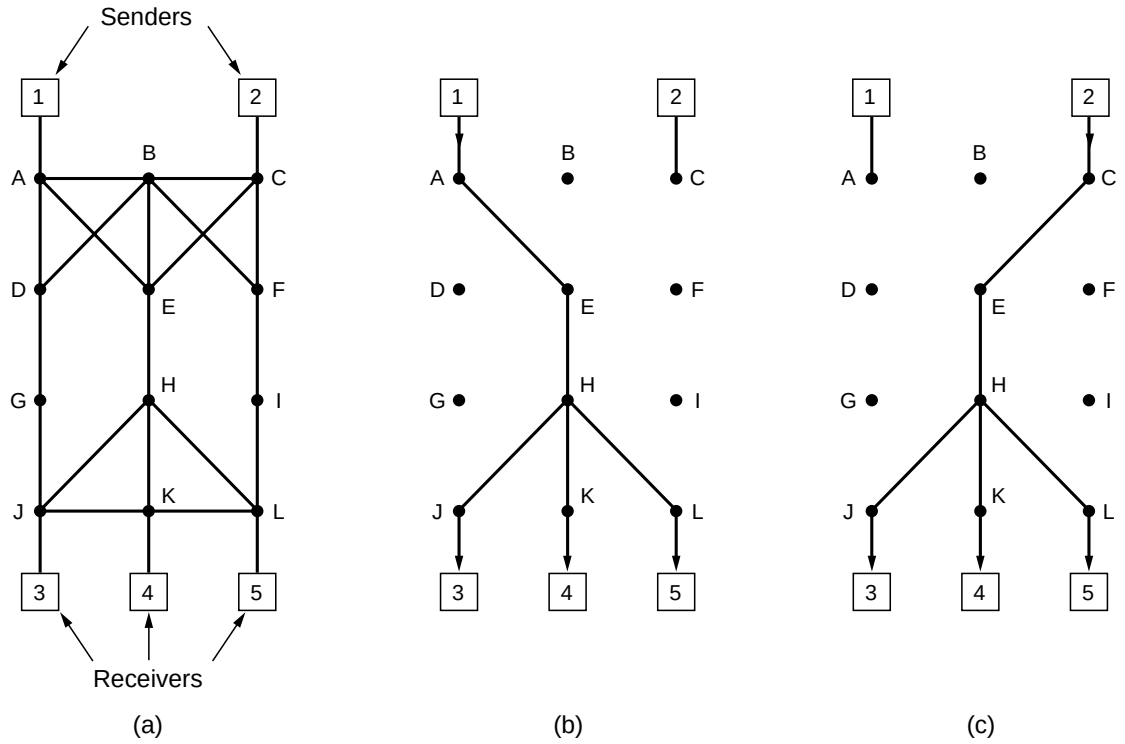


Fig. 5-37. (a) A network. (b) The multicast spanning tree for host 1. (c) The multicast spanning tree for host 2.

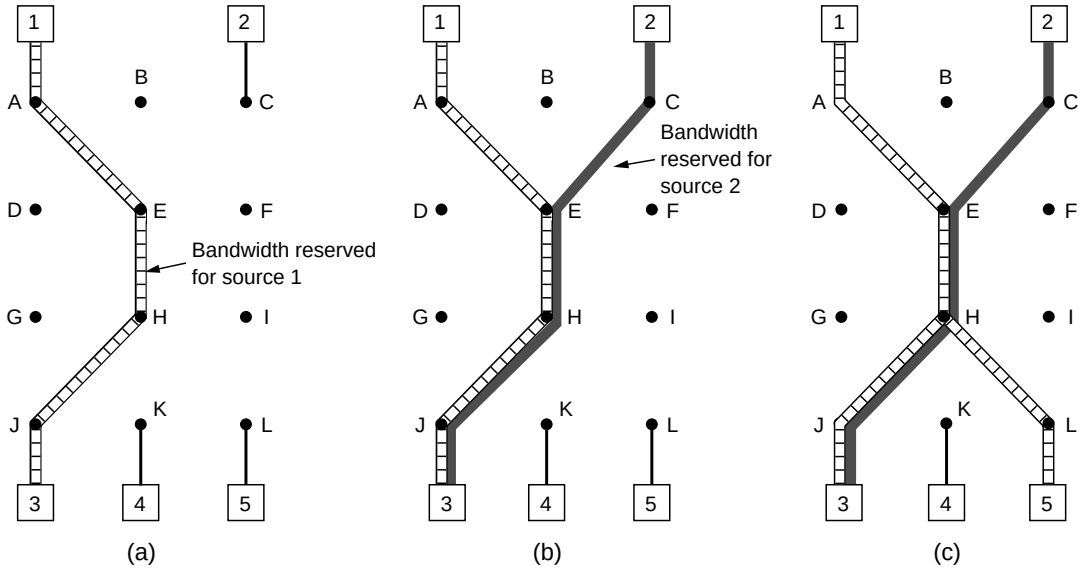


Fig. 5-38. (a) Host 3 requests a channel to host 1. (b) Host 3 then requests a second channel, to host 2. (c) Host 5 requests a channel to host 1.

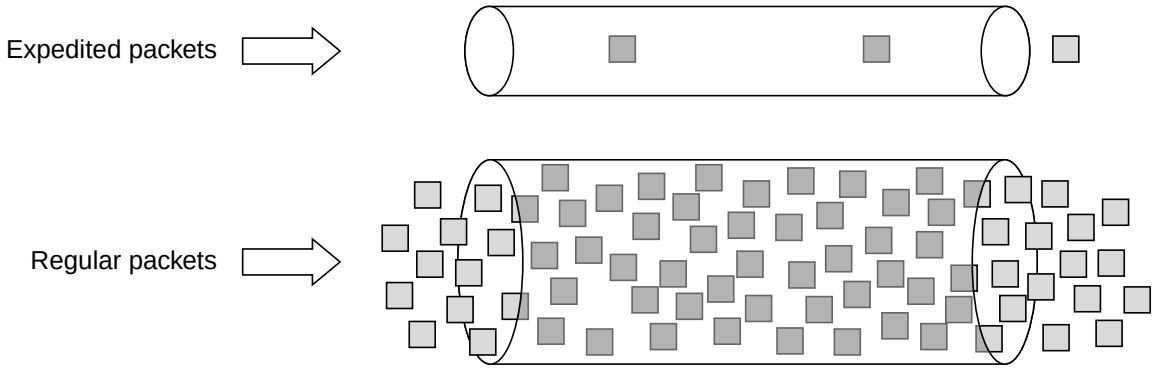


Fig. 5-39. Expedited packets experience a traffic-free network.

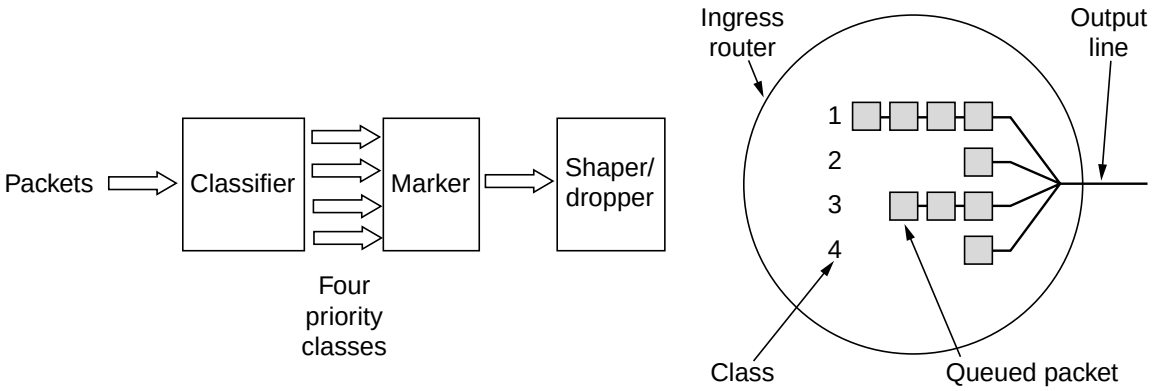


Fig. 5-40. A possible implementation of the data flow for assured forwarding.

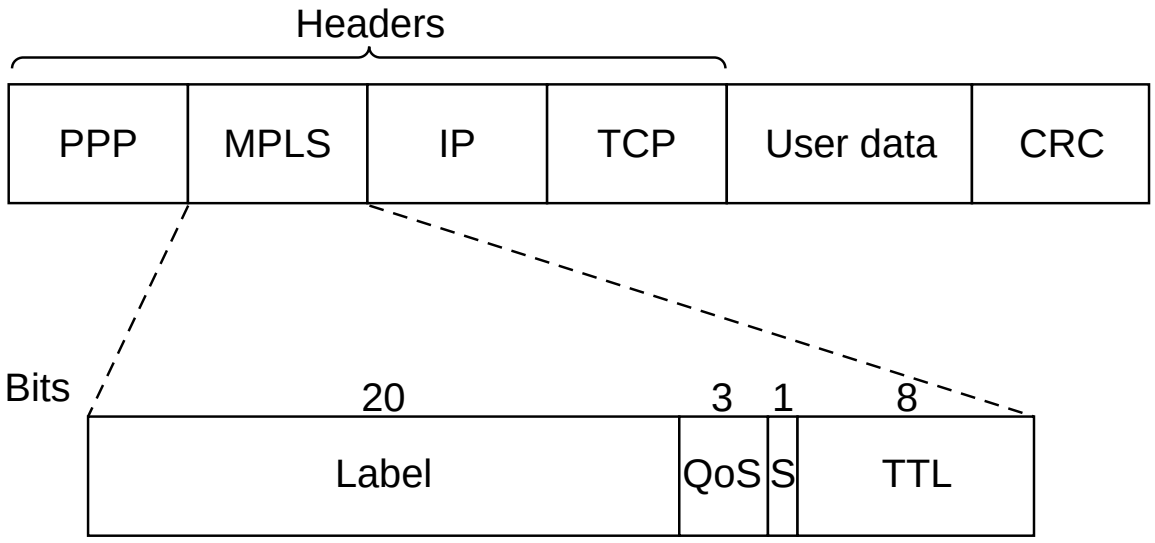


Fig. 5-41. Transmitting a TCP segment using IP, MPLS, and PPP.

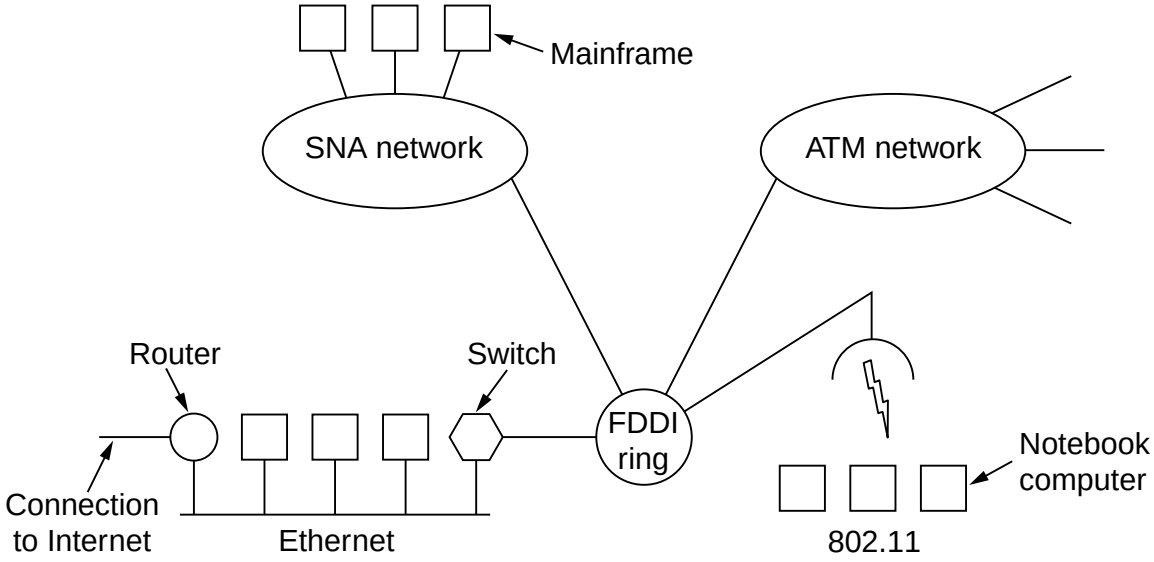


Fig. 5-42. A collection of interconnected networks.

Item	Some Possibilities
Service offered	Connection oriented versus connectionless
Protocols	IP, IPX, SNA, ATM, MPLS, AppleTalk, etc.
Addressing	Flat (802) versus hierarchical (IP)
Multicasting	Present or absent (also broadcasting)
Packet size	Every network has its own maximum
Quality of service	Present or absent; many different kinds
Error handling	Reliable, ordered, and unordered delivery
Flow control	Sliding window, rate control, other, or none
Congestion control	Leaky bucket, token bucket, RED, choke packets, etc.
Security	Privacy rules, encryption, etc.
Parameters	Different timeouts, flow specifications, etc.
Accounting	By connect time, by packet, by byte, or not at all

Fig. 5-43. Some of the many ways networks can differ.

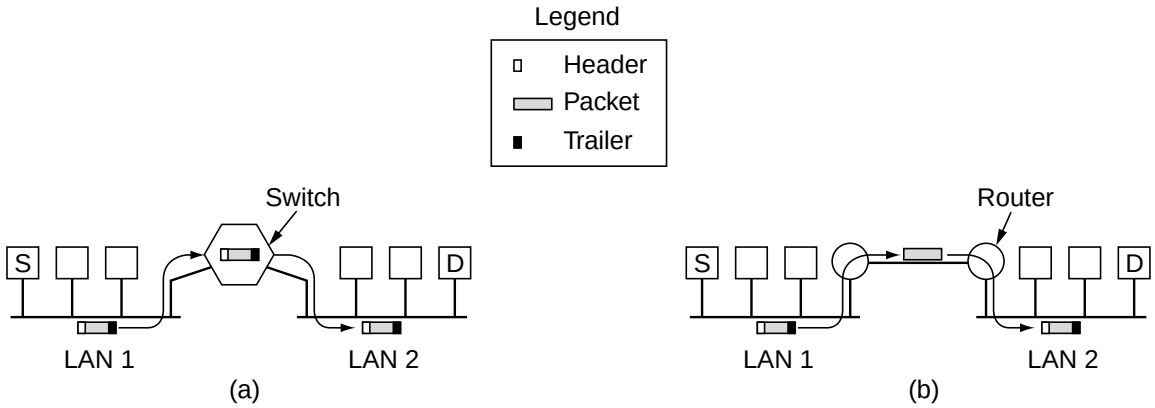


Fig. 5-44. (a) Two Ethernets connected by a switch. (b) Two Ethernets connected by routers.

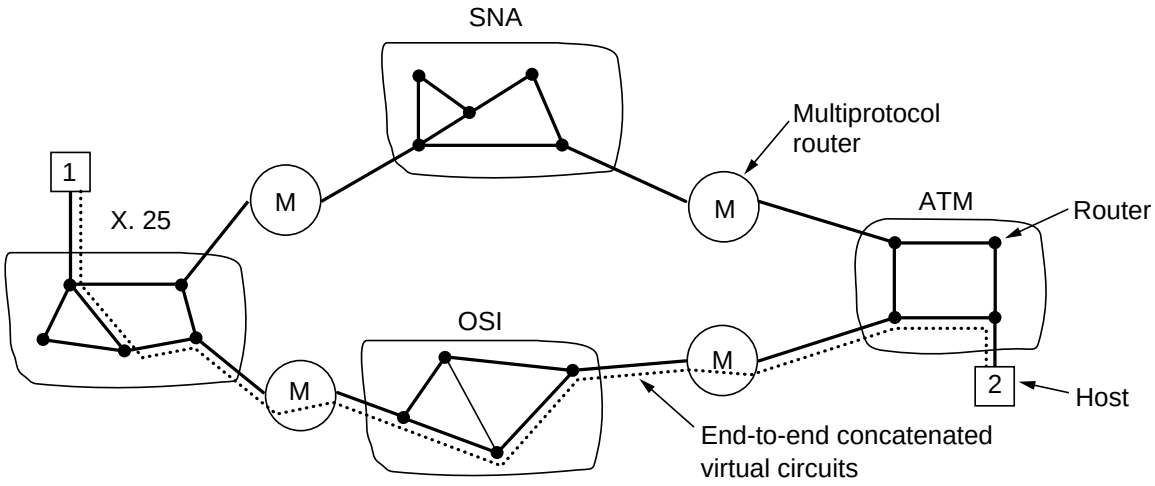


Fig. 5-45. Internetworking using concatenated virtual circuits.

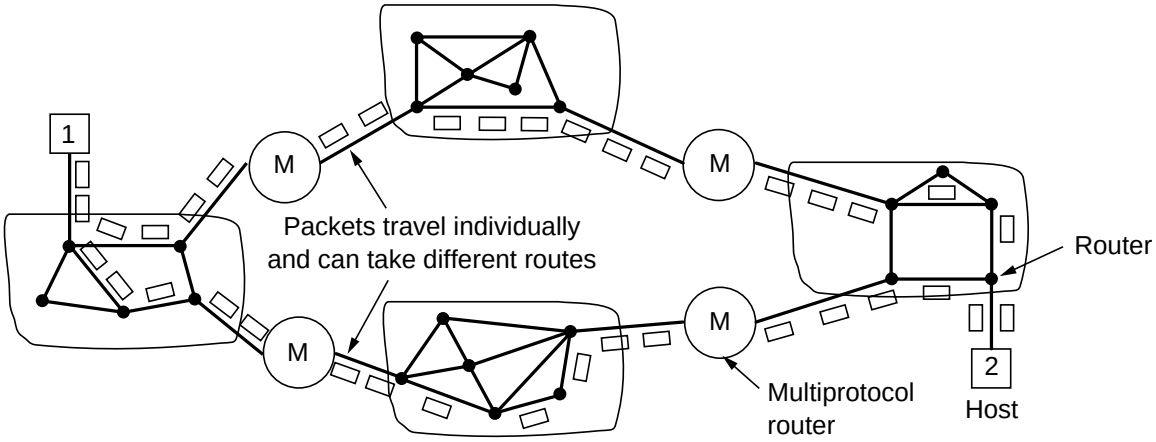


Fig. 5-46. A connectionless internet.

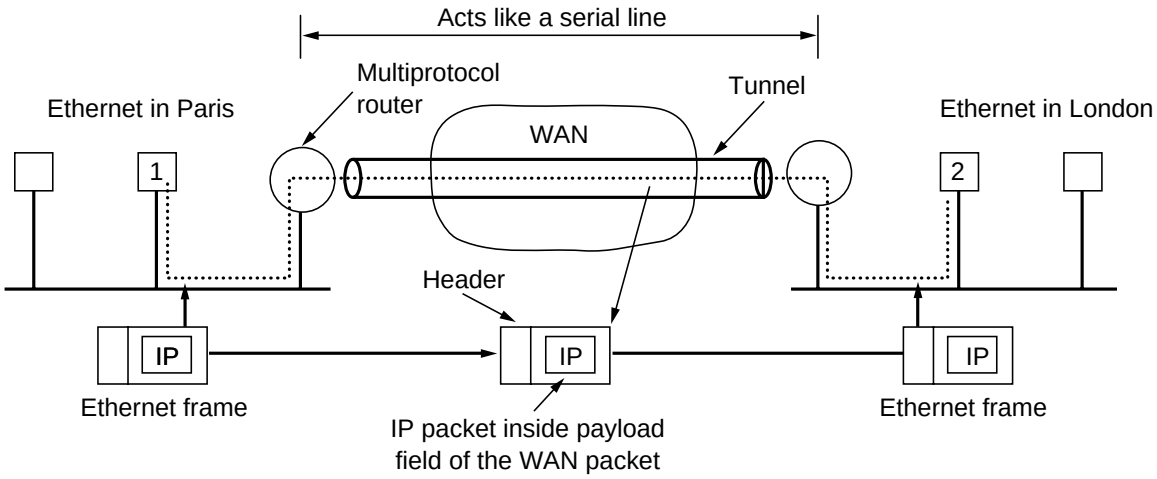


Fig. 5-47. Tunneling a packet from Paris to London.

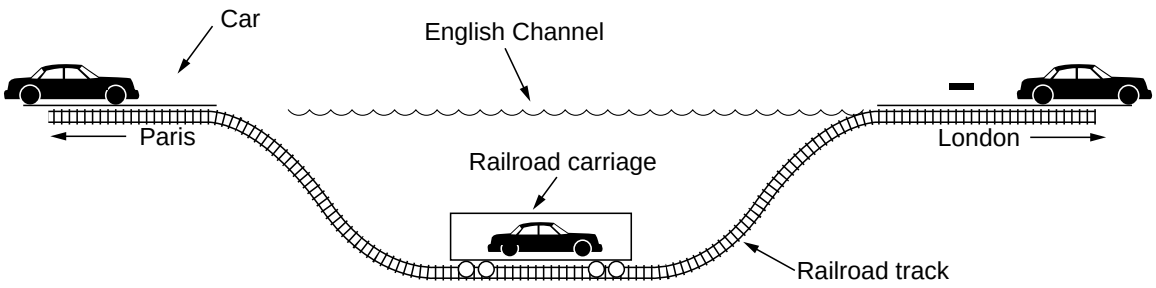


Fig. 5-48. Tunneling a car from France to England.

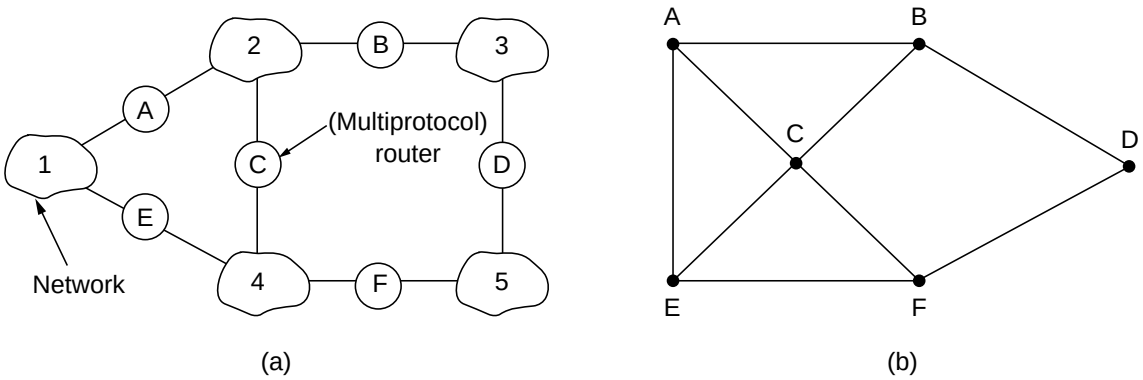


Fig. 5-49. (a) An internetwork. (b) A graph of the internetwork.

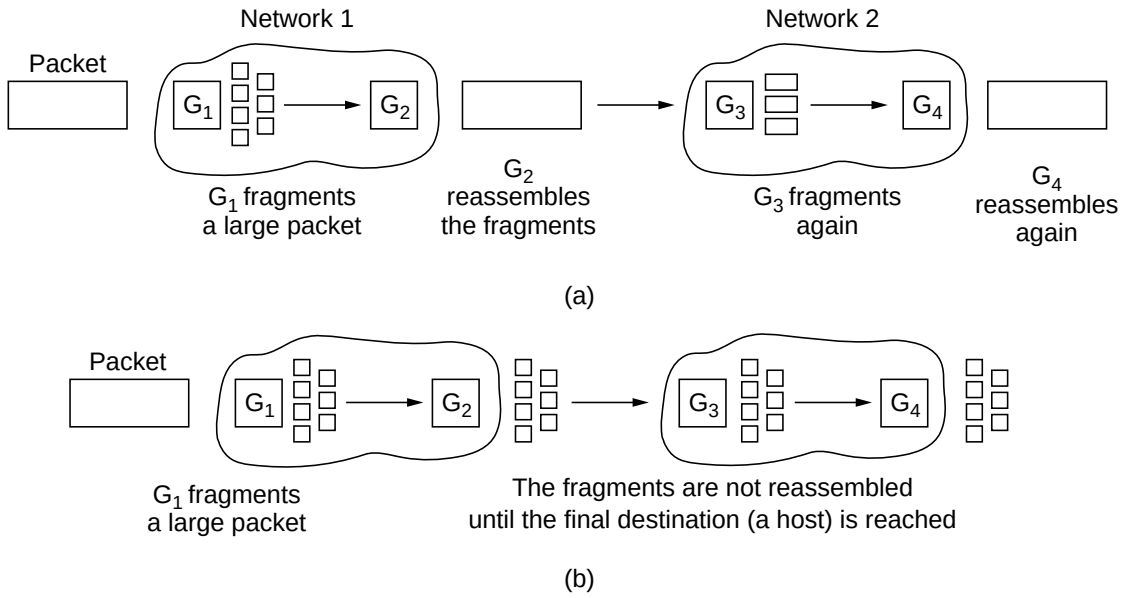


Fig. 5-50. (a) Transparent fragmentation. (b) Nontransparent fragmentation.

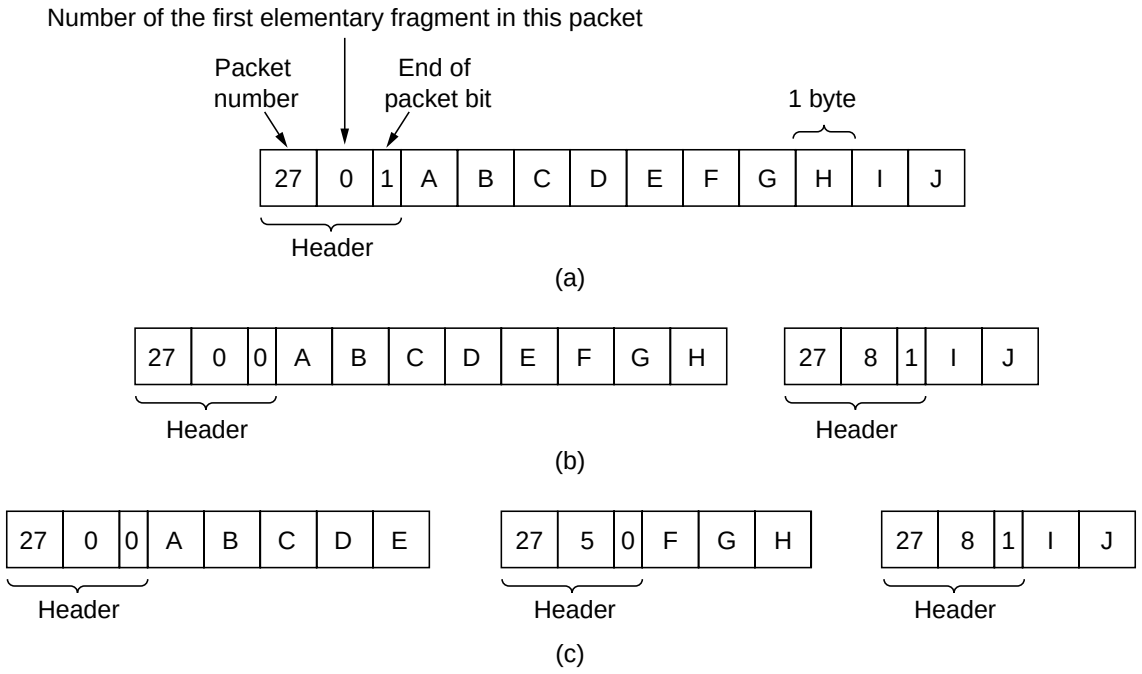


Fig. 5-51. Fragmentation when the elementary data size is 1 byte. (a) Original packet, containing 10 data bytes. (b) Fragments after passing through a network with maximum packet size of 8 payload bytes plus header. (c) Fragments after passing through a size 5 gateway.

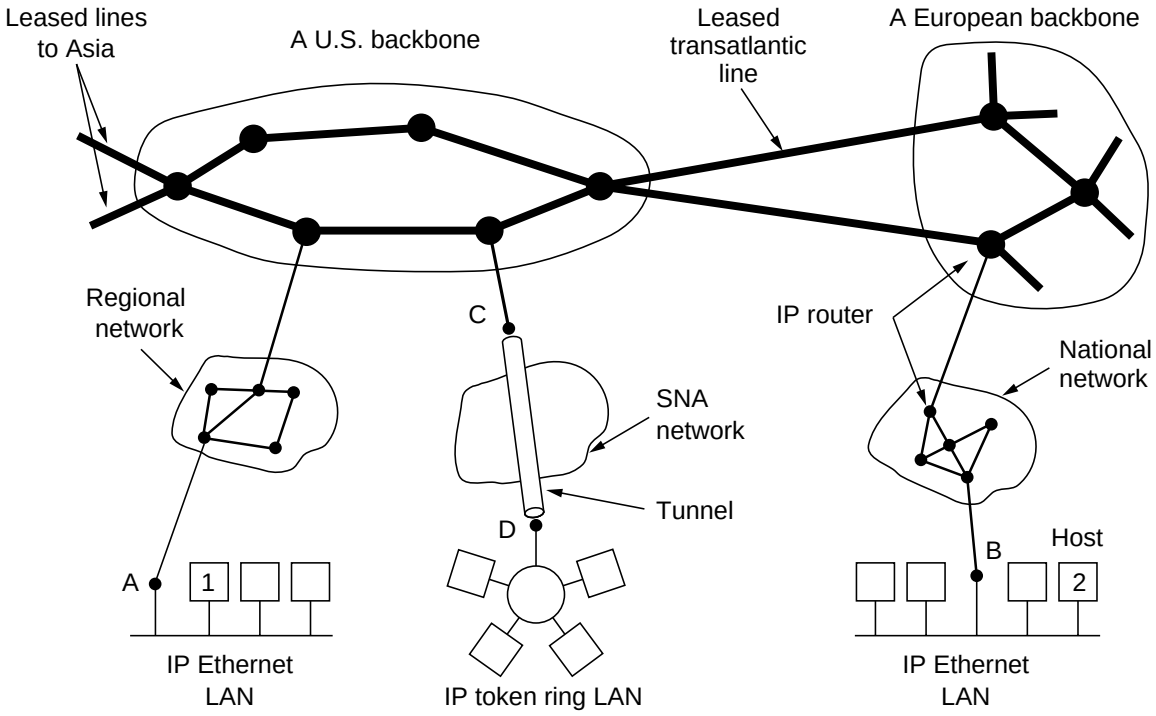


Fig. 5-52. The Internet is an interconnected collection of many networks.

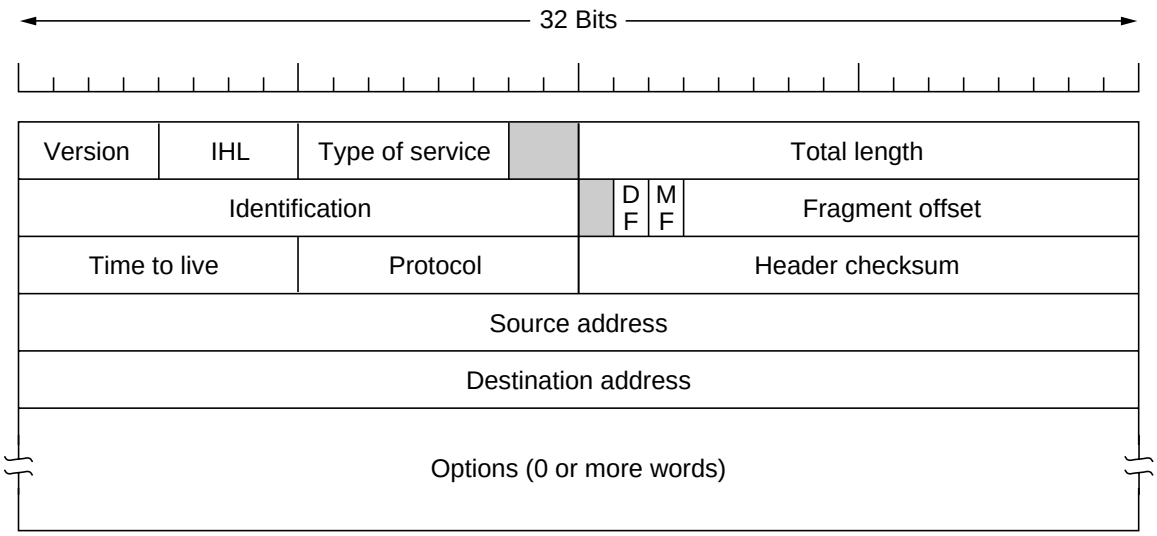


Fig. 5-53. The IPv4 (Internet Protocol) header.

Option	Description
Security	Specifies how secret the datagram is
Strict source routing	Gives the complete path to be followed
Loose source routing	Gives a list of routers not to be missed
Record route	Makes each router append its IP address
Timestamp	Makes each router append its address and timestamp

Fig. 5-54. Some of the IP options.

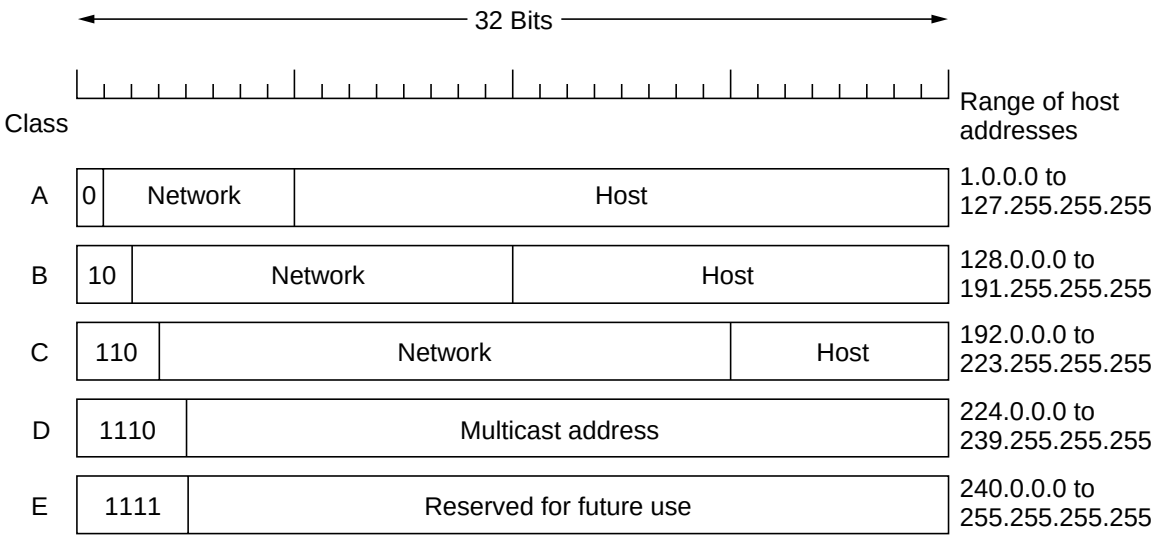


Fig. 5-55. IP address formats.

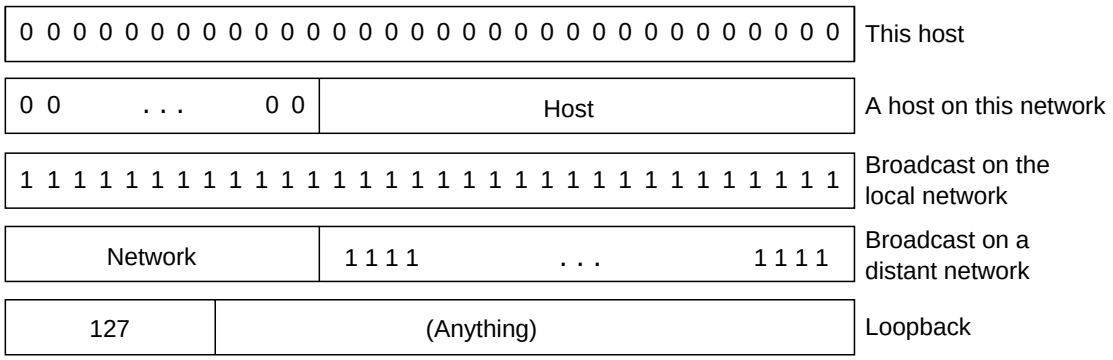


Fig. 5-56. Special IP addresses.

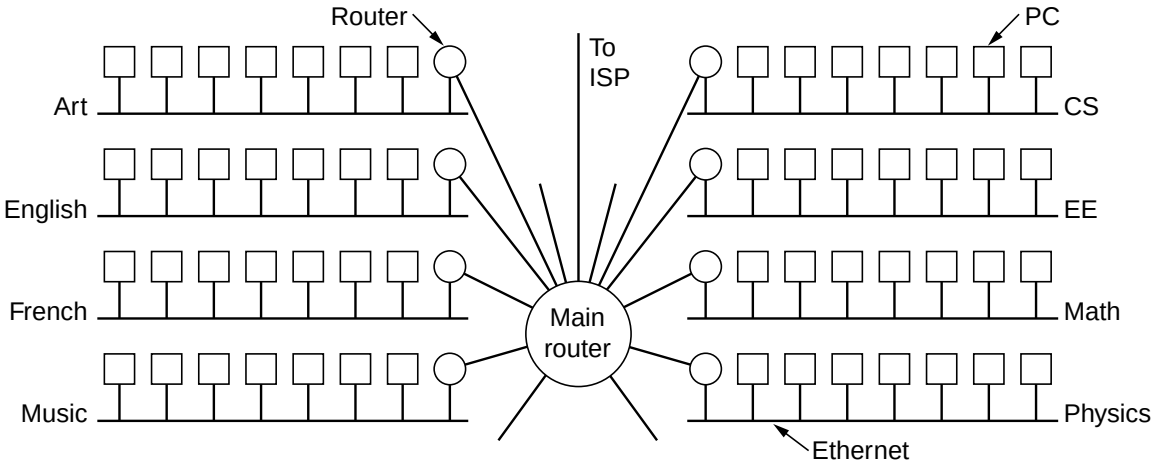


Fig. 5-57. A campus network consisting of LANs for various departments.

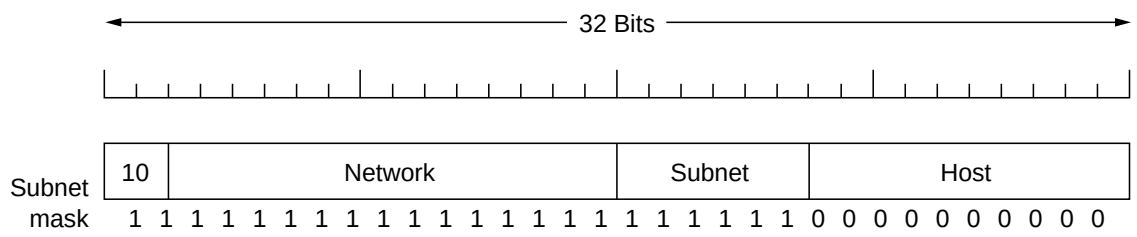


Fig. 5-58. A class B network subnetted into 64 subnets.

University	First address	Last address	How many	Written as
Cambridge	194.24.0.0	194.24.7.255	2048	194.24.0.0/21
Edinburgh	194.24.8.0	194.24.11.255	1024	194.24.8.0/22
(Available)	194.24.12.0	194.24.15.255	1024	194.24.12/22
Oxford	194.24.16.0	194.24.31.255	4096	194.24.16.0/20

Fig. 5-59. A set of IP address assignments.

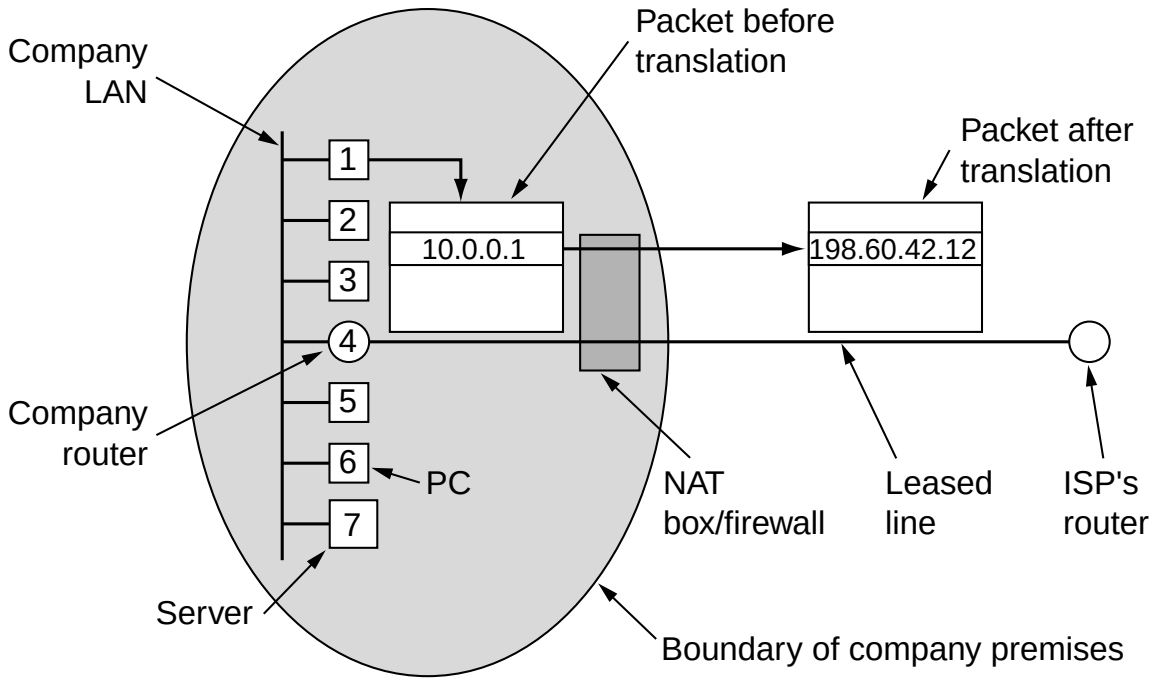


Fig. 5-60. Placement and operation of a NAT box.

Message type	Description
Destination unreachable	Packet could not be delivered
Time exceeded	Time to live field hit 0
Parameter problem	Invalid header field
Source quench	Choke packet
Redirect	Teach a router about geography
Echo	Ask a machine if it is alive
Echo reply	Yes, I am alive
Timestamp request	Same as Echo request, but with timestamp
Timestamp reply	Same as Echo reply, but with timestamp

Fig. 5-61. The principal ICMP message types.

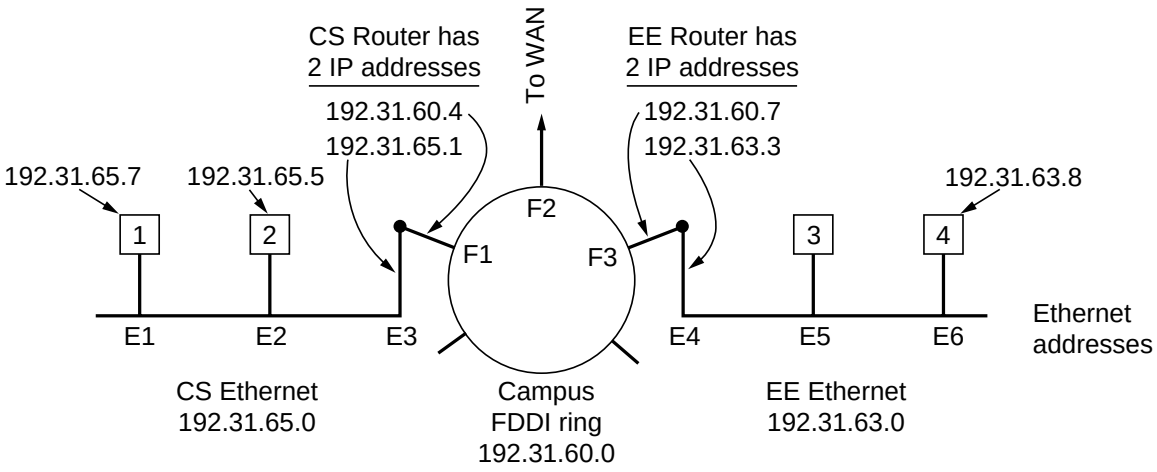


Fig. 5-62. Three interconnected /24 networks: two Ethernets and an FDDI ring.

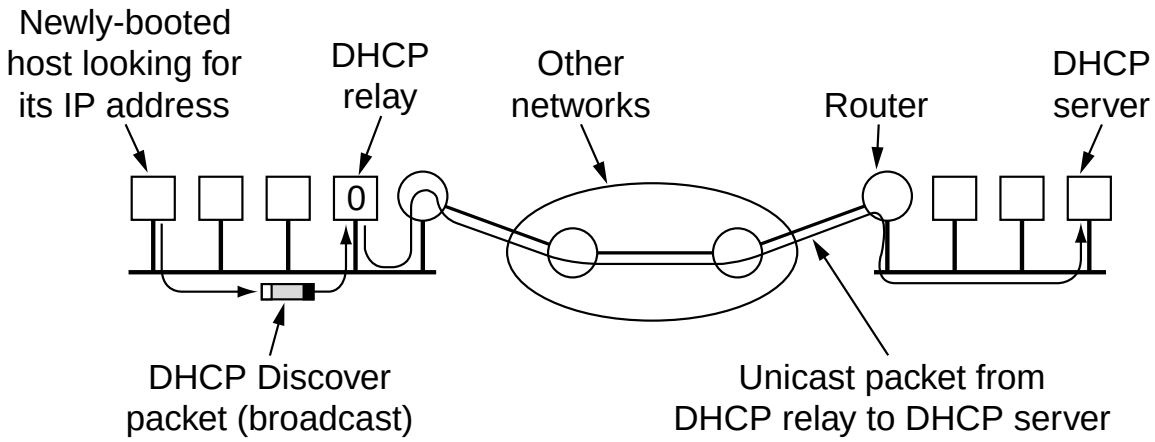


Fig. 5-63. Operation of DHCP.

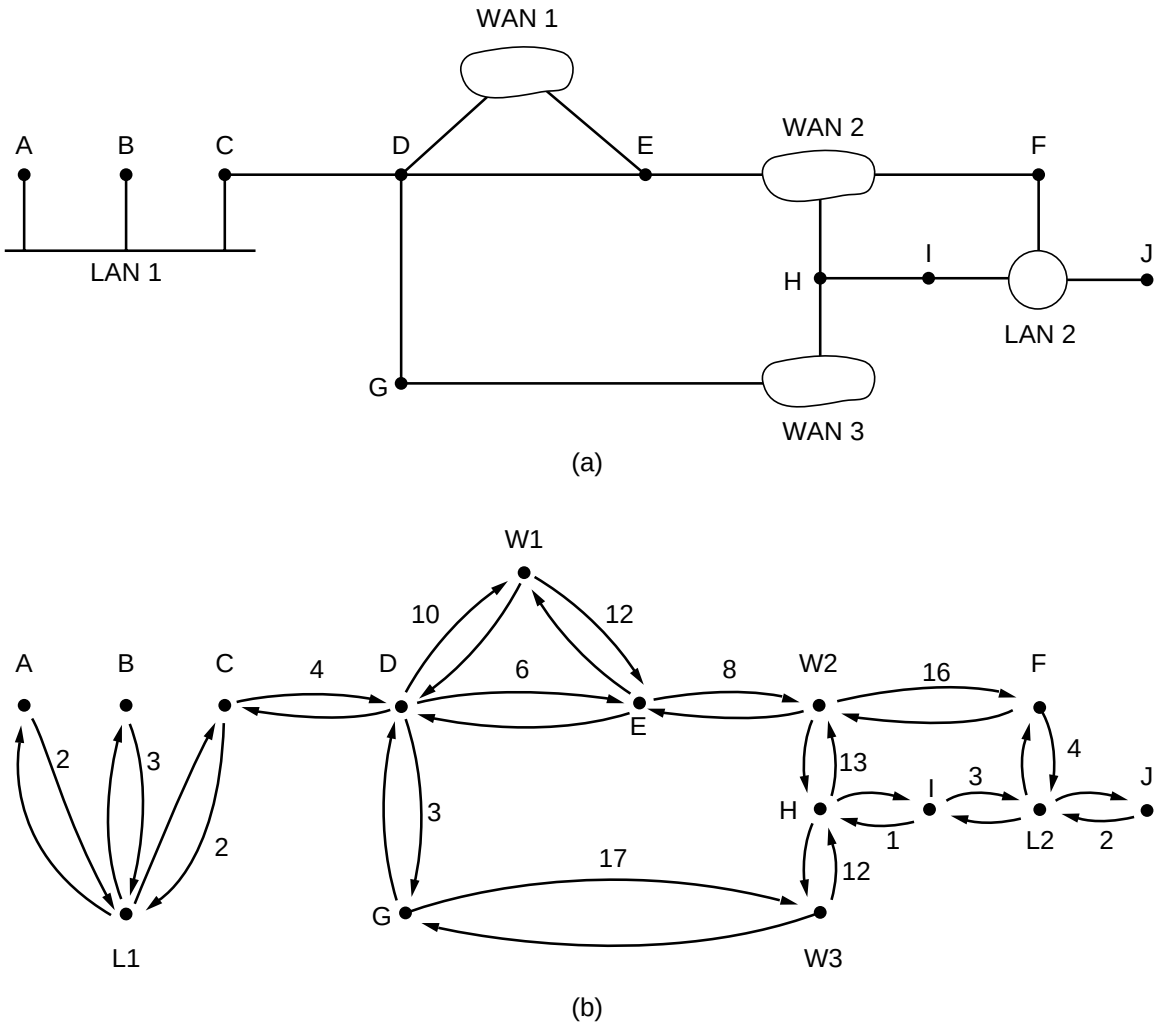


Fig. 5-64. (a) An autonomous system. (b) A graph representation of (a).

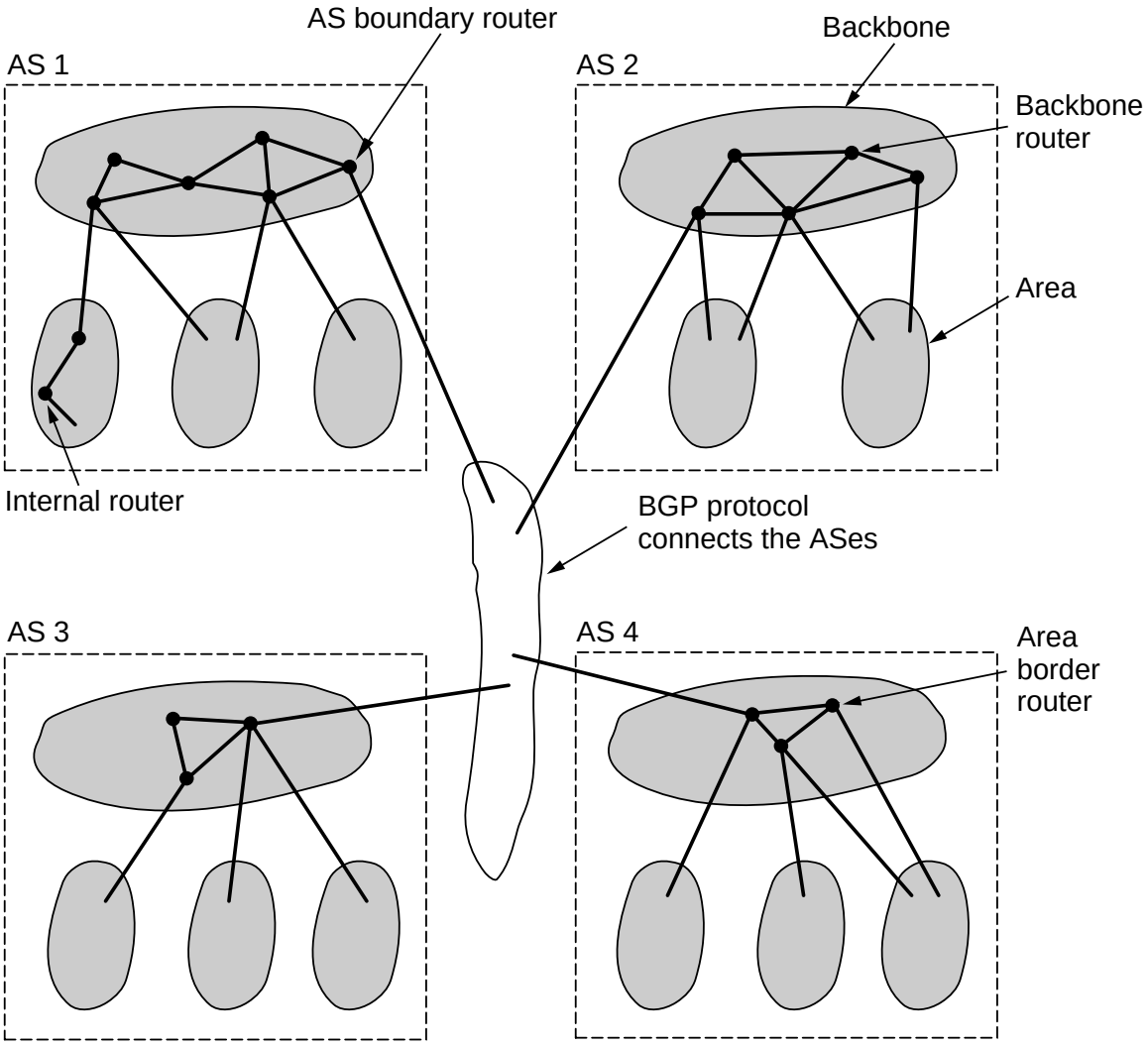
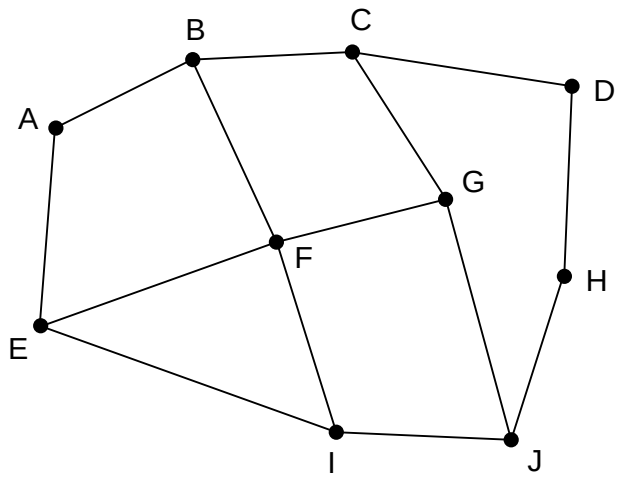


Fig. 5-65. The relation between ASes, backbones, and areas in OSPF.

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

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



(a)

Information F receives
from its neighbors about D

From B: "I use BCD"
From G: "I use GCD"
From I: "I use IFGCD"
From E: "I use EFGCD"

(b)

Fig. 5-67. (a) A set of BGP routers. (b) Information sent to *F*.

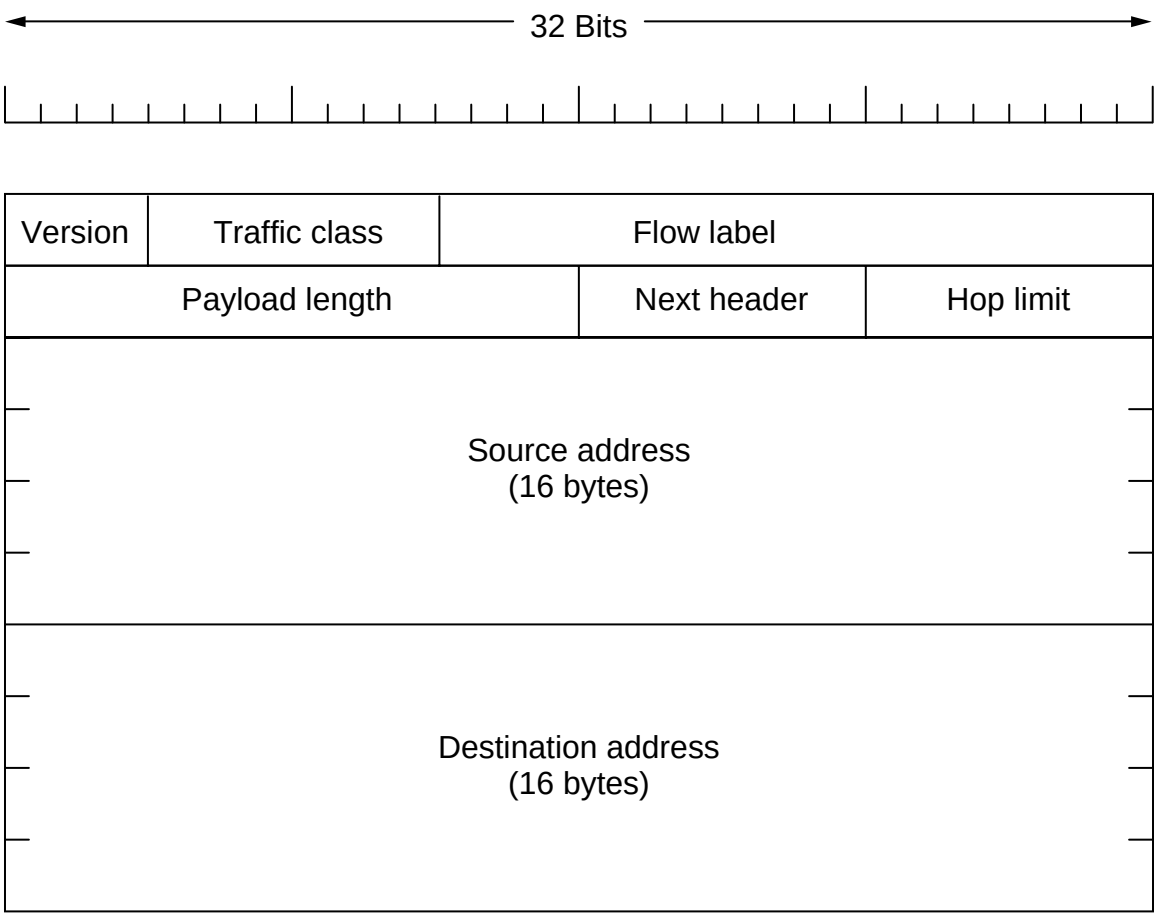


Fig. 5-68. The IPv6 fixed header (required).

Extension header	Description
Hop-by-hop options	Miscellaneous information for routers
Destination options	Additional information for the destination
Routing	Loose list of routers to visit
Fragmentation	Management of datagram fragments
Authentication	Verification of the sender’s identity
Encrypted security payload	Information about the encrypted contents

Fig. 5-69. IPv6 extension headers.

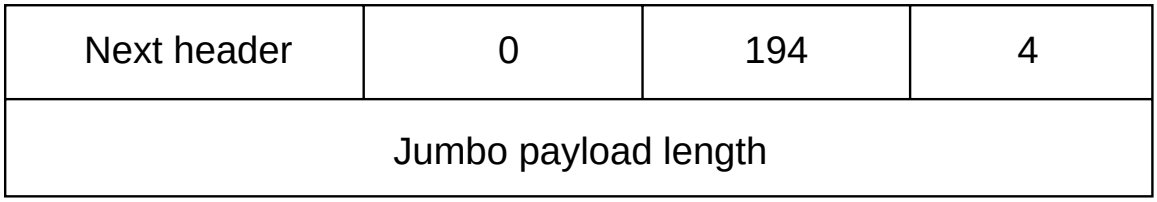


Fig. 5-70. The hop-by-hop extension header for large datagrams (jumbograms).

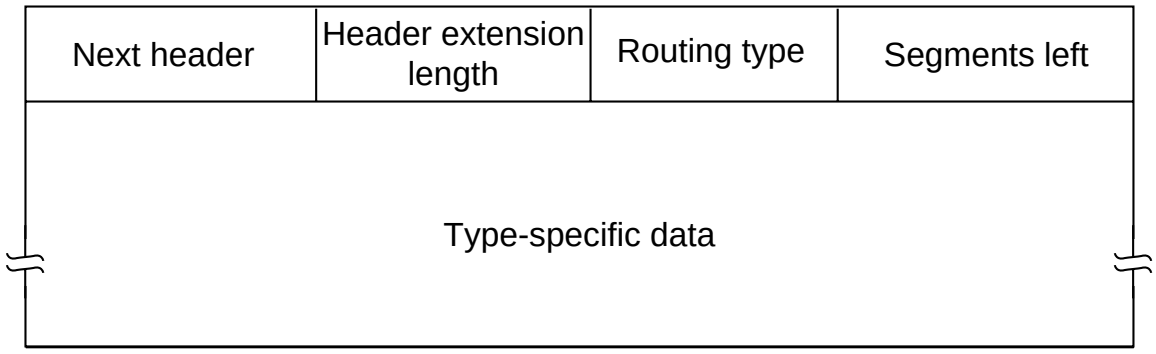


Fig. 5-71. The extension header for routing.