Chapter 4 Problems

Problem 1

a) With a connection-oriented network, every router failure will involve the routing of that connection. At a minimum, this will require the router that is "upstream" from the failed router to establish a new downstream part of the path to the destination node, with all of the requisite signaling involved in setting up a path. Moreover, all of the routers on the initial path that are downstream from the failed node must take down the failed connection, with all of the requisite signaling involved to do this.

   With a connectionless datagram network, no signaling is required to either set up a new downstream path or take down the old downstream path. We have seen, however, that routing tables will need to be updated (e.g., either via a distance vector algorithm or a link state algorithm) to take the failed router into account. We have seen that with distance vector algorithms, this routing table change can sometimes be localized to the area near the failed router. Thus, a datagram network would be preferable. Interestingly, the design criteria that the initial ARPAnet be able to function under stressful conditions was one of the reasons that datagram architecture was chosen for this Internet ancestor.

b) In order for a router to maintain an available fixed amount of capacity on the path between the source and destination node for that source-destination pair, it would need to know the characteristics of the traffic from all sessions passing through that link. That is, the router must have per-session state in the router. This is possible in a connection-oriented network, but not with a connectionless network. Thus, a connection-oriented VC network would be preferable.

c) In this scenario, datagram architecture has more control traffic overhead. This is due to the various packet headers needed to route the datagrams through the network. But in VC architecture, once all circuits are set up, they will never change. Thus, the signaling overhead is negligible over the long run.

Problem 2

a) Maximum number of VCs over a link = 28 = 256.
b) The centralized node could pick any VC number which is free from the set {0,1,…,28-1}. In this manner, it is not possible that there are fewer VCs in progress than 256 without there being any common free VC number.
c) Each of the links can independently allocate VC numbers from the set {0,1,…,28-1}. Thus, a VC will likely have a different VC number for each link along its path. Each router in the VC's path must replace the VC number of each arriving packet with the VC number associated with the outbound link.

Problem 3

For a VC forwarding table, the columns are : Incoming Interface, Incoming VC Number, Outgoing Interface, Outgoing VC Number. For a datagram forwarding table, the columns are: Destination Address, Outgoing Interface.

Problem 4

a) Data destined to host H3 is forwarded through interface 3

Destination Address	Link Interface
H3	3

b) No, because forwarding rule is only based on destination address.

c) One possible configuration is:

Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	12	3	22
2	63	4	18

Note, that the two flows could actually have the same VC numbers.

d) One possible configuration is:

Router B.

Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	22	2	24

Router C.

Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	18	2	50

Router D.

Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	24	3	70
2	50	**3**	**76**

Problem 5

a) No VC number can be assigned to the new VC; thus the new VC cannot be established in the network.
b) Each link has two available VC numbers. There are four links. So the number of combinations is 24 = 16. One example combination is (10,00,00,10).

Problem 6

In a virtual circuit network, there *is* an end-to-end connection in the sense that each router along the path must maintain state for the connection; hence the terminology *connection service*. In a connection-oriented transport service over a connectionless network layer, such as TCP over IP, the end systems maintain connection state; however the routers have no notion of any connections; hence the terminology *connection-oriented service*.

Problem 7

a) No, you can only transmit one packet at a time over a shared bus.

b) Yes, as discussed in the text, as long as the two packets use different input busses and different output busses, they can be forwarded in parallel.

c) No, in this case the two packets would have to be sent over the same output bus at the same time, which is not possible.

Problem 8

a) (n-1)D
b) (n-1)D
c) 0

Problem 9

The minimal number of time slots needed is 3. The scheduling is as follows.
Slot 1: send X in top input queue, send Y in middle input queue.
Slot 2: send X in middle input queue, send Y in bottom input queue
Slot 3: send Z in bottom input queue.

Largest number of slots is still 3. Actually, based on the assumption that a non-empty input queue is never idle, we see that the first time slot always consists of sending X in the top input queue and Y in either middle or bottom input queue, and in the second time slot, we can always send two more datagram, and the last datagram can be sent in third time slot.

NOTE: Actually, if the first datagram in the bottom input queue is X, then the worst case would require 4 time slots.

Problem 10

a)

Prefix Match	**Link Interface**
11100000 00	0

11100000 01000000	1
1110000	2
11100001 1	3
otherwise	3

b) Prefix match for first address is 5th entry: link interface 3
Prefix match for second address is 3nd entry: link interface 2
Prefix match for third address is 4th entry: link interface 3

Problem 11

Destination Address Range	**Link Interface**
00000000 through 00111111	0
01000000 through 01011111	1
01100000 through 01111111	2
10000000 through 10111111	2
11000000 through 11111111	3

number of addresses for interface 0 = $2^6 = 64$
number of addresses for interface 1 = $2^5 = 32$
number of addresses for interface 2 = $2^6 + 2^5 = 64 + 32 = 96$
number of addresses for interface 3 = $2^6 = 64$

Problem 12

Destination Address Range **Link Interface**

11000000

through (32 addresses) 11011111	0
10000000 through(64 addresses) 10111111	1
11100000 through (32 addresses) 11111111	2
00000000 through (128 addresses) 01111111	3

Problem 13

223.1.17.0/26
223.1.17.128/25
223.1.17.192/28

Problem 14

Destination Address	**Link Interface**
200.23.16/21	0
200.23.24/24	1
200.23.24/21	2
otherwise	3

Problem 15

Destination Address	**Link Interface**
11100000 00 (224.0/10)	0
11100000 01000000 (224.64/16)	1
1110000 (224/8)	2
11100001 1 (225.128/9)	3
otherwise	3

Problem 16

Any IP address in range 128.119.40.128 to 128.119.40.191

Four equal size subnets: 128.119.40.64/28, 128.119.40.80/28, 128.119.40.96/28, 128.119.40.112/28

Problem 17

From 214.97.254/23, possible assignments are

a) Subnet A: 214.97.255/24 (256 addresses)
Subnet B: 214.97.254.0/25 - 214.97.254.0/29 (128-8 = 120 addresses)
Subnet C: 214.97.254.128/25 (128 addresses)

Subnet D: 214.97.254.0/31 (2 addresses)
Subnet E: 214.97.254.2/31 (2 addresses)
Subnet F: 214.97.254.4/30 (4 addresses)

b) To simplify the solution, assume that no datagrams have router interfaces as ultimate destinations. Also, label D, E, F for the upper-right, bottom, and upper-left interior subnets, respectively.

<u>Router 1</u>

Longest Prefix Match	**Outgoing Interface**
11010110 01100001 11111111	Subnet A
11010110 01100001 11111110 0000000	Subnet D
11010110 01100001 11111110 000001	Subnet F

<u>Router 2</u>

Longest Prefix Match	**Outgoing Interface**
11010110 01100001 11111111 0000000	Subnet D
11010110 01100001 11111110 0	Subnet B
11010110 01100001 11111110 0000001	Subnet E

<u>Router 3</u>

Longest Prefix Match	**Outgoing Interface**
11010110 01100001 11111111 000001	Subnet F
11010110 01100001 11111110 0000001	Subnet E
11010110 01100001 11111110 1	Subnet C

Problem 18

The IP address blocks of Polytechnic Institute of New York University are:

NetRange: 128.238.0.0 - 128.238.255.255
CIDR: 128.238.0.0/16

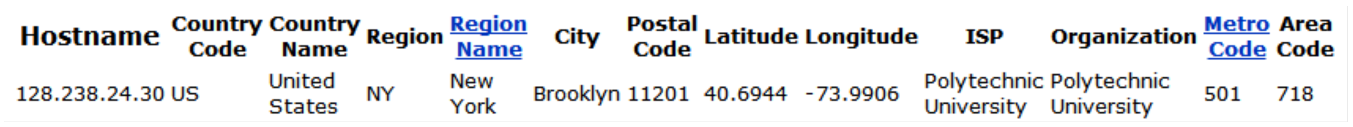
The IP address blocks Stanford University are:
NetRange: 171.64.0.0 - 171.67.255.255
CIDR: 171.64.0.0/14

The IP address blocks University of Washington are:
NetRange: 140.142.0.0 - 140.142.255.255
CIDR: 140.142.0.0/16

No, the whois services cannot be used to determine with certainty the geographical location of a specific IP address.

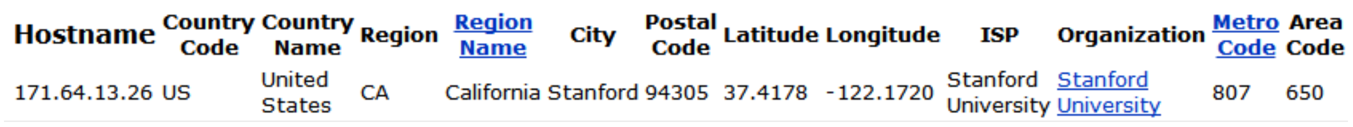
www.maxmind.com is used to determine the locations of the Web servers at Polytechnic Institute of New York University, Stanford University and University of Washington.

Locations of the Web server at Polytechnic Institute of New York University is



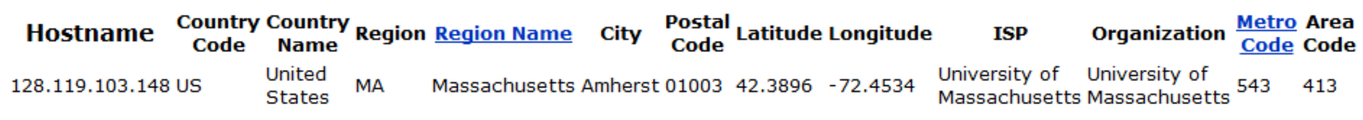
Hostname	Country Code	Country Name	Region	Region Name	City	Postal Code	Latitude	Longitude	ISP	Organization	Metro Code	Area Code
128.238.24.30	US	United States	NY	New York	Brooklyn	11201	40.6944	-73.9906	Polytechnic University	Polytechnic University	501	718

Locations of the Web server Stanford University is



Hostname	Country Code	Country Name	Region	Region Name	City	Postal Code	Latitude	Longitude	ISP	Organization	Metro Code	Area Code
171.64.13.26	US	United States	CA	California	Stanford	94305	37.4178	-122.1720	Stanford University	Stanford University	807	650

Locations of the Web server at University of Massachusetts is



Hostname	Country Code	Country Name	Region	Region Name	City	Postal Code	Latitude	Longitude	ISP	Organization	Metro Code	Area Code
128.119.103.148	US	United States	MA	Massachusetts	Amherst	01003	42.3896	-72.4534	University of Massachusetts	University of Massachusetts	543	413

Problem 19

The maximum size of data field in each fragment = 680 (because there are 20 bytes IP header).

Thus the number of required fragments $= \left\lceil \frac{2400-20}{680} \right\rceil = 4$

Each fragment will have Identification number 422. Each fragment except the last one will be of size 700 bytes (including IP header). The last datagram will be of size 360 bytes (including IP header). The offsets of the 4 fragments will be 0, 85, 170, 255. Each of the first 3 fragments will have flag=1; the last fragment will have flag=0.

Problem 20

MP3 file size = 5 million bytes. Assume the data is carried in TCP segments, with each TCP segment also having 20 bytes of header. Then each datagram can carry 1500-40=1460 bytes of the MP3 file

Number of datagrams required $= \left\lceil \frac{5\times 10^6}{1460} \right\rceil = 3425$. All but the last datagram will be 1,500 bytes; the last datagram will be 960+40 = 1000 bytes. Note that here there is no fragmentation – the source host does not create datagrams larger than 1500 bytes, and these datagrams are smaller than the MTUs of the links.

Problem 21

a) Home addresses: 192.168.1.1, 192.168.1.2, 192.168.1.3 with the router interface being 192.168.1.4

b)

NAT Translation Table

WAN Side	LAN Side
24.34.112.235, 4000	192.168.1.1, 3345
24.34.112.235, 4001	192.168.1.1, 3346
24.34.112.235, 4002	192.168.1.2, 3445
24.34.112.235, 4003	192.168.1.2, 3446
24.34.112.235, 4004	192.168.1.3, 3545
24.34.112.235, 4005	192.168.1.3, 3546

Problem 22

a) Since all IP packets are sent outside, so we can use a packet sniffer to record all IP packets generated by the hosts behind a NAT. As each host generates a sequence of IP packets with sequential numbers and a distinct (very likely, as they are randomly chosen from a large space) initial identification number (ID), we can group IP packets with consecutive IDs into a cluster. The number of clusters is the number of hosts behind the NAT.

For more practical algorithms, see the following papers.

"A Technique for Counting NATted Hosts", by Steven M. Bellovin, appeared in IMW'02, Nov. 6-8, 2002, Marseille, France.
"Exploiting the IPID field to infer network path and end-system characteristics."

Weifeng Chen, Yong Huang, Bruno F. Ribeiro, Kyoungwon Suh, Honggang Zhang, Edmundo de Souza e Silva, Jim Kurose, and Don Towsley.
PAM'05 Workshop, March 31 - April 01, 2005. Boston, MA, USA.

b) However, if those identification numbers are not sequentially assigned but randomly assigned, the technique suggested in part (a) won't work, as there won't be clusters in sniffed data.

Problem 23

It is not possible to devise such a technique. In order to establish a direct TCP connection between Arnold and Bernard, either Arnold or Bob must initiate a connection to the other. But the NATs covering Arnold and Bob drop SYN packets arriving from the WAN side. Thus neither Arnold nor Bob can initiate a TCP connection to the other if they are both behind NATs.

Problem 24

y-x-u, y-x-v-u, y-x-w-u, y-x-w-v-u,
y-w-u, y-w-v-u, y-w-x-u, y-w-x-v-u, y-w-v-x-u,
y-z-w-u, y-z-w-v-u, y-z-w-x-u, y-z-w-x-v-u, y-z-w-v-x-u,

Problem 25

x to z:
x-y-z, x-y-w-z,
x-w-z, x-w-y-z,
x-v-w-z, x-v-w-y-z,
x-u-w-z, x-u-w-y-z,
x-u-v-w-z, x-u-v-w-y-z

z to u:
z-w-u,
z-w-v-u, z-w-x-u, z-w-v-x-u, z-w-x-v-u, z-w-y-x-u, z-w-y-x-v-u,
z-y-x-u, z-y-x-v-u, z-y-x-w-u, z-y-x-w-y-u, z-y-x-v-w-u,
z-y-w-v-u, z-y-w-x-u, z-y-w-v-x-u, z-y-w-x-v-u, z-y-w-y-x-u, z-y-w-y-x-v-u

z to w:
z-w, z-y-w, z-y-x-w, z-y-x-v-w, z-y-x-u-w, z-y-x-u-v-w, z-y-x-v-u-w

Problem 26

Step	*N'*	*D(t),p(t)*	*D(u),p(u)*	*D(v),p(v)*	*D(w),p(w)*	*D(y),p(y)*	*D(z),p(z)*
0	x	∞	∞	3,x	6,x	6,x	8,x

1	xv	7,v	6,v	3,x	6,x	6,x	8,x
2	xvu	7,v	6,v	3,x	6,x	6,x	8,x
3	xvuw	7,v	6,v	3,x	6,x	6,x	8,x
4	xvuwy	7,v	6,v	3,x	6,x	6,x	8,x
5	xvuwyt	7,v	6,v	3,x	6,x	6,x	8,x
6	xvuwytz	7,v	6,v	3,x	6,x	6,x	8,x

Problem 27

a)

Step	*N'*	*D(x), p(x)*	*D(u),p(u)*	*D(v),p(v)*	*D(w),p(w)*	*D(y),p(y)*	*D(z),p(z)*
0	t	∞	2,t	4,t	∞	7,t	∞
1	tu	∞	2,t	4,t	5,u	7,t	∞
2	tuv	7,v	2,t	4,t	5,u	7,t	∞
3	tuvw	7,v	2,t	4,t	5,u	7,t	∞
4	tuvwx	7,v	2,t	4,t	5,u	7,t	15,x
5	tuvwxy	7,v	2,t	4,t	5,u	7,t	15,x
6	tuvwxyz	7,v	2,t	4,t	5,u	7,t	15,x

b)

Step	*N'*	*D(x), p(x)*	*D(t),p(t)*	*D(v),p(v)*	*D(w),p(w)*	*D(y),p(y)*	*D(z),p(z)*
	u	∞	2,u	3,u	3,u	∞	∞
	ut	∞	2,u	3,u	3,u	9,t	∞
	utv	6,v	2,u	3,u	3,u	9,t	∞
	utvw	6,v	2,u	3,u	3,u	9,t	∞
	utvwx	6,v	2,u	3,u	3,u	9,t	14,x
	utvwxy	6,v	2,u	3,u	3,u	9,t	14,x
	utvwxyz	6,v	2,u	3,u	3,u	9,t	14,x

c)

Step	*N'*	*D(x), p(x)*	*D(u),p(u)*	*D(t),pt)*	*D(w),p(w)*	*D(y),p(y)*	*D(z),p(z)*
	v	3,v	3,v	4,v	4,v	8,v	∞
	vx	3,v	3,v	4,v	4,v	8,v	11,x
	vxu	3,v	3,v	4,v	4,v	8,v	11,x
	vxut	3,v	3,v	4,v	4,v	8,v	11,x
	vxutw	3,v	3,v	4,v	4,v	8,v	11,x

	vxutwy	3,v	3,v	4,v	4,v	8,v	11,x
	vxutwyz	3,v	3,v	4,v	4,v	8,v	11,x

d)

Step	*N'*	*D(x), p(x)*	*D(u),p(u)*	*D(v),p(v)*	*D(t),p(t)*	*D(y),p(y)*	*D(z),p(z)*
	w	6,w	3,w	4,w	∞	∞	∞
	wu	6,w	3,w	4,w	5,u	∞	∞
	wuv	6,w	3,w	4,w	5,u	12,v	∞
	wuvt	6,w	3,w	4,w	5,u	12,v	∞
	wuvtx	6,w	3,w	4,w	5,u	12,v	14,x
	wuvtxy	6,w	3,w	4,w	5,u	12,v	14,x
	wuvtxyz	6,w	3,w	4,w	5,u	12,v	14,x

e)

Step	*N'*	*D(x), p(x)*	*D(u),p(u)*	*D(v),p(v)*	*D(w),p(w)*	*D(t),p(t)*	*D(z),p(z)*
	y	6,y	∞	8,y	∞	7,y	12,y
	yx	6,y	∞	8,y	12,x	7,y	12,y
	yxt	6,y	9,t	8,y	12,x	7,y	12,y
	yxtv	6,y	9,t	8,y	12,x	7,y	12,y
	yxtvu	6,y	**9,t**	8,y	12,x	7,y	12,y
	yxtvuw	6,y	9,t	8,y	12,x	7,y	12,y
	yxtvuwz	6,y	9,t	8,y	12,x	7,y	12,y

f)

Step	*N'*	*D(x), p(x)*	*D(u),p(u)*	*D(v),p(v)*	*D(w),p(w)*	*D(y),p(y)*	*D(t),p(t)*
	z	8,z	∞	∞	∞	12,z	∞
	zx	8,z	∞	11,x	14,x	12,z	∞
	zxv	8,z	14,v	**11,x**	14,x	12,z	15,v
	zxvy	8,z	14,v	11,x	14,x	**12,z**	15,v
	zxvyu	8,z	**14,v**	11,x	14,x	12,z	15,v
	zxvyuw	8,z	14,v	11,x	**14,x**	12,z	15,v
	zxvyuwt	8,z	14,v	11,x	**14,x**	12,z	15,v

Problem 28

Cost to

u v x y z

	v	∞	∞	∞	∞	∞
From	x	∞	∞	∞	∞	∞
	z	∞	6	2	∞	0

Cost to

		u	v	x	y	z
	v	1	0	3	∞	6
From	x	∞	3	0	3	2
	z	7	5	2	5	0

Cost to

		u	v	x	y	z
	v	1	0	3	3	5
From	x	4	3	0	3	2
	z	6	5	2	5	0

Cost to

		u	v	x	y	z
	v	1	0	3	3	5
From	x	4	3	0	3	2
	z	6	5	2	5	0

Problem 29

The wording of this question was a bit ambiguous. We meant this to mean, “the number of iterations from when the algorithm is run for the first time” (that is, assuming the only information the nodes initially have is the cost to their nearest neighbors). We assume that the algorithm runs synchronously (that is, in one step, all nodes compute their distance tables at the same time and then exchange tables).

At each iteration, a node exchanges distance tables with its neighbors. Thus, if you are node A, and your neighbor is B, all of B's neighbors (which will all be one or two hops from you) will know the shortest cost path of one or two hops to you after one iteration (i.e., after B tells them its cost to you).

Let d be the "diameter" of the network - the length of the longest path without loops between any two nodes in the network. Using the reasoning above, after $d-1$ iterations, all nodes will know the shortest path cost of d or fewer hops to all other nodes. Since any path with greater than d hops will have loops (and thus have a greater cost than that path with the loops removed), the algorithm will converge in at most $d-1$ iterations.

ASIDE: if the DV algorithm is run as a result of a change in link costs, there is no a priori bound on the number of iterations required until convergence unless one also specifies a bound on link costs.

Problem 30

a) Dx(w) = 2, Dx(y) = 4, Dx(u) = 7

b) First consider what happens if c(x,y) changes. If c(x,y) becomes larger or smaller (as long as c(x,y) >=1) , the least cost path from x to u will still have cost at least 7. Thus a change in c(x,y) (if c(x,y)>=1) will not cause x to inform its neighbors of any changes.
If c(x,y)= δ<1, then the least cost path now passes through y and has cost δ+6.

Now consider if c(x,w) changes. If c(x,w) = $\varepsilon \le 1$, then the least-cost path to u continues to pass through w and its cost changes to 5 + ε; x will inform its neighbors of this new cost. If c(x,w) = $\delta > 6$, then the least cost path now passes through y and has cost 11; again x will inform its neighbors of this new cost.

c) Any change in link cost c(x,y) (and as long as c(x,y) >=1) will not cause x to inform its neighbors of a new minimum-cost path to u .

Problem 31

Node x table

Cost to

		x	y	z
	x	0	3	4
From	y	∞	∞	∞
	z	∞	∞	∞

Cost to

		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Node y table

		Cost to		
		x	y	z
	x	∞	∞	∞
From	y	3	0	6
	z	∞	∞	∞

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Node z table

		Cost to		
		x	y	z
	x	∞	∞	∞
From	y	∞	∞	∞
	z	4	6	0

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Problem 32

NO, this is because that decreasing link cost won't cause a loop (caused by the next-hop relation of between two nodes of that link). Connecting two nodes with a link is equivalent to decreasing the link weight from infinite to the finite weight.

Problem 33

At each step, each updating of a node's distance vectors is based on the Bellman-Ford equation, i.e., only decreasing those values in its distance vector. There is no increasing in values. If no updating, then no message will be sent out. Thus, D(x) is non-increasing. Since those costs are finite, then eventually distance vectors will be stabilized in finite steps.

Problem 34

a)

Router z	Informs w, $D_z(x)=\infty$
	Informs y, $D_z(x)=6$

Router w	Informs y, $D_w(x)=\infty$
	Informs z, $D_w(x)=5$
Router y	Informs w, $D_y(x)=4$
	Informs z, $D_y(x)=4$

b) Yes, there will be a count-to-infinity problem. The following table shows the routing converging process. Assume that at time t0, link cost change happens. At time t1, y updates its distance vector and informs neighbors w and z. In the following table, "→" stands for "informs".

time	t0	t1	t2	t3	t4
Z	→ w, $D_z(x)=\infty$ → y, $D_z(x)=6$		No change	→ w, $D_z(x)=\infty$ → y, $D_z(x)=11$	
W	→ y, $D_w(x)=\infty$ → z, $D_w(x)=5$		→ y, $D_w(x)=\infty$ → z, $D_w(x)=10$		No change
Y	→ w, $D_y(x)=4$ → z, $D_y(x)=4$	→ w, $D_y(x)=9$ → z, $D_y(x)=\infty$		No change	→ w, $D_y(x)=14$ → z, $D_y(x)=\infty$

We see that w, y, z form a loop in their computation of the costs to router x. If we continue the iterations shown in the above table, then we will see that, at t27, z detects that its least cost to x is 50, via its direct link with x. At t29, w learns its least cost to x is 51 via z. At t30, y updates its least cost to x to be 52 (via w). Finally, at time t31, no updating, and the routing is stabilized.

time	t27	t28	t29	t30	t31
Z	→ w, $D_z(x)=50$ → y, $D_z(x)=50$				via w, ∞ via y, 55 via z, 50
W		→ y, $D_w(x)=\infty$ → z, $D_w(x)=50$	→ y, $D_w(x)=51$ → z, $D_w(x)=\infty$		via w, ∞ via y, ∞ via z, 51
Y		→ w, $D_y(x)=53$ → z, $D_y(x)=\infty$		→ w, $D_y(x)=\infty$ → z, $D_y(x)=52$	via w, 52 via y, 60 via z, 53

c) cut the link between y and z.

Problem 35

Since full AS path information is available from an AS to a destination in BGP, loop detection is simple – if a BGP peer receives a route that contains its own AS number in the AS path, then using that route would result in a loop.

Problem 36

The chosen path is not necessarily the shortest AS-path. Recall that there are many issues to be considered in the route selection process. It is very likely that a longer loop-free path is preferred over a shorter loop-free path due to economic reason. For example, an AS might prefer to send traffic to one neighbor instead of another neighbor with shorter AS distance.

Problem 37

a) eBGP

b) iBGP

c) eBGP

d) iBGP

Problem 38

a) I1 because this interface begins the least cost path from 1d towards the gateway router 1c.

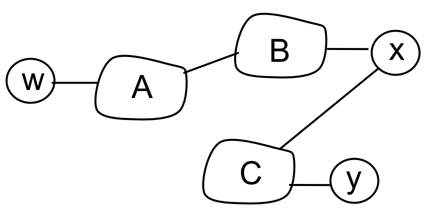
b) I2. Both routes have equal AS-PATH length but I2 begins the path that has the closest NEXT-HOP router.

c) I1. I1 begins the path that has the shortest AS-PATH.

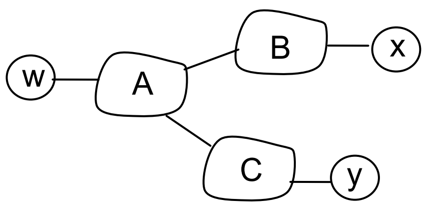
Problem 39

One way for C to force B to hand over all of B's traffic to D on the east coast is for C to only advertise its route to D via its east coast peering point with C.

Problem 40



X's view of the topology



W's view of the topology

In the above solution, X does not know about the AC link since X does not receive an advertised route to w or to y that contain the AC link (i.e., X receives no advertisement containing both AS A and AS C on the path to a destination.

Problem 41

BitTorrent file sharing and Skype P2P applications.
Consider a BitTorrent file sharing network in which peer 1, 2, and 3 are in stub networks W, X, and Y respectively. Due the mechanism of BitTorrent's file sharing, it is quire possible that peer 2 gets data chunks from peer 1 and then forwards those data chunks to 3. This is equivalent to B forwarding data that is finally destined to stub network Y.

Problem 42

A should advise to B two routes, AS-paths A-W and A-V.
A should advise to C only one route, A-V.
C receives AS paths: B-A-W, B-A-V, A-V.

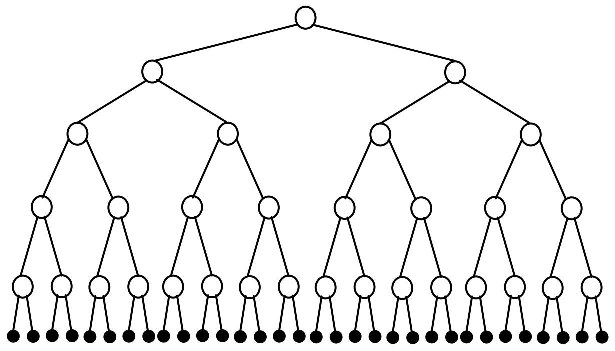
Problem 43

Since Z wants to transit Y's traffic, Z will send route advertisements to Y. In this manner, when Y has a datagram that is destined to an IP that can be reached through Z, Y will have the option of sending the datagram through Z. However, if Z advertizes routes to Y, Y can re-advertize those routes to X. Therefore, in this case, there is nothing Z can do from preventing traffic from X to transit through Z.

Problem 44

The minimal spanning tree has z connected to y via x at a cost of 14(=8+6).
z connected to v via x at a cost of 11(=8+3);
z connected to u via x and v, at a cost of 14(=8+3+3);
z connected to w via x, v, and u, at a cost of 17(=8+3+3+3).
This can be obtained by Prim's algorithm to grow a minimum spanning tree.

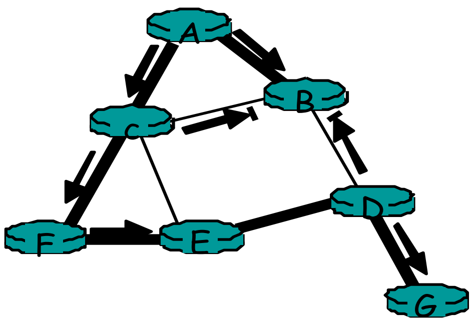
Problem 45



The 32 receives are shown connected to the sender in the binary tree configuration shown above. With network-layer broadcast, a copy of the message is forwarded over each link exactly once. There are thus 62 link crossings (2+4+8+16+32). With unicast emulation, the sender unicasts a copy to each receiver over a path with5 hops. There are thus 160 link crossings (5*32).

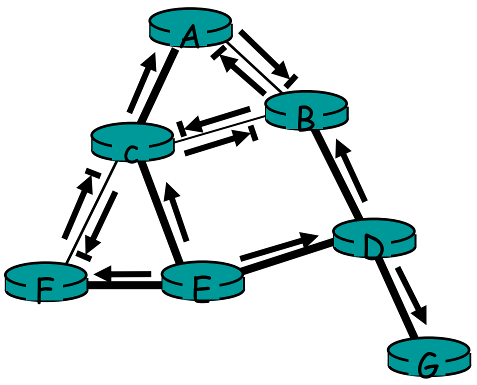
A topology in which all receivers are in a line, with the sender at one end of the line, will have the largest disparity between the cost of network-layer broadcast and unicast emulation.

Problem 46

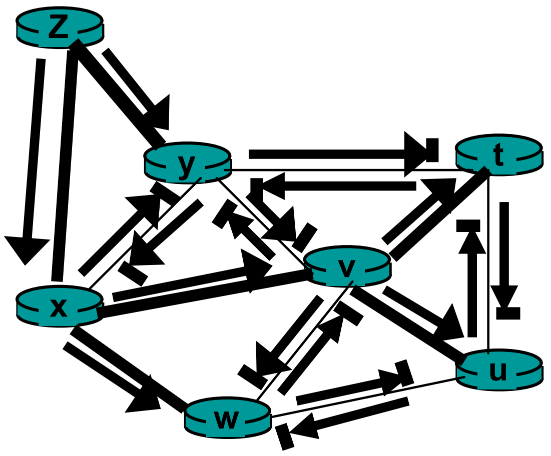


The thicker shaded lines represent The shortest path tree from A to all destination. Other solutions are possible, but in these solutions, B can not route to either C or D from A.

Problem 47



Problem 48



Problem 49

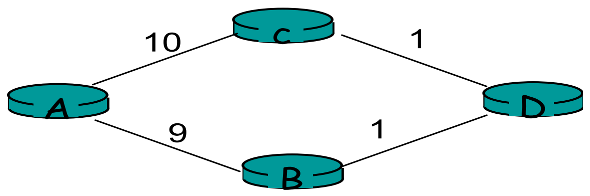
The center-based tree for the topology shown in the original figure connects A to C; B to C; E to C; and F to C (all directly). D connects to C via E, and G connects to C via D, E. This center-based tree is different from the minimal spanning tree shown in the figure.

Problem 50

The center-based tree for the topology shown in the original figure connects t to v; u to v; w to v; x to v; and y to v (all directly). And z connected to v via x. This center-based tree is different from the minimal spanning tree.

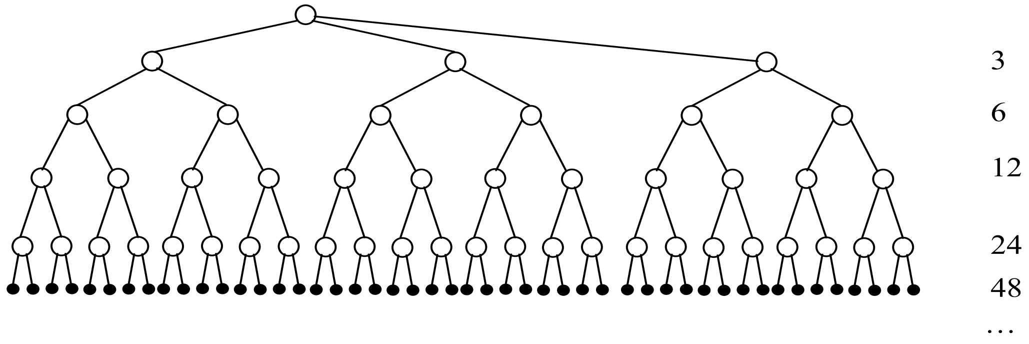
Problem 51

Dijkstra's algorithm for the network below, with node A as the source, results in a least-unicast-cost path tree of links AC, AB, and BD, with an overall free cost of 20. The minimum spanning tree contains links AB, BD, and DC, at a cost of 11.



Problem 52

After 1 step 3 copies are transmitted, after 2 steps 6 copies are transmitted. After 3 steps, 12 copies are transmitted, and so on. After k steps, $3*2^{k-1}$ copies will be transmitted in that step.



Problem 53

The protocol must be built at the application layer. For example, an application may periodically multicast its identity to all other group members in an application-layer message.

Problem 54

A simple application-layer protocol that will allow all members to know the identity of all other members in the group is for each instance of the application to send a multicast message containing its identity to all other members. This protocol sends message in-band, since the multicast channel is used to distribute the identification messages as well as multicast data from the application itself. The use of the in-band signaling makes use of the existing multicast distribution mechanism, leading to a very simple design.

Problem 55

$32-4=28$ bits are available for multicast addresses. Thus, the size of the multicast address space is $N=2^{28}$.

The probability that two groups choose the same address is

$$\frac{1}{N}=2^{-28}=3.73\cdot 10^{-9}$$

The probability that 1000 groups all have different addresses is

$$\frac{N \cdot (N-1) \cdot (N-2) \cdots (N-999)}{N^{1000}} = \left(1 - \frac{1}{N}\right)\left(1 - \frac{2}{N}\right) \cdots \left(1 - \frac{999}{N}\right)$$

Ignoring cross-product terms, this is approximately equal to

$$1 - \left(\frac{1 + 2 + \cdots + 999}{N}\right) = 1 - \frac{999 \cdot 1000}{2N} = 0.998$$