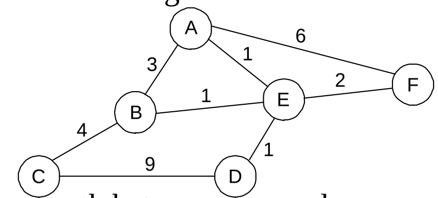
ELEN 602 Lecture 19

- Routing

Overview

- Forwarding vs Routing
  - forwarding: to select an output port based on destination address and routing table
  - routing: process by which routing table is built
- Network as a Graph



- Problem: Find lowest cost path between two nodes
- Factors
  - static: topology
  - dynamic: load

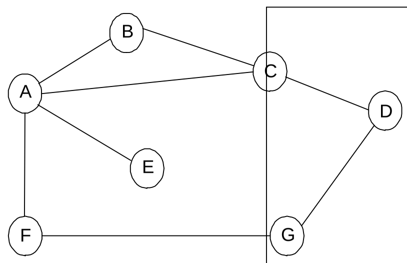
Routing Algorithms

- Distance Vector
  - Each node constructs a vector of distances to its neighbors and transmits it to neighbors
  - Use this information in building routing tables
- Link State
  - Disseminate link state information of neighbors to everyone
  - Construct routing tables based on the sum of information from all the nodes

Distance Vector

- Each node maintains a set of triples
  - **`(Destination, Cost, NextHop)`**
- Exchange updates with directly connected neighbors
  - periodically (on the order of several seconds)
  - whenever its table changes (called *triggered* update)
- Each update is a list of pairs:
  - **`(Destination, Cost)`**
- Update local table if receive a "better" route
  - smaller cost
  - came from next-hop
- Refresh existing routes; delete if they time out

Example



Destination	Cost	NextHop
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	A

Initial Distances(Global View)

Information at	Distance to Reach Node						
Node	A	B	C	D	E	F	G
A	0	1	1	Inf	1	1	Inf
B	1	0	1	Inf	Inf	Inf	Inf
C	1	1	0	1	Inf	Inf	Inf
D	Inf	Inf	1	0	Inf	Inf	1
E	1	Inf	Inf	Inf	0	Inf	Inf
F	1	Inf	Inf	Inf	Inf	0	1
G	Inf	Inf	Inf	1	Inf	1	0

Initial Routing Table at A

Destination	Cost	Nexthop
B	1	B
C	1	C
D	Inf	---
E	1	E
F	1	E
G	Inf.	----

Routing Table Updates

- Update an entry if a new distance vector shows a shorter route to destination.
- A shortest distance path to a destination consists of shortest distance paths between intermediate nodes.
  - D(I, J) = min (D(I, K) + D(K, J)), over all K
- Bellman-Ford or Ford-Fulkerson Algorithm
- Periodic Updates based on timers
- Triggered updates --based on noticed changes in distance vector
- Recompute distances based on updates.

Example Network

- When A hears from C, it finds a distance 2 path to D through C - update routing table entry for D.
- When A hears from D, it finds a distance 3 path to G through D -updates routing table entry for G.
- When A hears from F, it finds a distance 2 path to G through F -updates routing table entry for G.

Final Routing Table at A

Destination	Cost	Nexthop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	E
G	2	F

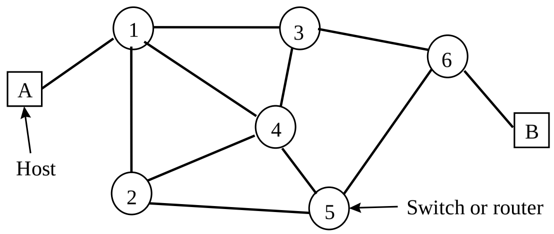
Final Distances(Global View)

Information at	Distance to Reach Node						
Node	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Routing Loops

- Example 1
  - F detects that link to G has failed
  - F sets distance to G to infinity and sends update t o A
  - A sets distance to G to infinity since it uses F to reach G
  - A receives periodic update from C with 2-hop path to G
  - A sets distance to G to 3 and sends update to F
  - F decides it can reach G in 4 hops via A
- Example 2
  - link from A to E fails
  - A advertises distance of infinity to E
  - B and C advertise a distance of 2 to E
  - B decides it can reach E in 3 hops through C; advertises this to A
  - A decides it can read E in 4 hops; advertises this to C
  - C decides that it can reach E in 5 hops…

Example Network



Routing Tables for Example Network

Node 1

Destination	Next node
2	2
3	3
4	4
5	2
6	3

Node 2

Destination	Next node
1	1
3	1
4	4
5	5
6	5

Node 3

Destination	Next node
1	1
2	4
4	4
5	6
6	6

Node 4

Destination	Next node
1	1
2	2
3	3
5	5
6	3

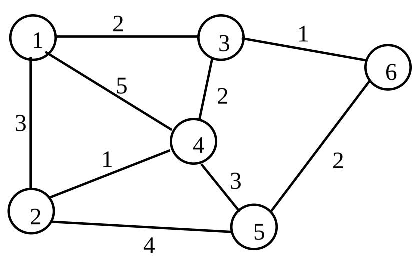
Node 5

Destination	Next node
1	4
2	2
3	4
4	4
6	6

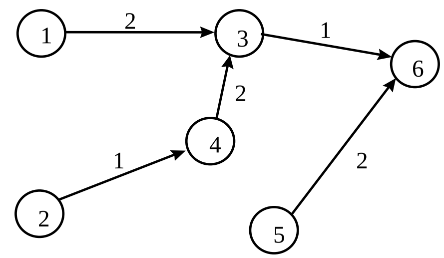
Node 6

Destination	Next node
1	3
2	5
3	3
4	3
5	5

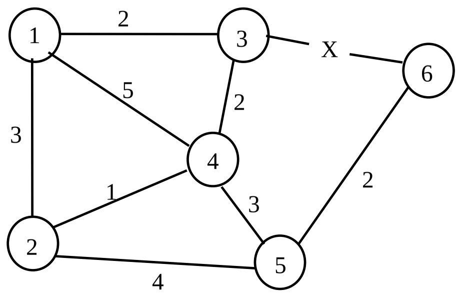
Sample Network with link costs



Shortest-path Tree to Node 6



New Network with link 3-6 broken



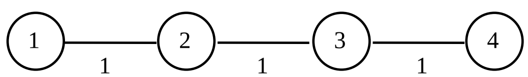
Routing entries to Destination Node 6

Update	1	2	3	4	5
Before break	(3,3)	(4,4)	(6,1)	(3,3)	(6,2)
1	(3,3)	(4,4)	(4,5)	(3,3)	(6,2)
2	(3,7)	(4,4)	(4,5)	(2,5)	(6,2)
3	(3,7)	(4,6)	(4,7)	(2,5)	(6,2)
4	(2,9)	(4,6)	(4,7)	(5,5)	(6,2)
5	(2,9)	(4,6)	(4,7)	(5,5)	(6,2)

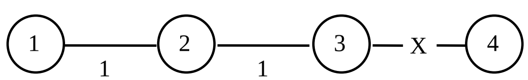
Each entry = (Next node, distance)

Problems with link failures

(a)



(b)



Routing Table

Update	Node 1	Node 2	Node3
Before Break	(2,3)	(3,2)	(4,1)
After Break	(2,3)	(3,2)	(3,3)
1	(2,3)	(3,4)	(3,3)
2	(2,5)	(3,4)	(3,5)
3	(2,5)	(3,6)	(3,5)
4	(2,7)	(3,6)	(3,7)
…….	……..	…….	…...

Loop-Breaking Heuristics

- Set infinity to 16
- Split horizon
  - Don’t send minimum cost to neighbor if neighbor is the NextHop
- Split horizon with poison reverse
  - Send minimum cost to all neighbors, but set infinity to the NextHop neighbor
- Work satisfactorily in some cases

Routing Table Example (Split Horizon with Poisoned Reverse)

Update	Node 1	Node 2	Node3
Before break	(2,3)	(3,2)	(4,1)
After break	(2,3)	(3,2)	(-1,Inf)
1	(2,3)	(-1, Inf)	(-1,Inf)
2	(-1, Inf)	(-1,Inf)	(-1,Inf)

Link State

- Strategy
  - send to all nodes (not just neighbors) information about directly connected links (not entire routing table)
- Link State Packet (LSP)
  - id of the node that created the LSP
  - cost of the link to each directly connected neighbor
  - sequence number (SEQNO)
  - time-to-live (TTL) for this packet

Link State (cont)

- Reliable flooding
  - store most recent LSP from each node
  - forward LSP to all nodes but one that sent it
  - generate new LSP periodically
    - increment SEQNO
  - start SEQNO at 0 when reboot
  - decrement TTL of each stored LSP
    - discard when TTL=0

Route Calculation

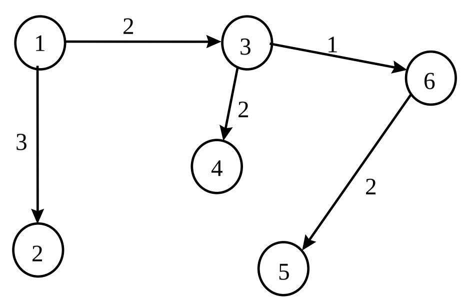
- Dijkstra's shortest path algorithm
- Let
  - *N* denotes set of nodes in the graph
  - *l* (*i*, *j*) denotes non-negative cost (weight) for edge (*i*, *j*)
  - *s* denotes this node
  - *M* denotes the set of nodes incorporated so far
  - *C*(*n*) denotes cost of the path from *s* to node *n*

```
M = {s}
for each n in N - {s}
    C(n) = l(s, n)
while (N != M)
    M = M union {w} such that C(w)
    is the minimum for all w in (N - M)
    for each n in (N - M)
        C(n) = MIN(C(n), C (w) + l(w, n ))
```


Example Route Calculation

Iteration	N	2	3	4	5	6
0	{1}	3	2	5	Inf	Inf
1	{1,3}	3	2	4	Inf	3
2	{1,2,3}	3	2	4	7	3
3	{1,2,3,6}	3	4	4	5	3
4	{1,2,3,4,6}	3	2	4	5	3
5	{1,2,3,4,5,6}	3	2	4	5	3

Shortest-path tree from node 1



Metrics

- Original ARPANET metric
  - measures number of packets enqueued on each link
  - took neither latency or bandwidth into consideration
- New ARPANET metric
  - stamp each incoming packet with its arrival time (**AT**)
  - record departure time (**DT**)
  - when link-level ACK arrives, compute
    Delay = (DT - AT) + Transmit + Latency
  - if timeout, reset **DT** to departure time for retransmission
  - link cost = average delay over some time period
- Fine Tuning
  - compressed dynamic range
  - replaced dynamic with link utilization

How to Make Routing Scale

- Flat versus Hierarchical Addresses
- Inefficient use of Hierarchical Address Space
  - class C with 2 hosts (2/255 = 0.78% efficient)
  - class B with 256 hosts (256/65535 = 0.39% efficient)
- Still Too Many Networks
  - routing tables do not scale
  - route propagation protocols do not scale

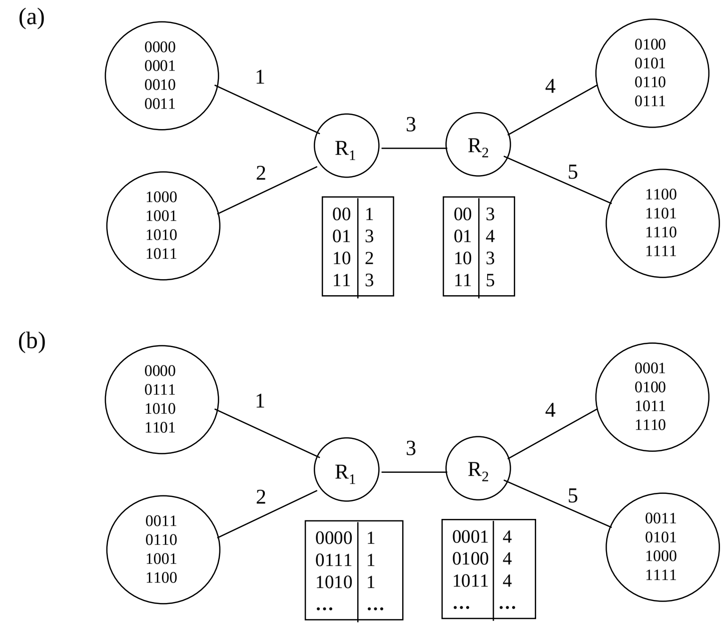
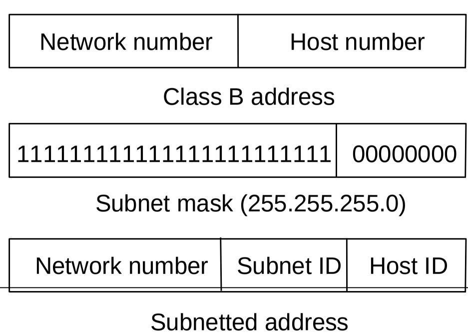


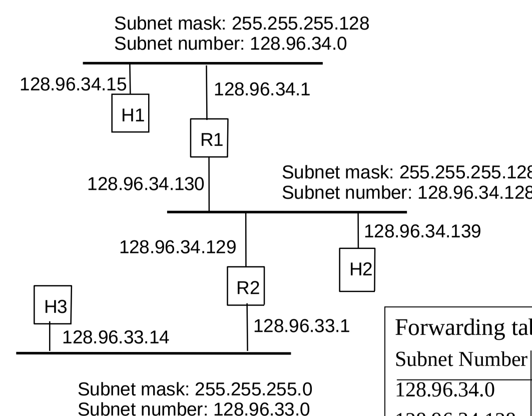
Figure 7.27

Subnetting

- Add another level to address/routing hierarchy: *subnet*
- *Subnet masks* define variable partition of host part
- Subnets visible only within site



Subnet Example



Forwarding table at router R1

Subnet Number	Subnet Mask	Next Hop
128.96.34.0	255.255.255.128	interface 0
128.96.34.128	255.255.255.128	interface 1
128.96.33.0	255.255.255.0	R2

Forwarding Algorithm

```
D = destination IP address
for each entry (SubnetNum, SubnetMask, NextHop)
   D1 = SubnetMask & D
   if D1 = SubnetNum
      if NextHop is an interface
         deliver datagram directly to D
      else
         deliver datagram to NextHop
```

- Use a default router if nothing matches
- Not necessary for all 1s in subnet mask to be contiguous
- Can put multiple subnets on one physical network
- Subnets not visible from the rest of the Internet

Supernetting

- Assign block of contiguous network numbers to nearby networks
- Called CIDR: Classless Inter-Domain Routing
- Represent blocks with a single pair
  `(first_network_address, count)`
- Restrict block sizes to powers of 2
- Use a bit mask (CIDR mask) to identify block size
- All routers must understand CIDR addressing

Route Propagation

- Know a smarter router
  – hosts know local router
  – local routers know site routers
  – site routers know core router
  – core routers know everything
- Autonomous System (AS)
  – corresponds to an administrative domain
  – examples: University, company, backbone network
  – assign each AS a 16-bit number
- Two-level route propagation hierarchy
  – interior gateway protocol (each AS selects its own)
  – exterior gateway protocol (Internet-wide standard)