Lecture 5 - The Network Layer

Computer Communication Networks
CS35201 - (001/ 002)
Fall 2022

Kent State University

Department of Computer Science
Betis Baheri
bbaheri@kent.edu

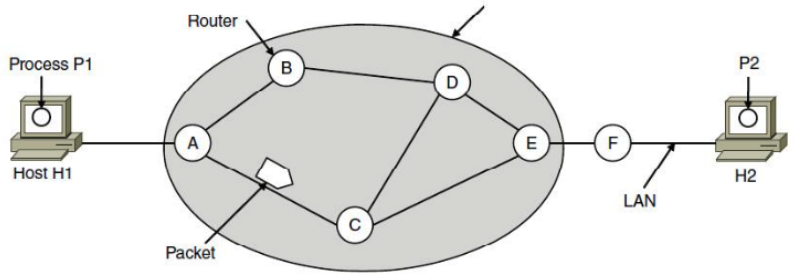
October 17, 2022

Outline I

Network Layer Design Issues

1. Store-and Forward packet switching
2. Facilitating getting data from a source to a destination
3. Data link layer move frames only one hop
4. Must know the topology of network and available paths
5. Load balancing $\Rightarrow$ traffic management
6. Services provided to transport layer
7. Implementation of
   - Connection-oriented services
   - Connection-less services
8. Comparison of virtual-circuit and data-gram network

Store-and-Forward Packet Switching



- Packet Delivery Model
  - A global addressing scheme is needed
- Service Model
  - Connection-less (datagram-based) hat supports
    - Connection-orient transport protocol ⇒ Transport Control Portocol (TCP)
    - Connection-less transport protocol ⇒ User Data Protocol (UDP)
- Best-effort delivery (unrealiable service)
  - packets are lost
  - packets are delivered out of order
  - duplicate copies of a packet are delivered
  - packets can be delayed for a long time

Services Provided to the Transport Layer

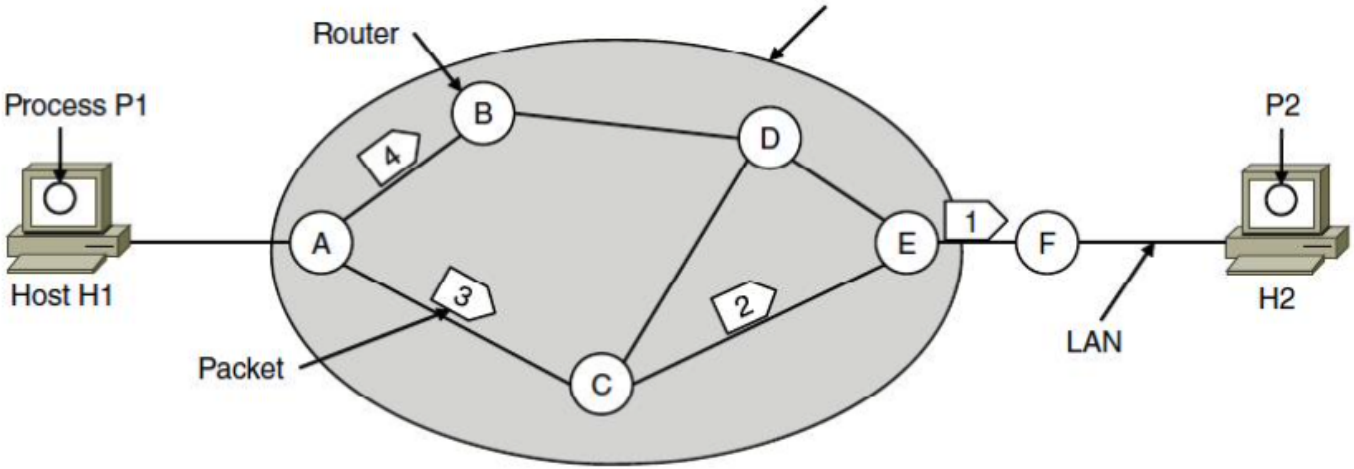
- Services are generally independent of router technology
- Transport layer is shielded from number, type, topology of routers
- Network addresses available to transport layer use uniform numbering plan

Two Implementation Techniques

1. Virtual circuits:
   - The complete route is set up in advance
2. Datagrams
   - Each is routed independently
   - Route is determined on the fly
     - They hop from router to router

Implementation of Connectionless Service

Routing within a datagram network



A's table (initially)

Dest.	Line
A	
B	B
C	C
D	B
E	C
F	C

A's table (later)

A	
B	B
C	C
D	B
E	D
F	D

C's table

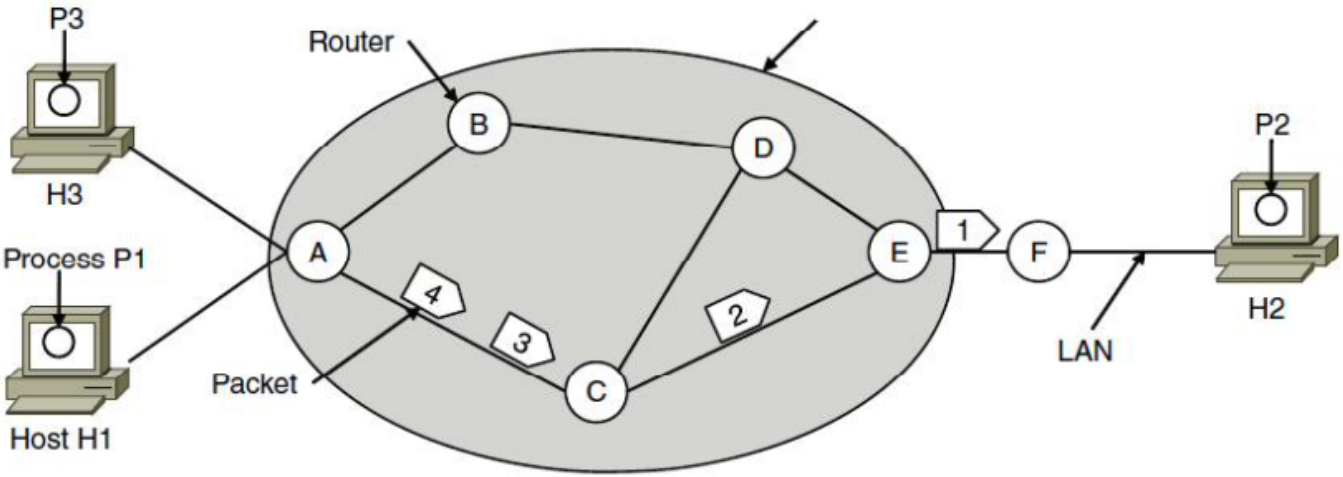
A	A
B	A
C	
D	E
E	E
F	E

E's table

A	C
B	D
C	C
D	D
E	
F	F

Implementation of Connection-Oriented Service

Routing within a virtual-circuit network



A's table

In		Out	
H1	1	C	1
H3	1	C	2

C's table

In		Out	
A	1	E	1
A	2	E	2

E's table

In		Out	
C	1	F	1
C	2	F	2

Two connections:

H3-A-C-E-F-H2 ⇒ 1A1C1E1F1 ⇒ label switching

H1-A-C-E-F-H2 ⇒ 1A2C1E1F1 2 circuits/channels between A and C

Virtual-Circuit vs. Datagram Networks

Issue	Datagram network	Virtual-circuit network
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

Outline I

Routing

- Routing tables carry
  - Internet Protocol (IP) addresses to get to distant networks
  - IP addresses to get to local hosts
- Lookup table is used for incoming IP packets
- If the network is not present, the packet is forwarded to a default router with more extensive table
- Each router only has to keep track of other networks and local hosts, not network pairs, reducing table size

Forwarding versus Routing

- **Forwarding:** selects an output port based on destination address and routing table
- **Routing:** process by which routing table is built

Routing

- Process of finding a path from a Src to a Dst
- Issues
  - What route should you take?
  - Does a shorter route exist?
  - What if a link along the route goes down?
  - What if you are on a mobile wireless link?
- A routing protocol sets up a routing table in routers switch controllers
- A node makes a local choice depending on global topology
  - This is a fundamental problem
- Other problems
  - How to make correct local decisions?
    - Each router must know something about global states
      inherently large ⇓
      dynamic ⇓
      hard to collect ⇓
  - Routing protocol must intelligently deduce relevant information

Routing Objectives

- Minimize routing table space for
  - fast look up
  - less to exchange
- Minimize the number and frequency of messages
- Avoid black holes, loops and oscillations
- Use optimal paths
- Factors:
  - Static: topology
  - Dynamic: load

Low Cost Routing

- Two major approaches
  - Distance Vector: Bellman-Ford Algorithm
    - Each router sends a vector of global distances to its neighbors
    - Global information to local neighbors
  - Link State: Dijkstra Algorithm
    - Each router sends a vector of local distances to all nodes
    - Local neighbor information to global nodes
    - $\Rightarrow$ Internet approach
- Both assume router knows
  - address of each neighbor
  - cost of reaching each neighbor
- Both allow a router to determine global routing information by talking to its neighbors

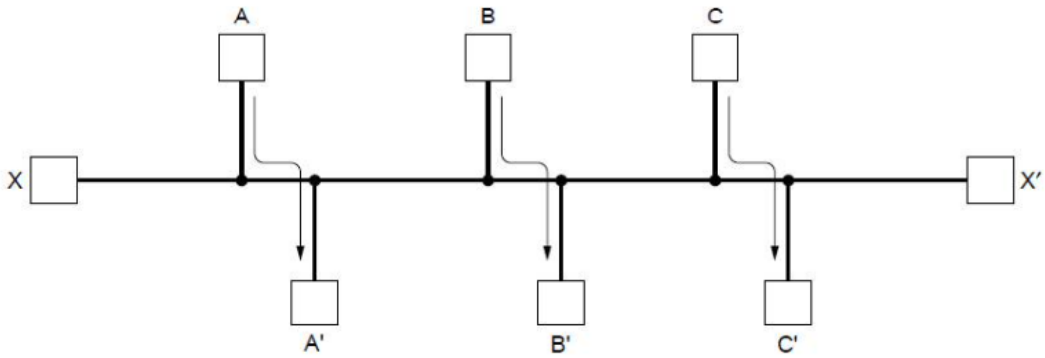
Low Cost Routing

1. Optimality principle
2. Shortest path algorithm
3. Flooding
4. Distance vector routing
5. Link state routing
6. Routing in ad hoc networks
7. Broadcast routing
8. Multicast routing
9. Anycast routing
10. Routing for mobile hosts
11. Routing in ad hoc networks

⋮

Fairness vs. Efficiency

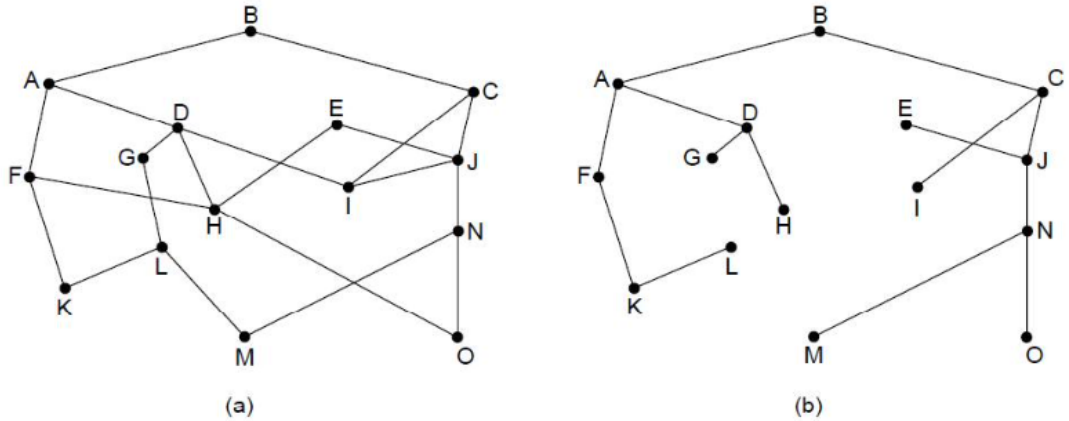
100% efficient, but unfair *Why?*



- X to X' is restricted (blocked) by other connections
- Routing is topology-sensitive

The Optimality Principle

- Routers should cooperate to find the best routes between all pairs of stations
- All optimal routes from station A to other stations in the network, jointly constitute a sink tree

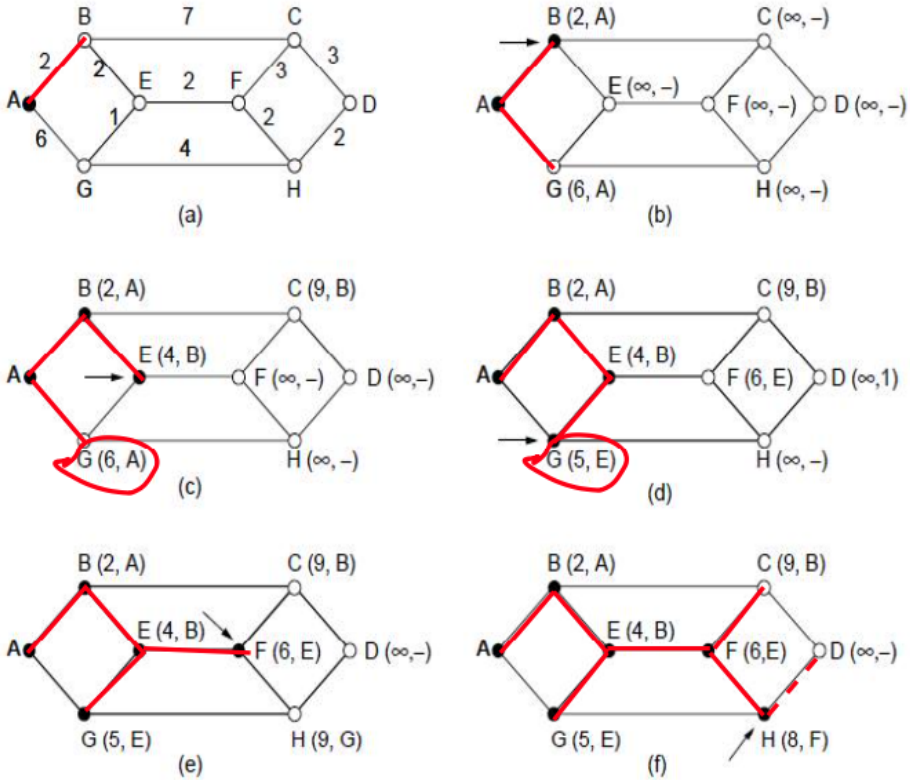


- Routers have to collaborate to build the sink tree for each source station

Shortest Path Algorithm

Dijkstra's Algorithm

- Labels on the arcs represent the cost (e.g., distance, delay,, etc.)
- It select a newly reachable node at the lowest cost $O(N)$
  - Creates an edge of the tree



The first five steps used to find the shortest path from A to D. The arrows indicate the working node

Flooding

- Forward an incoming packet across every outgoing line
  - Except the one it came in through
- Problem: how to avoid cycling ? $\Rightarrow$ drowning by packets?

Anti-cycling

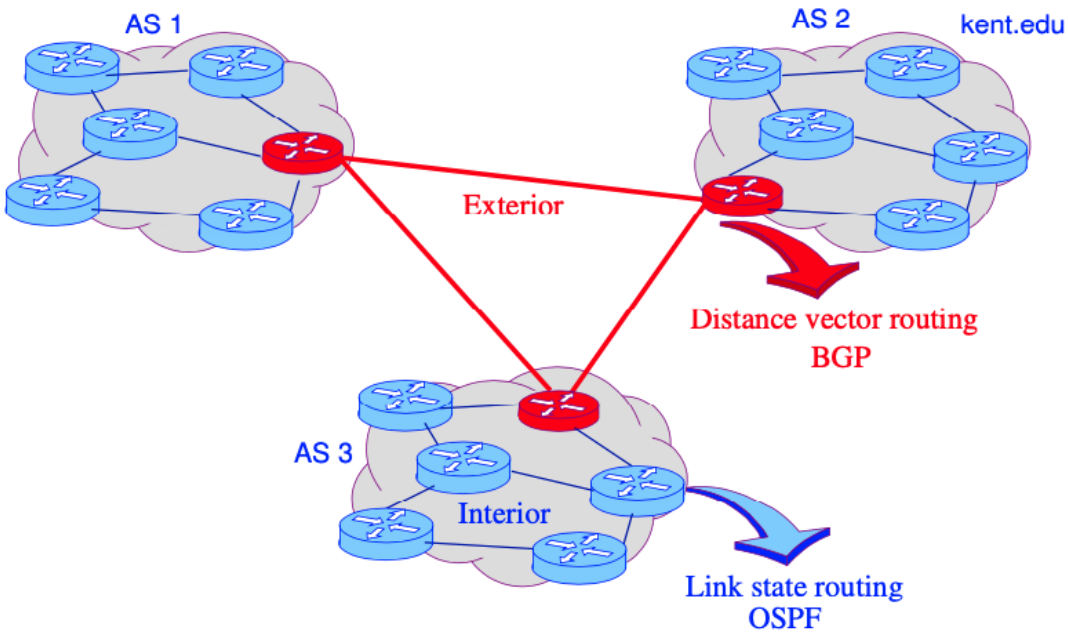
1. Use a hop counter $\Rightarrow$ IP does that
   - After a packet has been forwarded across N routers, it is discarded
   - How to find the right hop count? $\Rightarrow$ network diameter
2. Be sure to forward a packet only once $\Rightarrow$ avoid directed cycles
   - This requires sequence numbers per source router
   - Each router keeps track of the last sequence number per source router
   - Not scalable
3. Flood selectively
   - Only in the direction that makes sense
   - Requires some knowledge of network topology

Flooding

- Flooding always chooses the **shortest path** *Why?*
  - Because all path are explored in parallel
  - The overhead grows significantly ⇓
- Flooding makes sense only when **robustness** is needed
- Flooding is useful (natural) in wireless networks *Why?*
  - Broadcasting:
    - A message transmitted by a station is received by all other stations in the range
- These protocols are static
  - They do not take the **current** network load into account
- Dynamic Protocols⇒ Later
  1. **Distance Vector** Routing ⇒ Bellman-Ford Algorithm
  2. **Link State** Routing ⇒ Dijkstra Algorithm

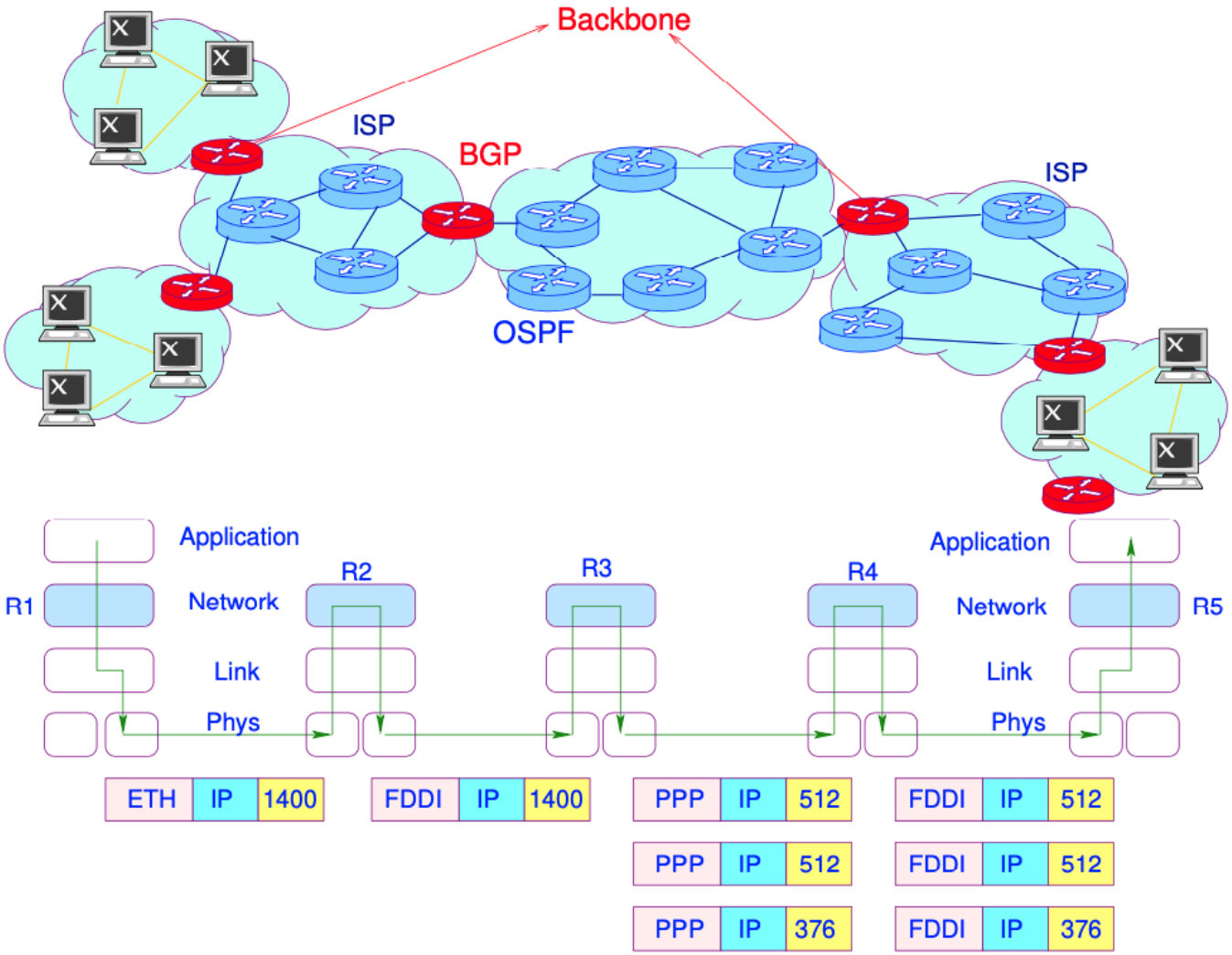
Outline I

Internet Infrastructure



- Within Autonomous Systems (ASs)
  $\Rightarrow$ use Link Sate (Open Shortest Path First (OSPF))
- Between ASs $\Rightarrow$ use Distance Vector (Boarder Gateway Protocol (BGP))

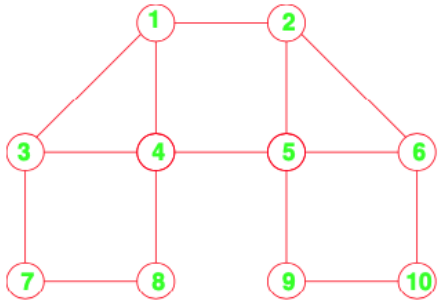
Internet Routing



Distance Vector Routing
Bellman-Ford Algorithm

1. A node receives distance vectors (cost) from neighbors
   - Nodes tell neighbors the best way to get to other nodes
2. The node updates its distance (cost) to the destinations
3. The node advertises this information to its neighbors

- Features
  - Distributed algorithm
  - Adapts to traffic changes and failures



Routing table for Node 4

Dst	Next hop
1	1
2	5
3	3
5	5
6	5

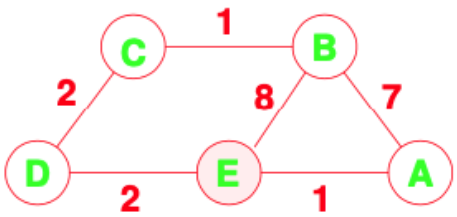
Dst	Next hop
7	3
8	8
9	5
10	5

Why Does it Work?

- Each node knows its cost to its neighbors
- This information is spread to its neighbors
- Subsequent dissemination spreads the truth one hop at a time
- Eventually, the information is incorporated into routing tables

Distance Vector Routing
Bellman-Ford Algorithm

- Iterative
  - Continues until no nodes exchange information
  - Self-terminating ⇒ no stopping mechanism
- Asynchronous ⇒ no lock step
- Distributed
  - Each node communicates only with its neighbors to find new routes



Src E	via A	via B	via D
Dst A	1	14*	5*
B	7*	8	5
C	6*	9	4
D	4*	11	2

* loop

- For node E d(X,Y,Z) means distance from X to Y via Z
  - d(E,A,A)=EA(1) or d(E,A,B)=EBCDEA(14*) or d(E,A,D) = EDEA (5*)
  - d(E,B,A)=EAEDCB(7*) or d(E,B,B)=EB(8) or d(E,B,D) = EDCB(5)
  - d(E,C,A)=EAEDC(6*) or d(E,C,B)=EBC(9) or d(E,C,D)=EDC(4)
  - d(E,D,A)=EAED(4*) or d(E,D,B)=EBCD(11) or d(E,D,D)=ED(2)

Distance Vector Routing
Bellman-Ford Algorithm

Distance Vector ⇒ Routing Table

Src E	via A	via B	via D
Dst A	1	14	5
Dst B	7	8	5
Dst C	6	9	4
Dst D	4	11	2

Routing table for E

Src E	Cost
Dst A	1
Dst B	5
Dst C	4
Dst D	2

E shares this table wit its neighbots

- Routing table is updated if least cost path to any destination has changed

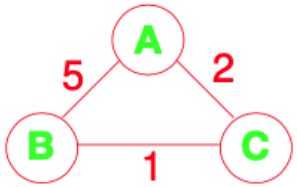
Distance Vector Algorithm (Bellman Ford)

For all nodes, X

1. Initialization: For all adjacent node Y,
   - d(X,Y) = ∞
2. For all destination Z,
   - Advertise Min d(Z,Y) to each neighbor

Distance Vector Example

Example I



Exchange tables (distance vectors)

A	B	C
B	5	∞
C	∞	2

B	A	C
A	5	∞
C	∞	1

C	A	B
A	2	∞
B	∞	1

A finds BC=1, CB=1 ⇒

B finds CA=2, AC=2 ⇒

C finds AB=5, BA=5

A	B	C
B	5	*3*
C	6	*2*

B	A	C
A	5	*3*
C	7	*1*

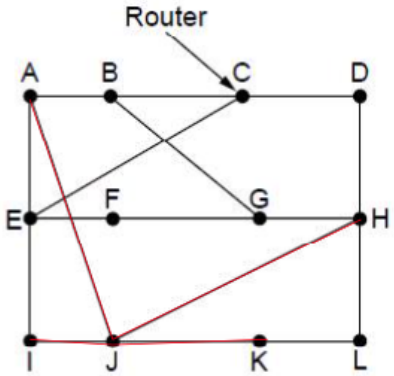
C	A	B
A	*2*	6
B	7	*1*

➥ *Until no new shortest route found*

Distance Vector Example

Example II

⇒ Consider node J and its neighbors A, I, H, K



(a)

New estimated delay from J

To	A	I	H	K	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

Vectors received from J's four neighbors

New routing table for J

(b)

(a) A network
(b) J receives input from A, I, H, K, and calculate a new routing table for J.

Link cost =1

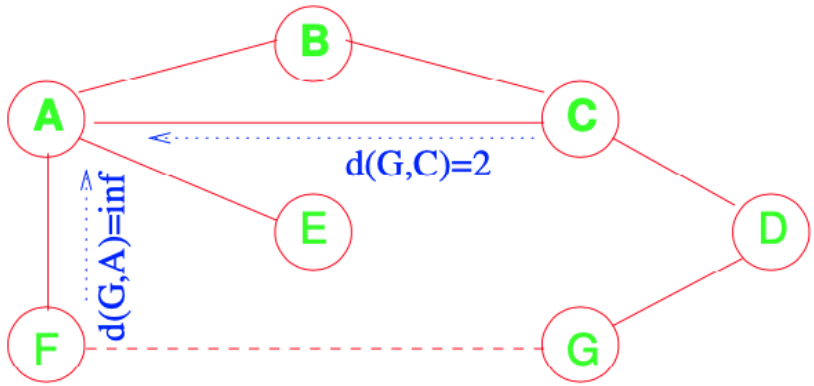
⇒ DV is expensive ⇒ $O(n^3)$ ⇒ Too slow to converge

Why?

Routing Process in Practice

No Failure

1. F detects that the link to G has failed $\Rightarrow$ d(G,F) = ∞
2. F advertise d(G,F) = ∞ A find d(G,F) = ∞
3. A advertise d(G,A) = ∞
4. A also receives d(G,C) = 2
5. A now advertise d(G,C) = 3
6. F knows d(A,G) = 4

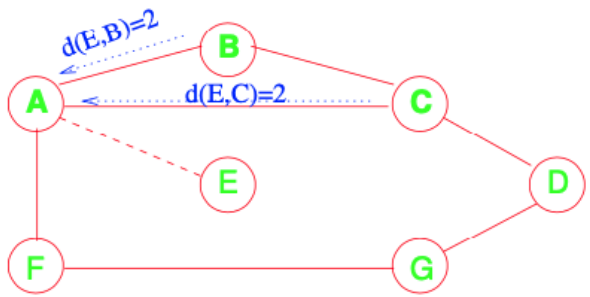


Count to Infinity Problem
Distance Vector

Senario: Link A to E Fails

1. A detects that link to E is failed $\Rightarrow$ d(E,A) = ∞
   - B, C, D, G, F have not received this information yet
   - C and F tell A that d(E,B) =2 and d(E,C)=2
2. A advertises d(E,A) = ∞
3. A receives d(E,B) =2 and d(E,C)=2
4. B finds and advertises d(E,C) =3
5. A finds and advertises d(E,C) =4
6. C finds and advertises d(E,A) =5
7. A finds and advertises d(E,C) =6

$\Rightarrow$ this goes on



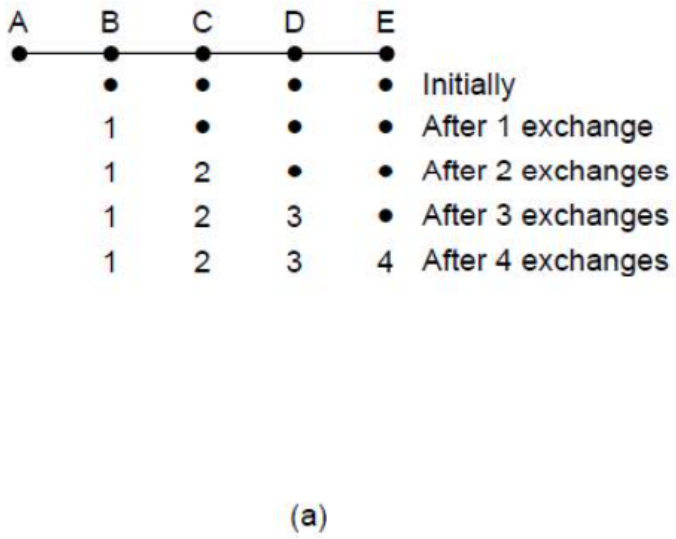
- A wrongly calculate a new distance to E from inaccurate information received by B and C
- What do you observe? E has only one link to the rest $\Rightarrow$ deg(E) =1

Count to Infinity Problem
Distance Vector

Another Scenario (count to infinity

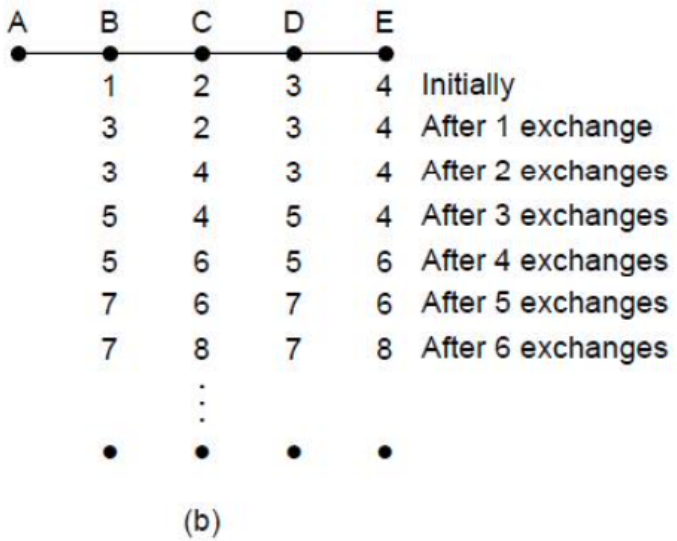
- A suddenly goes down
- When A is down C claims it is away from A by 2 hops via B
  - C propagates this information
  - B and D add 1 to their distance to A $\Rightarrow$ 3
- C finds from B it can reach A in 3 hops, it calculates its distance (4)

Initial build up



(a)

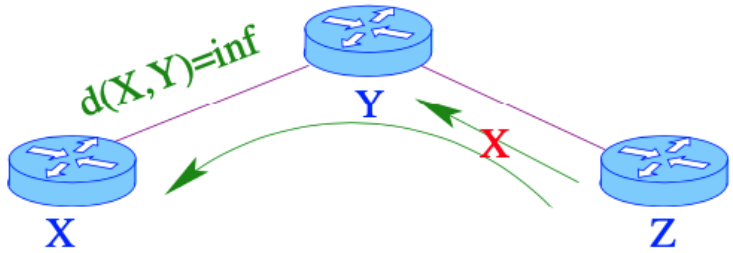
count to infinity propagation



(b)

A Few Partial Solutions

- Use a relatively small number instead of ∞
  - Diameter of the network
- Split horizon
  - If Z route through Y to get to X
    - Y is an entry in X's routing table
  - Z does not advertise its route to X via Y
    - Y has a lower cost to X than Z



- Split horizon with poison reverse
  - If Z routes through Y to get X
  - It tells Y, d(X,Z) =∞

Outline I

Link State Routing
OSPF

- Distance vector routing was used in the ARPANET until 1979
- Then it was replaced by Link State Routing
  - Broadcast information to the entire network
  - Let each router calculate its own sink tree
    - Minimum spanning tree
- Broadcast info on the entire network topology to all routers
  - Let each router calculate a sink tree to the other routers

Each router does the following steps:

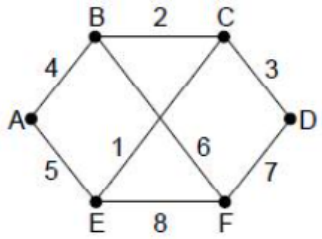
1. Finds out who its neighbors are and get their network addresses
2. Calculates the cost (time) for getting a packet to a neighbor
3. Constructs a LinK State Packet (LSP) telling all it has just learned
4. Sends LSP to all other routers (not just neighbors)
5. Runs Dijkstra's algorithm locally

Link State Routing
OSPF

- Strategy: Send to all nodes (not just neighbors) information about directly connected links (not the entire routing table)
- LSP contains the:
  - ID of the node that created the LSP
  - Cost of link to each directly connected neighbor
  - Sequence Number (SEQNO)
  - Time-To-Live (TTL) for this packet
- 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

Link State Routing
OSPF

- How to measure the delay?
  - Just send an ECHO packet through the interface and measure Round Trip Time (RTT)
- Do we take local load into account?
  - Queuing delay? We should
- This could redirect traffic and cause load oscillations
  - Every node uses flooding to sends its LSP to all other routers



(a)

A		B		C		D		E		F	
Seq		Seq		Seq		Seq		Seq		Seq	
Age		Age		Age		Age		Age		Age	
B	4	A	4	B	2	C	3	A	5	B	6
E	5	C	2	D	3	F	7	C	1	D	7
		F	6	E	1			F	8	E	8

Link State Packets

(b)

(a) A network. (b) The link state packets for this network

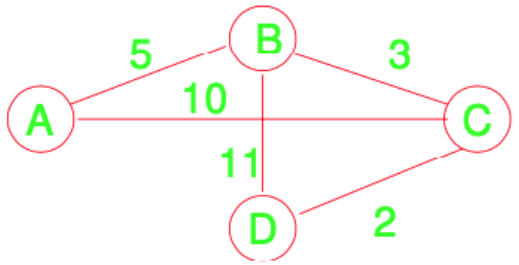
- The combination of Seq. and Age identify recycled LSP packets
- Entries age and purged

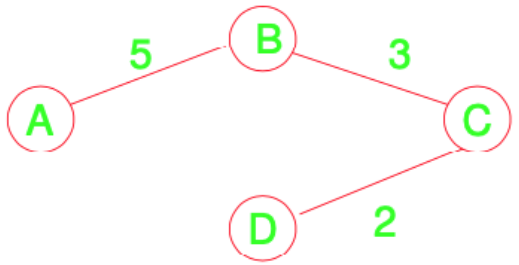
LS Route Calculation

Example I

Example 5.1 (Dijkstra's Shortest)

Step	Confirmed	Tentative
1.	(D,0,-)	
2.	(D,0,-)	(B,11,B) (C,2,C)
3.	(D,0,-) (C,2,C)	(B,11,B)
4.	(D,0,-) (C,2,C)	(B,5,C) (A,12,C)
5.	(D,0,-) (C,2,C) (B,5,C)	(A,12,C)
6.	(D,0,-) (C,2,C) (B,5,C)	(A,10,C)
7.	(D,0,-) (C,2,C) (B,5,C) (A,10,C)	





Spanning tree

LS Route Calculation

Dijkstra's shortest path algorithm (OSPF)

- Let N denotes the set of nodes in the graph
- $\ell(i,j)$ denotes non-negative cost (weight) for edge (i,j)
- $s \in N$ denotes the source
- M denotes the set of nodes incorporated so far
- $C(n)$ denotes cost of the path from s to node n

1. Initially $M = \{s\}$, the source
2. For each $n \in N - \{s\}$ calculate $C(n) = \ell(s,n)$
3. While $(N \neq M)$
   a. Find w such that $C(w)$ is minimum $\forall w \in N - M$
   b. $M = M \cup \{w\}$
   c. For each $n \in N - M$ update $C(n) = \min\{C(n), \underbrace{C(n) + \ell(w,n)}_{\text{new route found}}\}$

LS Route Calculation

Dijkstra's shortest path algorithm (OSPF)

- It is a forward search algorithm
- Each router maintains two lists: Tentative and Confirmed
- Each list contains a set of triples: (Destination, Cost, NextHop)

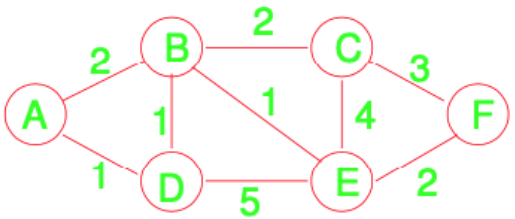
Dijkstra's Shortest Path Algorithm

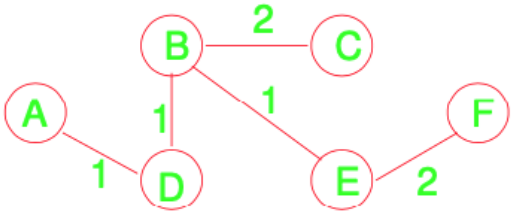
1. Initialized Confirmed with entry for me; cost = 0
2. For the node just added to Confirmed (call it Next) select its LSP
3. For each Neighbor of Next, calculate the Cost to reach this Neighbor as the sum of the cost from me to Next and from Next to Neighbor
   a. If Neighbor is currently in neither Confirmed or Tentative, add (Neighbor, Cost, NextHop) to Tentative, where NextHop is the direction to reach Next
   b. If Neighbor is currently in Tentative and Cost is less that current cost for Neighbor, then replace current entry with (Neighbor, Cost, NextHop), where NextHop is the direction to reach Next
   c. If Tentative is empty, stop. Otherwise, pick entry from Tentative with the lowest cost, move it to Confirmed, and return to step 2.

LS Route Calculation
OSPF Example II

Example 5.2 (Dijkstra's Shortest)

Step	Confirmed	Tentative
1	(A, 0 , –)	(B, 2, B) (D, 1, D)
2	(A, 0 , –) (D, 1 , D)	(B, 2, B) (B, 2, D) (E, 6, D)
3	(A, 0 , –) (D, 1 , D) (B, 2 , D)	(E, 3, D) (C, 4, D)
4	(A, 0 , –) (D, 1 , D) (B, 2 , D) (E, 3 , D)	(C, 7, D) X (C, 4, D) (F, 5, D)
4	(A, 0 , –) (D, 1 , D) (B, 2 , D) (E, 3 , D) (C, 4, D)	(F, 5, D)
5	(A, 0 , –) (D, 1 , D) (B, 2 , D) (E, 3 , D)	





Spanning tree

OSPF (Open Shortest Path First)

- Open : publicly available
- Recent Internet standard
- Supports load balancing
- Supports authentication
- Uses Link State algorithm
  - LSP packet dissemination
  - Topology map at each node
  - Route computation using Dijkstra's algorithm
- Advertisements carry one entry per neighbor router
- Advertisements disseminated via flooding

Distance Vector vs. Link State

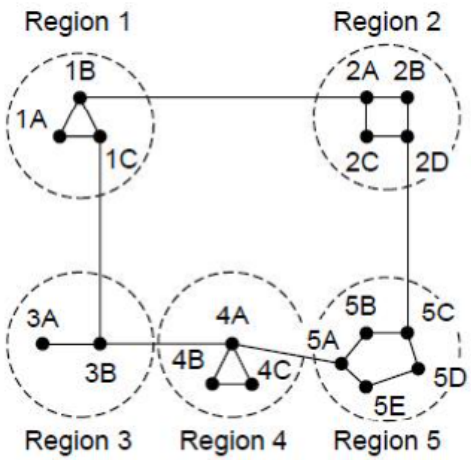
Metrics	Distance Vector Routing	Link State Routing
How	Global information (distances) shared with local neighbors	Local information shared with global network
Bandwidth	Less required due to local sharing, small packets, no flooding	More required due to flooding, sending large link state packets
Knowledge	Local knowledge, updates gathered from neighbors	Based on global knowledge, a router has knowledge about the entire network
Algorithm	Uses Bellman Ford Algorithm	Uses Dijkstra Algorithm
Complexity	$O(\mid V \mid \times \mid E \mid)$	$O(\mid E \mid + \mid V \mid \log \mid V \mid)$
Traffic	Less traffic generated	More traffic generated
Convergence	Slow	Fast
Count to ∞	Count to infinity problem	No Count to infinity problem
Looping	Persistent looping problem, could loop forever	No persistent looping problem, only transient loops
Use	Used in Routing Information Protocol (RIP) and Interior Gateway Routing Protocol (IGRP)	Used in OSPF and Intermediate System to Intermediate System (IS-IS) protocol

Outline I

Hierarchical Routing

- No routing algorithm discussed so far can scale
  - Link State (LS), BGP, Bellman Ford Algorithm, $O(\mid V \mid \times \mid E \mid)$
  - Distance Vector (DV), Interior Gateway Protocol (IGP), Dijkstra Algorithm, $O(\mid E \mid + \mid V \mid \log \mid V \mid)$, faster
  - All of them require each router to know about all others
- One solution is to go suboptimal
  1. Regions
  2. Inter Regions
  3. Intera Regions
- Two-level hierarchy: local area, backbone $\Rightarrow$ mostly geographic
  - Each node has detailed area topology; only knows direction (shortest path) to network in other areas
  - LS advertisements only in area
- Area Border Router (ABR): summarize distances to the network in own area
  - Advertise to other ABRs
- Backbone routers: run OSPF; routing limited to backbone
- Boundary routers: connect to other AS's

Hierarchical Routing
Example



(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)

- No optimal routes any more, e.g., 1A $\Rightarrow$ 5C

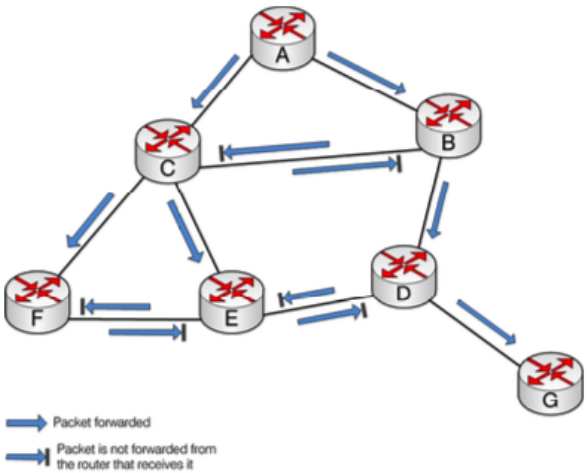
Broadcast Routing

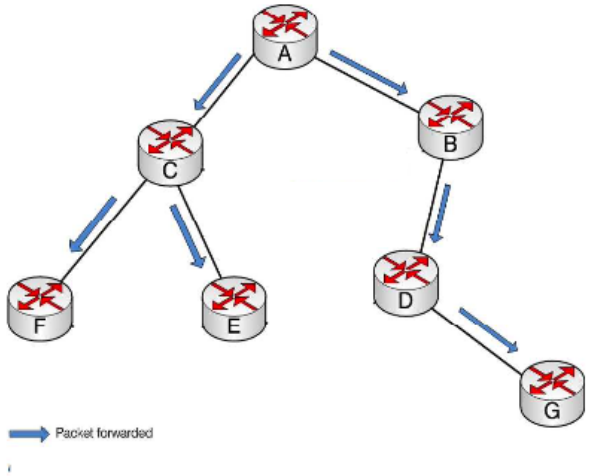
- We want to send a message to (almost) every host on the network
  - This means nearly a complete graph
- Options:
  1. Send the message to each host individually
     - Not really good *Why?*
       - ☆ $O(n)$
  2. Use flooding
     - Acceptable, provided that we can dam the flood
  3. Use multi-destination routing
     - A router checks the destinations, and splits the list when forwarding it across different output lines
     - The message must contain all the destinations
  4. Build a sink tree at the source and use that as your multicast route
     - The sink tree is a spanning tree (as in Dijkstra)
     - The routers need to know the trees

➥ *Reverse Path Forwarding*

Reverse Path Forwarding (RPF)

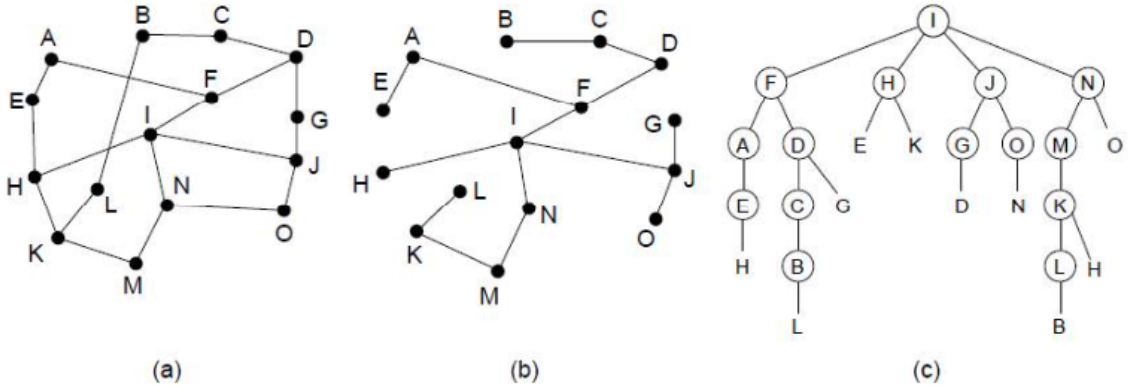
- A technique used in modern routers to ensure loop-free forwarding of multicast packets
  - Also to prevent IP address spoofing in unicast routing
- In standard unicast IP routing, a router forwards packets away from the source to make a spanning (lop free) tree
- In contrast, in Router Protocol Filtering (RPF), every router forwards a broadcast packet to every adjacent router item except the one where it received the packet router $\Rightarrow$ anti-looping
- Organizing tables based on the reverse path, from the receiver back to the root





Reverse Path Forwarding (RPF)

- Each router has a simple forwarding algorithm
- **If** multicast datagram received on incoming link on shortest path back to root,
  - **then** flood datagram onto all outgoing links
  - **else** ignore datagram
- Difference between sink tree and reverse path forwarding



(a) A network. (b) Uncast sink tree. (c) Reverse path forwarding

Multicast Routing

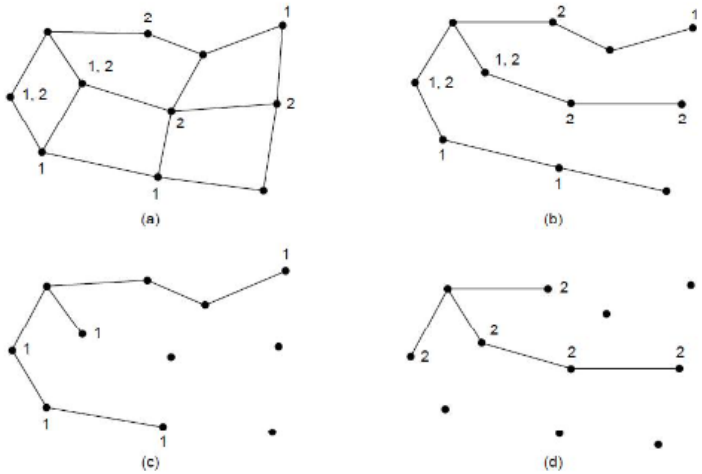
Question 5.1

Suppose that we want to send a message to only a subset of all the nodes in a network. How do we do that?

Answer:

- *We Construct a spanning tree (at each router) for the entire network.*
- *We prune paths to nodes that do not contain members for that group*

Two Multicasting Groups



(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

$\Rightarrow$ example: Chat groups

Multicast Routing

Question 5.2

What is the problem with the just presented solution ?

Answer: Scalability! We need to maintain a spanning tree per each broadcasting source

Outline I

Routing in Mobile Networks

Question 5.3

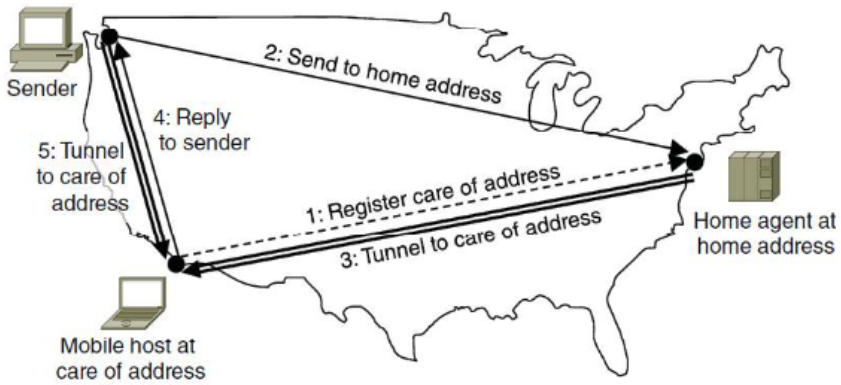
How one can forward packets to nodes that are constantly on the move?

Answer: *Use* *Home* *and* *Foreign* *agents* How?

1. The mobile hosts register with the foreign agents
2. The foreign agent contacts the mobile host's home agent and says:
   - One of your hosts is over here
3. When a packet is sent to a mobile host, it is routed to the host's home LAN
4. The home agent then forward the message to the foreign agent

- Similar to C/O in postal service
- Similar to Virtual Private Network (VPN)
- Similar protocol also used in Global System for Mobile communication (GSM) to track mobile users

Routing in Mobile Networks
Example



- Tunneling: sending an IP packet in an IP packet

Routing in Mobile Ad Hoc Networks

- Host Mobility + Wireless + Router Mobility
- No fixed infrastructure $\Rightarrow$ battlefield
- Dynamically changing topology
- Traditional routing are impractical

➥ *Ad Hoc Distance Vector (AoDV)*

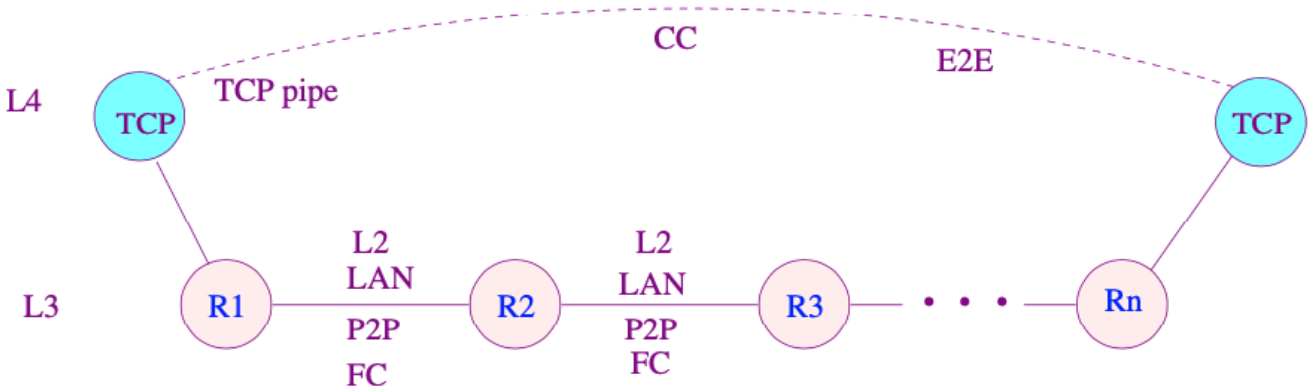
- It is an on-demand protocol :
  - Routes are set up only when required
- Represent the network as a graph
  - Two nodes are connected if they can communicate directly

Outline I

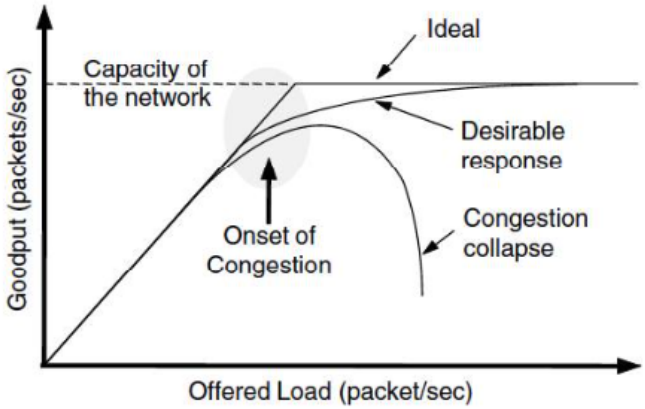
Congestion Control
Sources of Congestion

- Traffic burstiness $\Rightarrow$ high mean/var
- Lack of bandwidth
- Misconfiguration or slow routers
- Link failure
- Bottleneck
- Traffic oscillation due to re-routing
- Significant number of packet drops
- Flooding, flushing, DoS, DDoS,

$\vdots$



Congestion Control



$$\rho = \frac{\lambda}{\mu} = \frac{arrival\ rate}{departure\ rate(capacity)} < 1$$

➥ $\rho = 1$ *may result in congestion collapse*

Solutions for Congestion Control

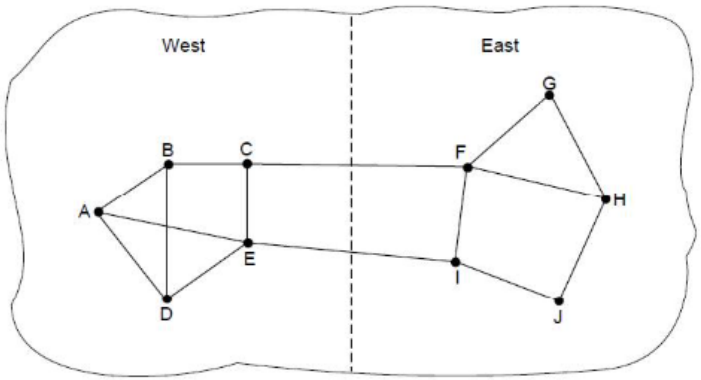
1. Network provisioning
2. Traffic aware routing
3. Admission control
4. Traffic shaping
5. Traffic throttling
6. Load shedding

Network provisioning

- Resource provisioning
  - Throw resources at the problem
- May solve the problem, but not for long
  - Not scalable

Traffic-Aware Routing

- Route based on the knowledge of traffic on links
  - May not be available $\Rightarrow$ dynamic traffic
- Bridged network

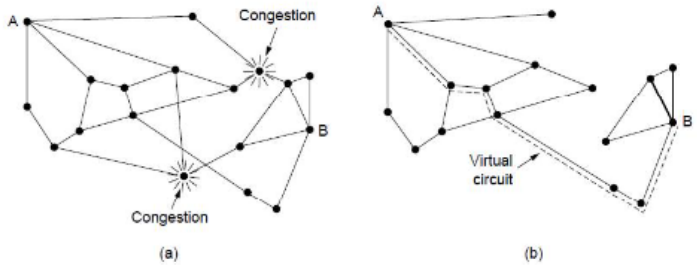


Admission Control

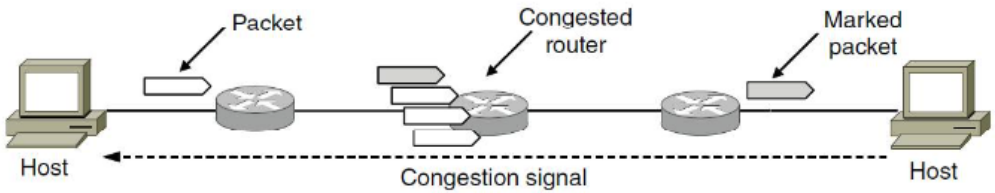
- When you set up a circuit, be sure that congestion can be avoided
- Refuse to set up a virtual circuit if close to congestion
- Similar to refusing an ftp request

Traffic Throttling

1. Select alternative routes when a part in the network is getting overloaded

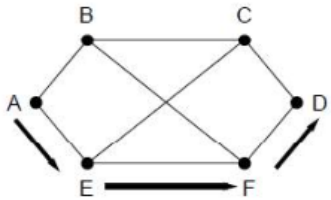


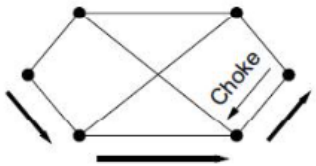
2. Explicit congestion notification

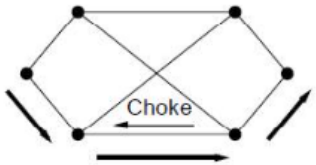


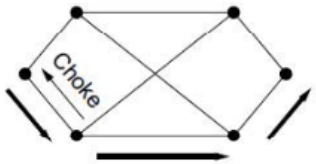
Load Shedding

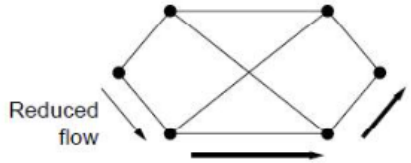
1. A choke packet is used that affects **only the source**

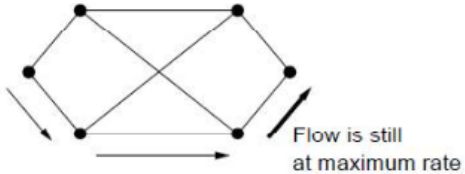


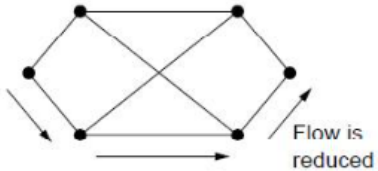








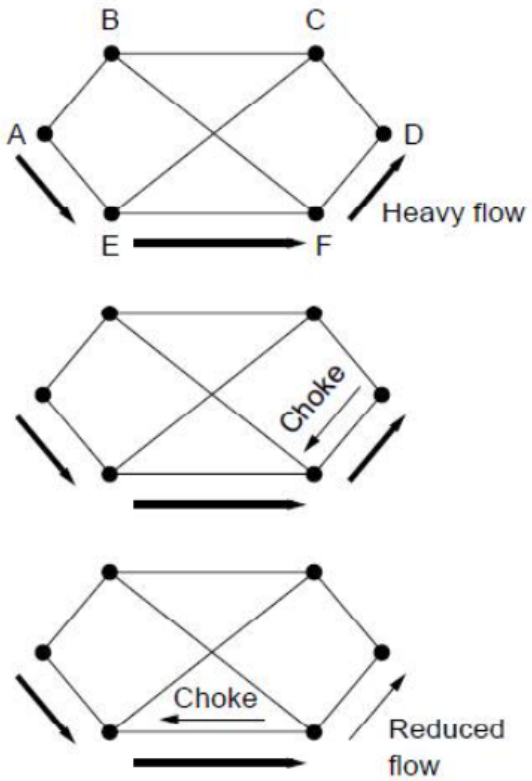


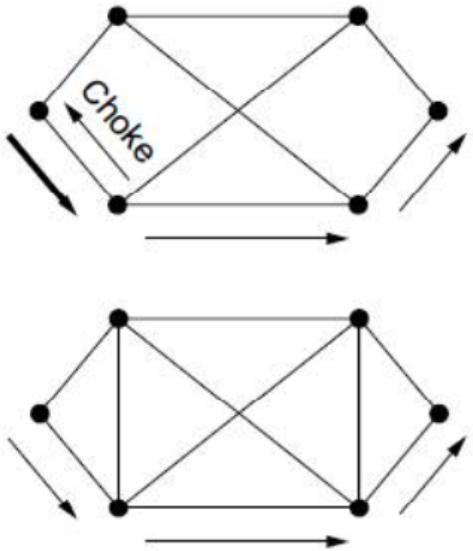


- Source may not be the source of congestion ⇓
- Takes time to inform the source ⇓
  - Congestion may disappear
    ⇒ unnecessary choke

Load Shedding

2. A choke packet that affects each hop it passes through





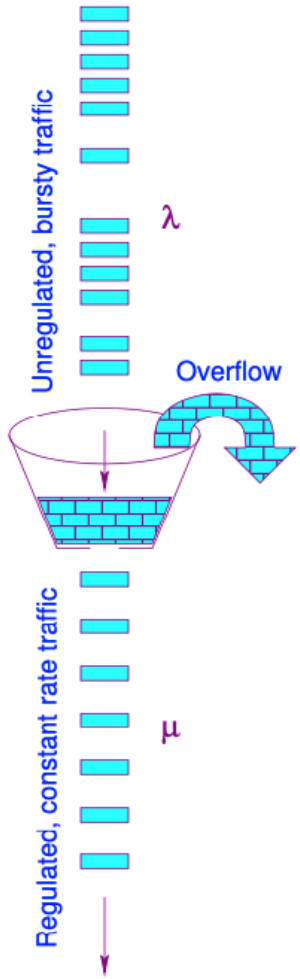
- All nodes reduce rate
  - Can cause oscillation

Traffic Shaping

- Bursty traffic is one of the causes of congestion
  - User behavior
  - Fair (round robin) queuing
  - Cumulative Acknowledgements
  - Denial of Service Attack (DoS) and Distributed Denial of Service Attack (DDoS)

Leaky Bucket: Rate and jitter control

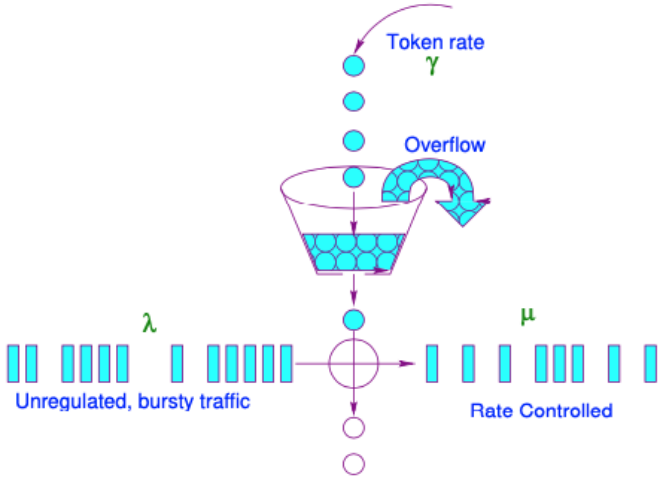
- Regulates/shape bursty traffic
  - Bursty traffic $\Rightarrow$ source of congestion
- Short bursts can be tolerated
  - If the bucket has enough room
- Long burst may overflow the bucket
  - Packets dropped
- What bucket size?
- Average arrival rate (λ) with high variance
- Constant departure rate (μ) with zero variance
- $\mu < \lambda$ when the bucket is full (overflowed)
  - Stable queues have $\mu > \lambda$



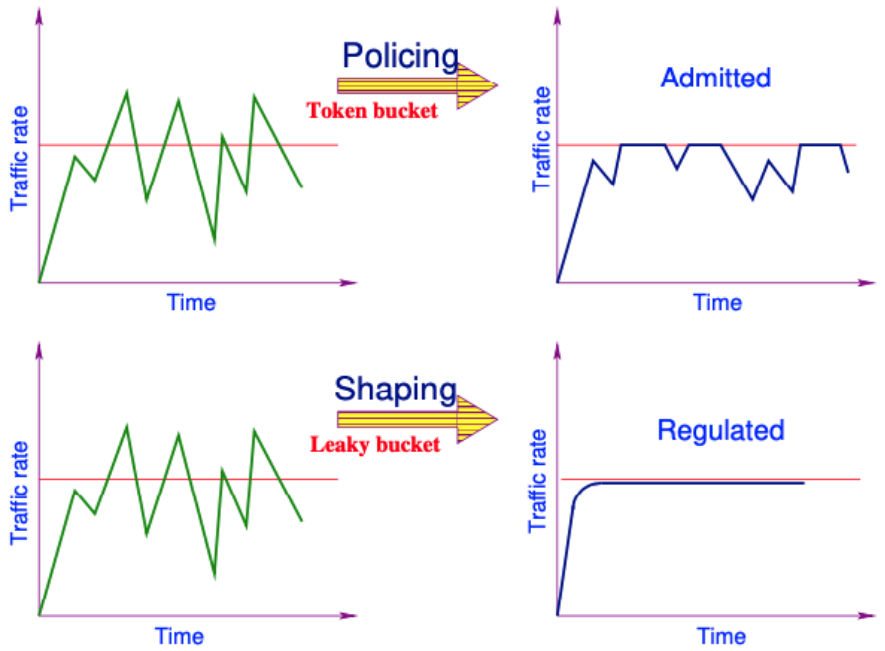
Traffic Policing (Regulating)

Token Bucket: Rate control

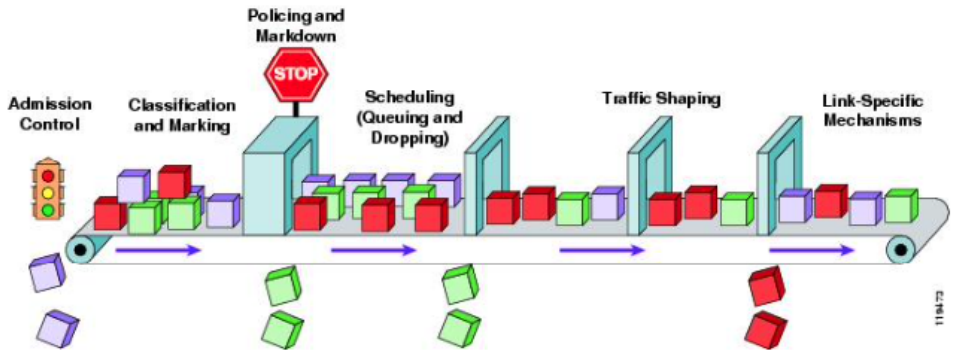
- Tokens are added at a constant rate (γ)
- An arriving packet facing an empty bucket cannot be forwarded
- Internet Service Providers (ISPs) use Token Bucket to allocate bandwidth to customers
  - Works like credit card limit
  - Controls rate but not burstiness
- How to control burstiness (jitter)?
  - Put a leaky bucket behind a token bucket (with a larger rate)
- Which one to control first? $\Rightarrow$ Rate or Shape?



Traffic Management in Practice



Cisco Approach



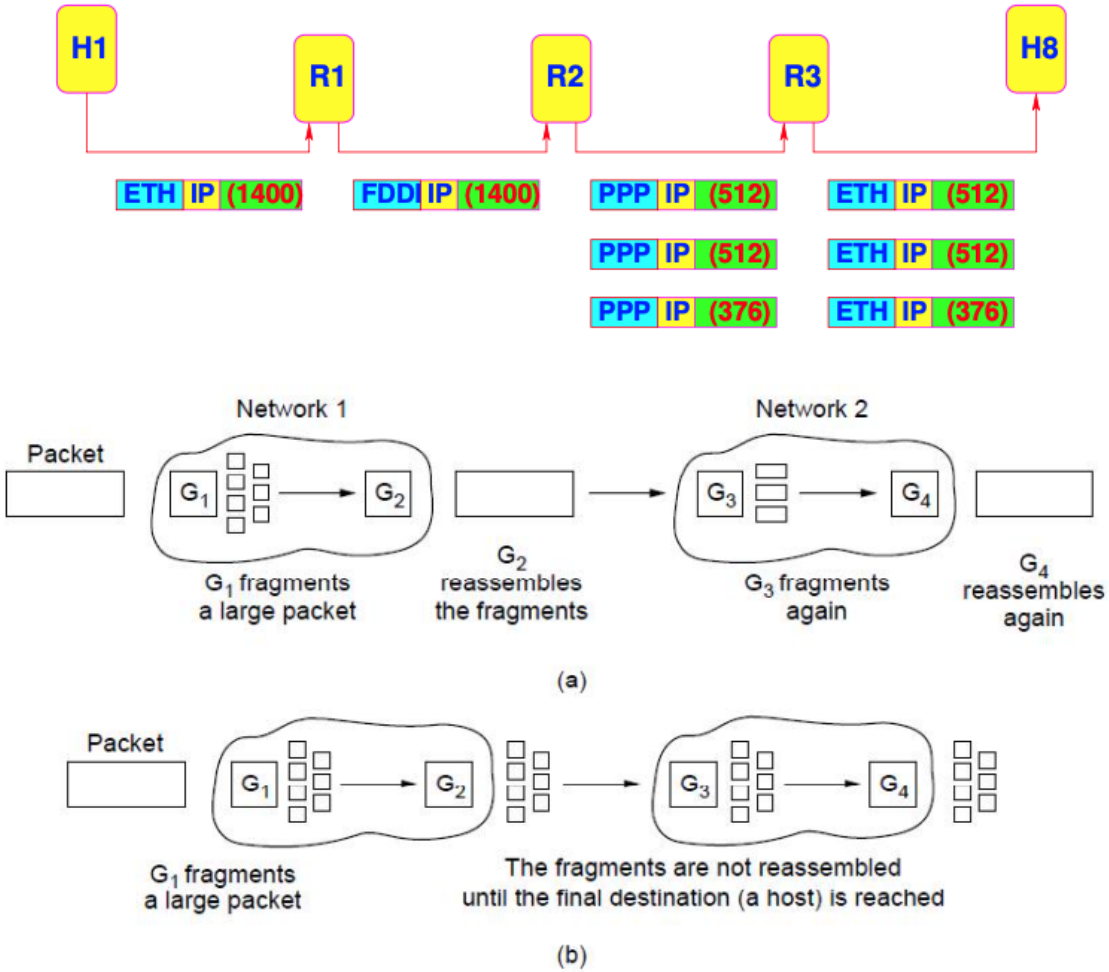
Quality of Service

- The needs of each flow are determined by
  1. Reliability ⇒ loss rate
  2. Delay
  3. Jitter) delay variation
  4. Bandwidth
- How stringent the quality-of-service requirements are?

Application	Bandwidth	Delay	Jitter	Loss
Email	Low	Low	Low	Medium
File sharing	High	Low	Low	Medium
Web access	Medium	Medium	Low	Medium
Remote login	Low	Medium	Medium	Medium
Audio on demand	Low	Low	High	Low
Video on demand	High	Low	High	Low
Telephony	Low	High	High	Low
Videoconferencing	High	High	High	Low

- How do we set the network to make them all happy?

Fragmentation and Reassembly



(a) Transparent fragmentation. (b) Non-transparent fragmentation

Fragmentation and Reassembly

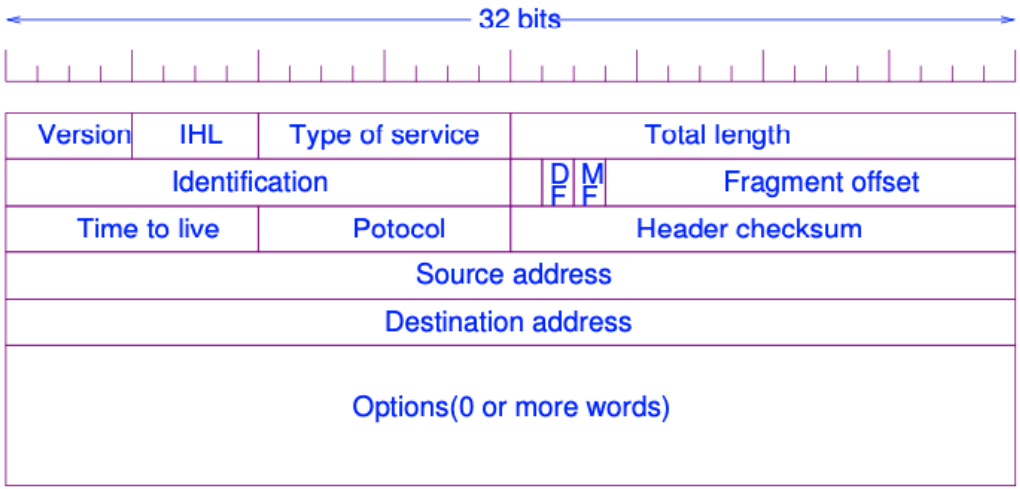
- Each network has a Max Transmission Unit (MTU)
- Strategy:
  - Fragment when necessary (Datagram $>$ MTU)
  - Try to avoid fragmentation at source host — *Why?*
    - re-fragmentation is possible — *How?*
  - Fragments are self-contained datagrams
  - Delay reassembly until destination host, — *Why?*
  - Do not recover from lost fragments — *Why?*

Outline I

The Network Layer in the Internet

1. The IP Version 4 Protocol
2. IP Addresses
3. IP Version 6
4. Internet Control Protocols
5. Label Switching and Multi-protocol Label Switching (MPLS)
6. OSPF: An Interior Gateway Routing Protocol
7. BGP: The Exterior Gateway Routing Protocol
8. Internet Multicasting
9. Mobile IP

IPv4

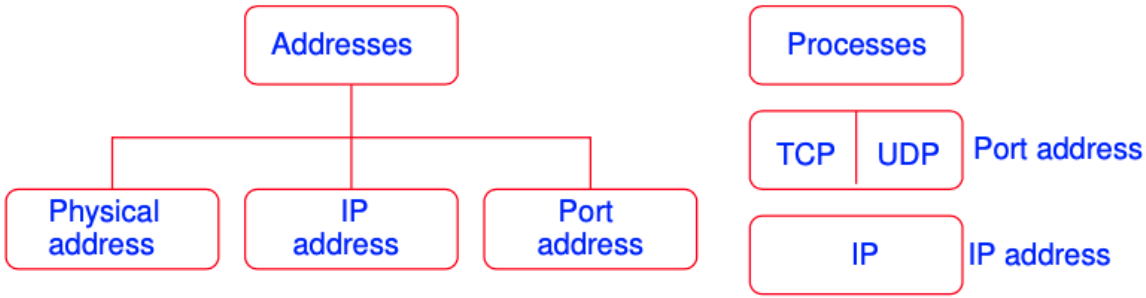


Some of the IP options

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

IP Addresses

Address Structure

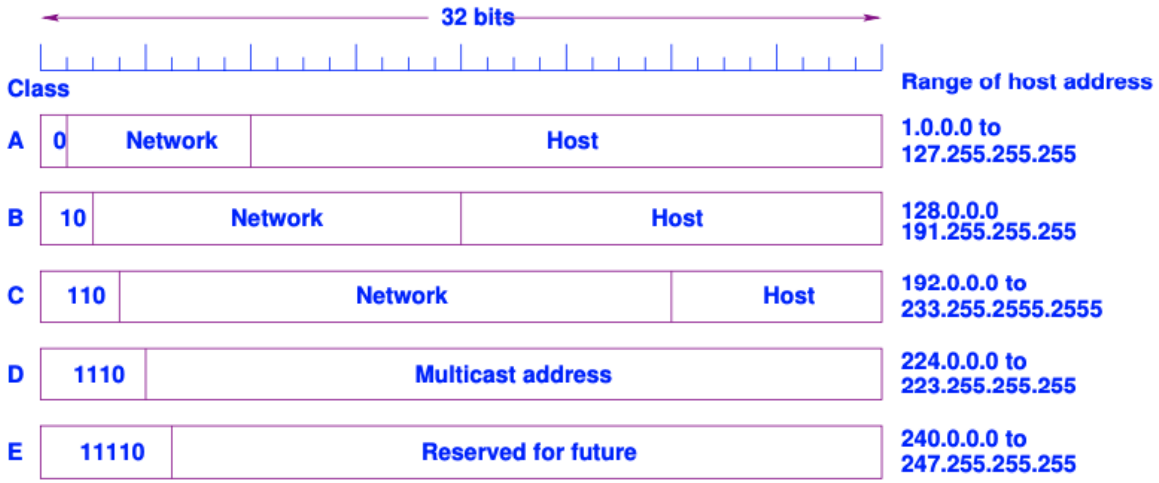


- Properties
  - Should be globally unique
  - Hierarchical: network + host
- Notation/Interpretation
  - Class-based
    - 10.3.2.4
    - 128.96.33.81
    - 192.12.69.77
  - Classless InterDomain Routing (CIDR) notation

Class-based Addresses

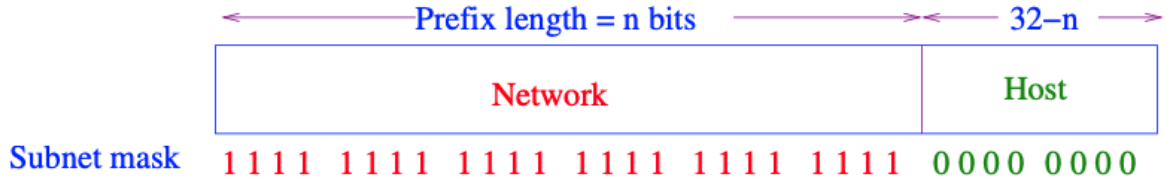
- An IP address contains 32 bits of data
- The leftmost four (4) bits determine its class ⇒ non uniformly

Class	Leftmost bits	Start addresses	Finish addresses
A	0xxx	0.0.0.0	127.255.255.255
B	10xx	128.0.0.0	191.255.255.255
C	110x	192.0.0.0	223.255.255.255
D	1110	224.0.0.0	239.255.255.255
E	1111	240.0.0.0	255.255.255.255



Classless InterDomain Routing (CIDR) notation

- Variable Length Subnet Masking $\Rightarrow$ x.y.z.t/n



- 192.168.100.14/24 represents the IPv4 address 192.168.100.14 with subnet mask is 255.255.255.0 which has 24 leading 1-bits associated routing prefix, 192.168.100.0
- $n = 1 \Rightarrow$ 0 1000000 00000000 00000000 00000000 $\Rightarrow$ 128 class C
- $n = 6 \Rightarrow$ 0 1100000 00000000 00000000 00000000 $\Rightarrow$ 192 class C

/n	Mask	/n	Mask	/n	Mask	/n	Mask
/1	128.0.0.0	/9	255.128.0.0	/17	255.255.128.0	/25	255.255.255.128
/2	192.0.0.0	/10	255.192.0.0	/18	255.255.192.0	/26	255.255.255.192
/3	224.0.0.0	/11	255.224.0.0	/19	255.255.224.0	/27	255.255.255.224
/4	240.0.0.0	/12	255.240.0.0	/20	255.255.240.0	/28	255.255.255.240
/5	248.0.0.0	/13	255.248.0.0	/21	255.255.248.0	/29	255.255.255.248
/6	252.0.0.0	/14	255.252.0.0	/22	255.255.252.0	/30	255.255.255.252
/7	254.0.0.0	/15	255.254.0.0	/23	255.255.254.0	/31	255.255.255.254
/8	255.0.0.0	/16	255.255.0.0	/24	255.255.255.0	/32	255.255.255.255

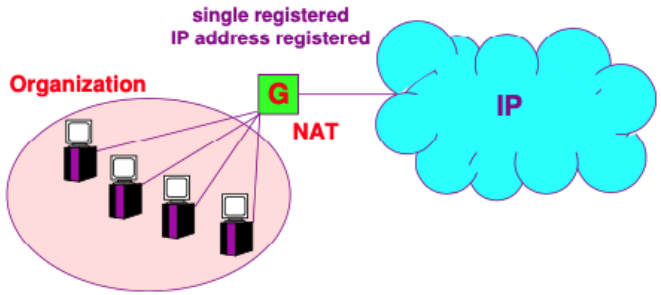
Private Networks and Subnets

- IP has reserved certain networks for internal use

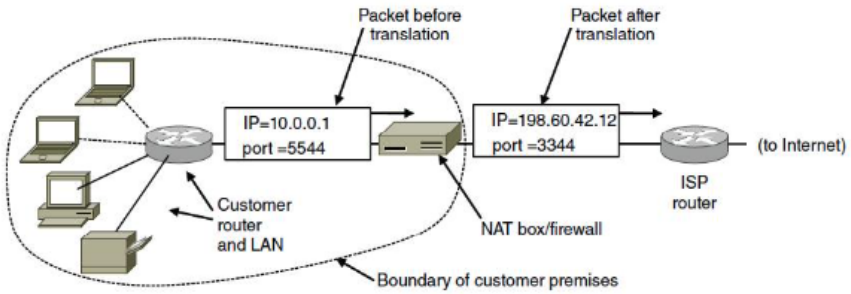
Network address	Default mask
10.0.0.0	255.0.0.0
172.16.0.0	255.240.0.0
192.168.0.0	255.255.0.0

Private Networks and Subnets

- Many local computers with many local addresses, but one public, registered address for the entire organization



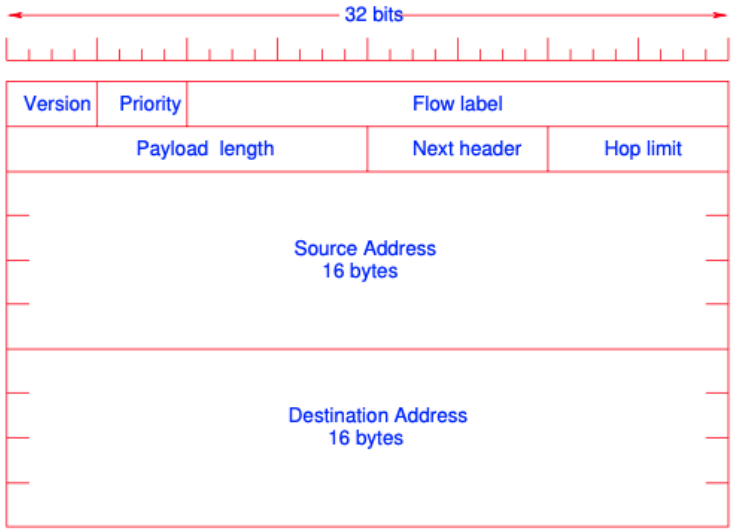
- Allows hosts within private (non-routable) addresses to talk to the global Internet.
  - Replaces SrcAddr with G for outgoing
  - Replaces DstAddr with correct host address
  - Uses a translation table to convert G
  - The table is a cache



IPv6

- IPv6 Goals
  - Supports billions of hosts (flat or class)
  - Reduces the size of the routing tables
  - Simplifies the protocol $\Rightarrow$ process packets faster
  - Provides better security/authentication/privacy
  - Supports more ToS, particularly for real-time data
  - Aids multicasting by allowing scopes to be specified
  - Supports roaming without changing address
  - Allows the protocol to evolve in the future
  - Permits old/new protocols to coexist
- Major Features
  - 128-bit addresses
  - Multicast
  - Real-time service
  - Authentication and security
  - Auto-configuration
  - End-to-end fragmentation
  - Protocol extensions

IPv6 Header



The IPv6 fixed header

- 40-byte header, fixed (required)
- Version field: 6 for IPv6 and 4 for IPv4
- Priority field: distinguishes flow controlled packets
  - 0-7: slowing down in the event of congestion
    0: less important; news:1, FTP:4, telnet:6, ..
  - 8-15 are for real-time traffic(audio, video)
- Payload length: # of bytes follow the header
- Next header: tells which of the (currently) six extension headers follows this one.
  If this header is IP header the next header could be TCP, UDP, etc.

IPv6 Header

- Hop limit: keeps packets from living forever
- Source/Destination: 16-byte addresses
  - Addresses with 80 0s are reserved for IPv4
  - Separate prefixes are assigned to different ISPs
- Extension header
  - fragmentation
  - source routing
  - authentication and security
  - other options
- Extension Headers

Extension header	Description
Hop-by-hop options	Miscellaneous information for routers
Routing	Full or partial route to follow
Fragmentation	Management of datagram fragments
Authentication	Verification of the sender's identity
Encrypted payload	Information about encrypted contents
Destination options	Additional information for the destination

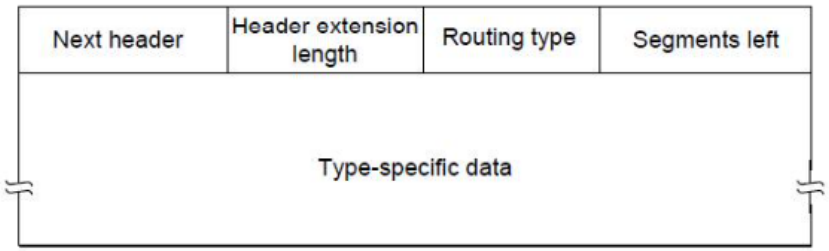
IPv6 Header

- IPv6 Addresses
  - Notation: x:x:x:x:x:x:x:x (x = 16-bit hex number)

- The hop-by-hop extension header for large datagrams
  - jumbograms, beyond 65,535 bytes
  - No fragmentation

Next header	0	194	4
Jumbo payload length			

- The extension header for routing



Label Switching

- Routers establish connections
- Add a connection-ID to datagrams
  - Routers use the ID to choose outgoing interfaces
- Very closely to virtual circuits *How?*

Question 5.4

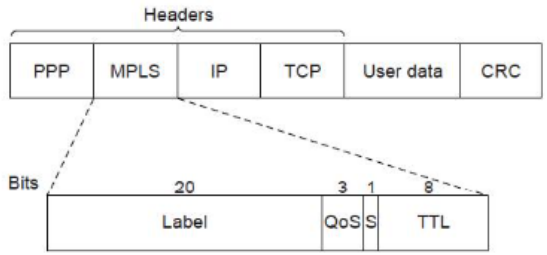
What is the difference between label switching and virtual circuit?

Answer:

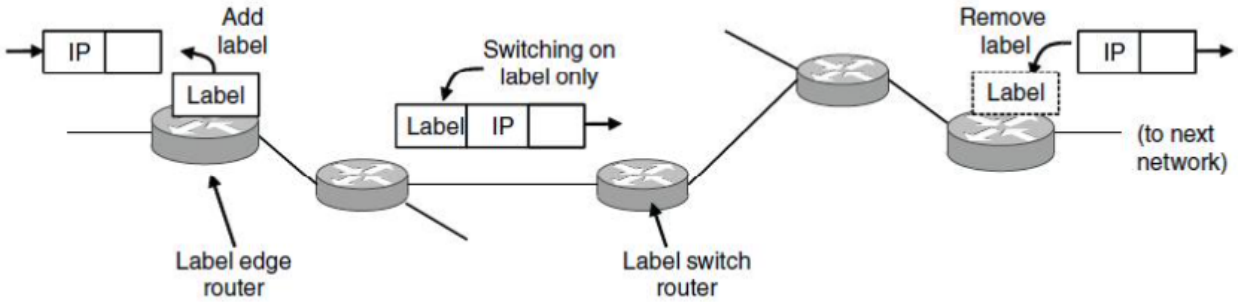
- *In VC, there is a setup phase for each connection*
- *In LS, routes are data (packet) driven*
  - *Label are created by routers at boot time*

Label Switching

- Internet Engineering Task Force (IETF) this under MultiProtocol Label Switching (MLPS)
- Transmitting a TCP segment using IP, MPLS, and Point-to-Point Protocol (PPP)



- Forwarding an IP packet through an MPLS network



Outline I

References

[Tanenbaum and Wetherall, 2011] Tanenbaum, A. S. and Wetherall, D. J. (2011). Computer Networks: 5th Edition. Prentice Hall PTR.