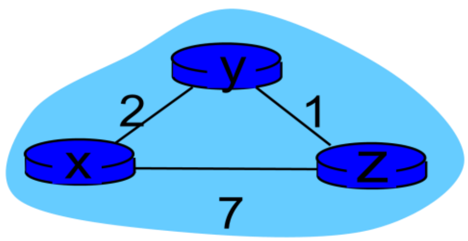
Scheme Of Evaluation
Internal Assessment Test 2 – October.2019

Sub:	Computer Networks							**Code:**	17CS52
Date:	12 / 10 / 2019	Duration:	90mins	Max Marks:	50	**Sem:**	VI	**Branch:**	ISE

Note: Answer Any Five Questions

Question #		Description	Marks Distribution		Max Marks
1	a)	What is routing? Explain the structure of a router. • Routing definition • Architecture	2M 8M	10M	10 M
2	a)	Interpret how connection is established and tear down in TCP. • Connection establishment • Termination	2.5M 2.5M	5M	10 M
	b)	Label TCP segment structure • Message Format	5M	5M	
3	a)	Discuss the path attributes in BGP & steps to select the BGP routes. • Path attributes • Selection policy	2M 4M	6M	10 M
	b)	List the advancements made in OSPF. • Advancements	4M	4M	
4	a)	Discuss the working of RIP protocol with example. • RIP protocol working • Example	5M 5M	10M	10 M
5	a)	Describe multicast routing. • Definition & Introduction • Algorithms	4M 6M	10M	10 M

6	a)	Summarize how controlled flooding is implemented in broadcast • Sequence Number flooding • Reverse path forwarding	3M 3 M	6M	10 M
	b)	Label IPv6 packet format. • Packet format	4M	4M	
7	a)	Consider the following network. With the indicated link costs, use distance vector algorithm to compute the shortest path between all network nodes.  • Solving	10M	10M	10 M
8	a)	Summarize the working of TCP congestion control. • Introduction • Phases	2M 8M	10M	10 M

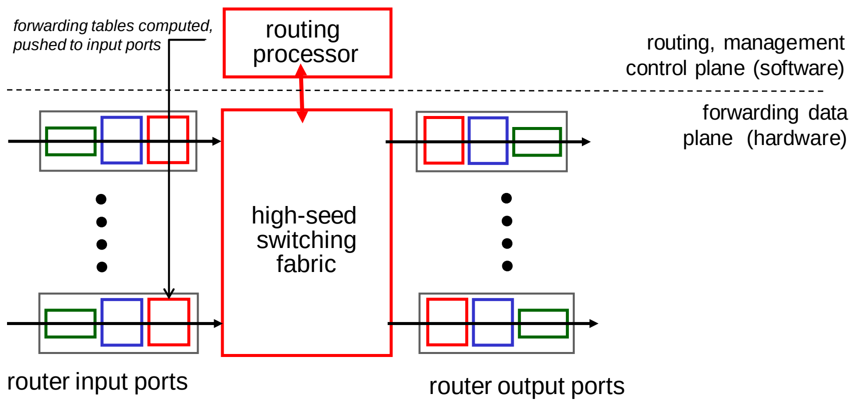
Answers

1 (a) What is routing? Explain the structure of a router.

Router architecture overview

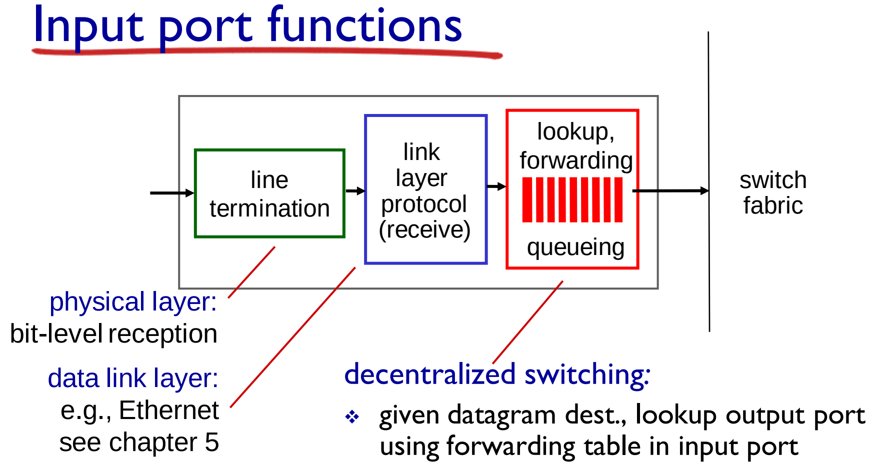
two key router functions:

- run routing algorithms/protocol (RIP, OSPF, BGP)
- *forwarding* datagrams from incoming to outgoing link



Network Layer 4-8

Input port functions



decentralized switching:

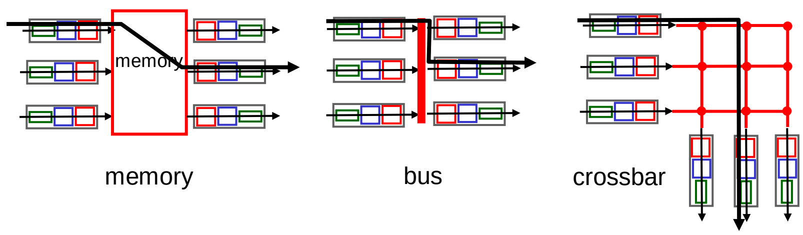
- given datagram dest., lookup output port using forwarding table in input port memory *("match plus action")*
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

Network Layer 4-9

Switching fabrics

- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs

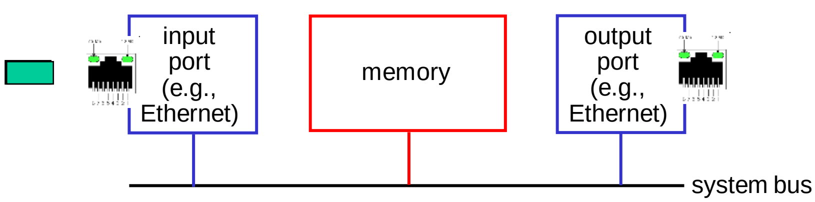
- three types of switching fabrics



Switching via memory

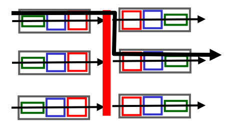
first generation routers:

- traditional computers with switching under direct control of CPU(routing processor)
- packet copied to system' s memory
- only one packet can be send at a time



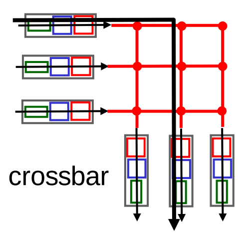
Switching via a bus

- datagram from input port memory to output port memory via a shared bus
- *if multiple packets arrives they have to wait as one packet can cross the bus*

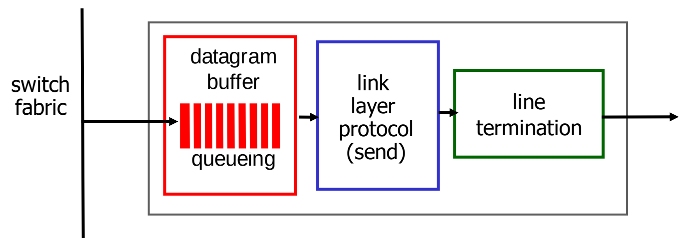


Switching via interconnection network

- overcome bus bandwidth limitations
- a cross bar switch is an interconnection network consisting of 2N buses that connect N input ports to N output ports.
- capable of forwarding multiple packets in parallel.



Output ports

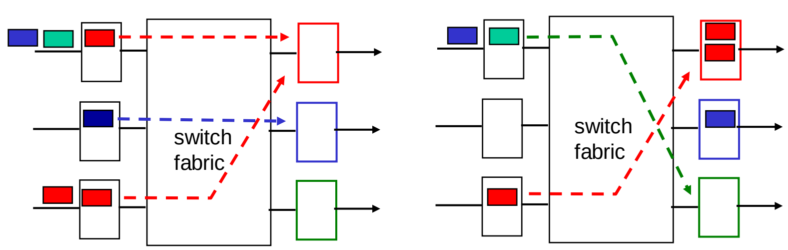


- *buffering* required from fabric faster rate
- *scheduling* datagrams for transmission

Datagram (packets) can be lost due to congestion, lack of buffers

Priority scheduling

Output port queueing



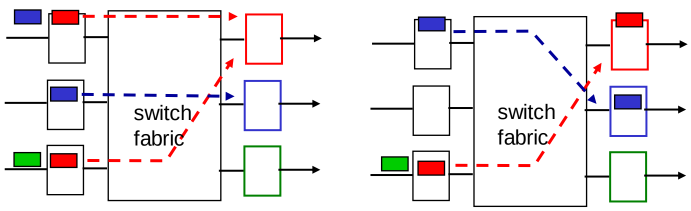
at *t*, packets more from input to output

one packet time later

- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

Input port queuing

- fabric slower than input ports combined -> queueing may occur at input queues
  - *queueing delay and loss due to input buffer overflow!*
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward



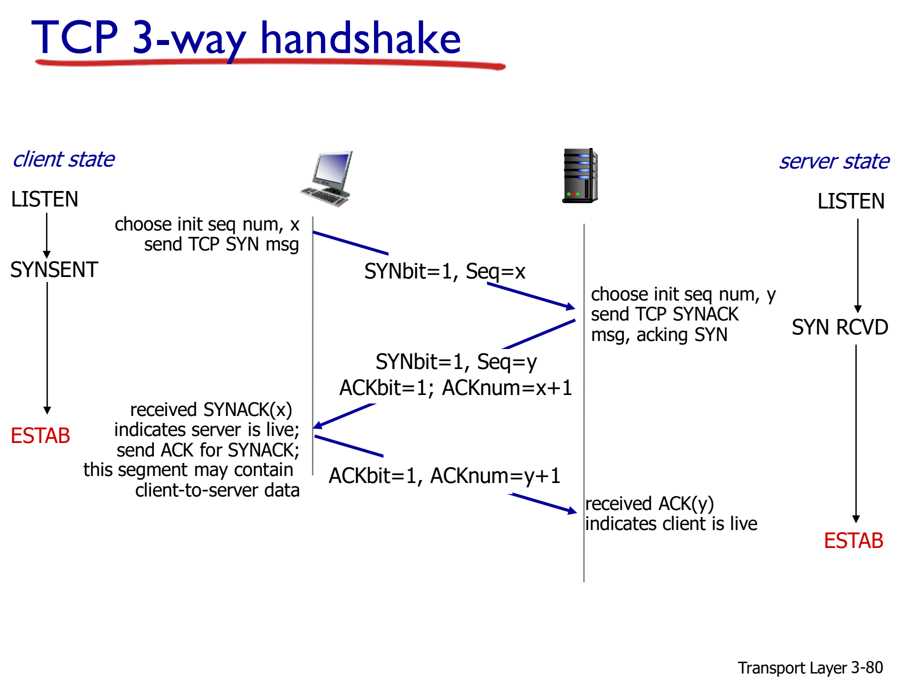
output port contention: only one red datagram can be transferred. *lower red packet is blocked*

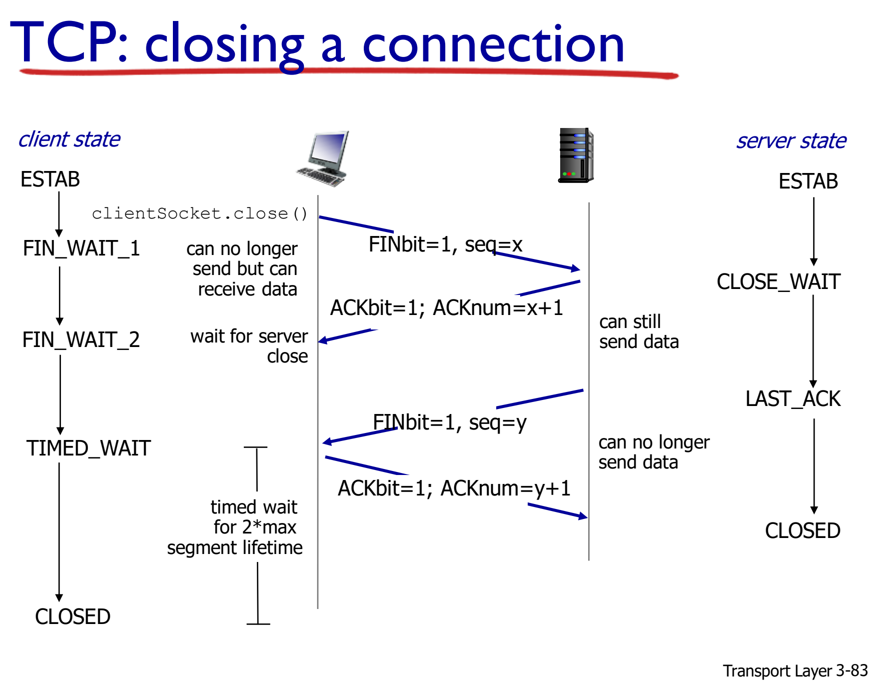
one packet time later: green packet experiences HOL blocking

Packet Scheduling

- O/P port will choose packets for transmission based on FCFS or Weighted Fair Queuing
- If there is no more space in buffer
- Decision= either drop the arriving packet(drop tail) else remove already queued packet
- It's better tp drop a packet before the buffer is full(Random early detection)

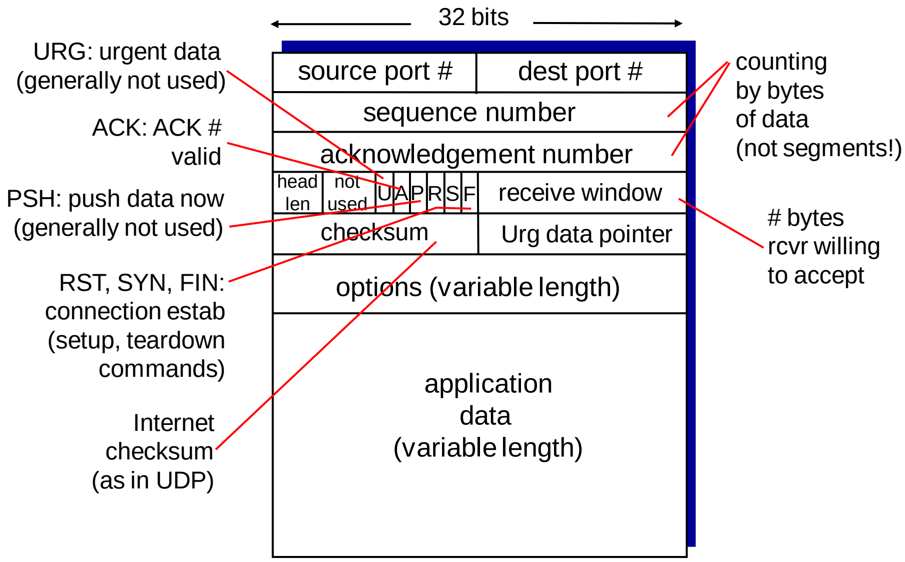
2. a. Interpret how connection is established and tear down in TCP.





2.b. Label TCP segment structure

TCP segment structure



3. a. Elaborate the path attributes in BGP & steps to select the BGP routes.

Path attributes and BGP routes

- advertised prefix includes BGP attributes
  - prefix + attributes = "route"
- two important attributes:
  - AS-PATH: contains ASs through which prefix advertisement has passed: e.g., AS 67, AS 17
  - NEXT-HOP: indicates specific internal-AS router to next-hop AS. (may be multiple links from current AS to next-hop-AS)
- gateway router receiving route advertisement uses import policy to accept/decline
  - e.g., never route through AS x
  - *policy-based* routing

BGP route selection

- router may learn about more than 1 route to destination AS, selects route based on:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria

3. b. List the advancements made in OSPF.

OSPF "advanced" features (not in RIP)

- *security:* all OSPF messages authenticated (to prevent malicious intrusion)
- multiple same-cost paths allowed (only one path in RIP)
- for each link, multiple cost metrics for different TOS (e.g., satellite link cost set "low" for best effort ToS; high for real time ToS)
- integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- hierarchical OSPF in large domains.

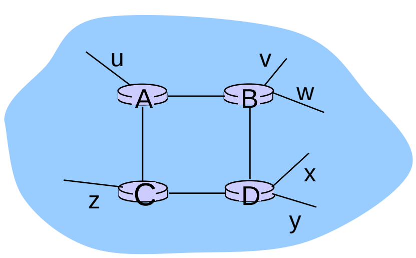
Hierarchical OSPF

- *two-level hierarchy:* local area, backbone.
  - link-state advertisements only in area
  - each nodes has detailed area topology; only know direction (shortest path) to nets in other areas.
- *area border routers:* "summarize" distances to nets in own area, advertise to other Area Border routers.
- *backbone routers:* run OSPF routing limited to backbone.
- *boundary routers:* connect to other AS' s.

4. a. Discuss the working of RIP protocol with example.

RIP (Routing Information Protocol)

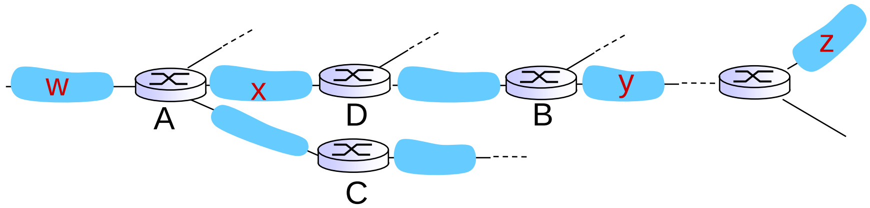
- included in BSD-UNIX distribution in 1982
- distance vector algorithm
  - distance metric: # hops (max = 15 hops), each link has cost 1
  - DVs exchanged with neighbors every 30 sec in response message (aka advertisement)
  - each advertisement: list of up to 25 destination subnets *(in IP addressing sense)*



from router A to destination *subnets:*

subnet	hops
u	1
v	2
w	2
x	3
y	3
z	2

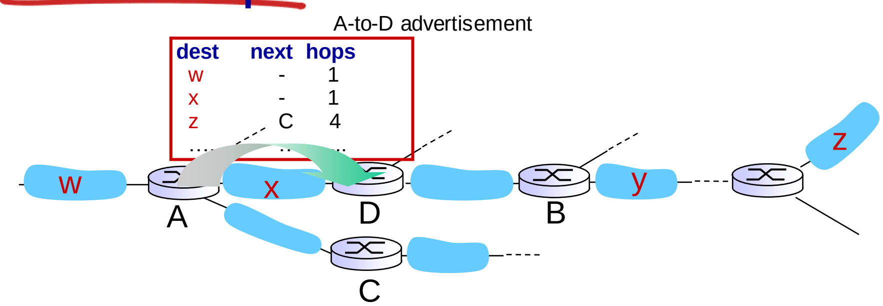
RIP: example



routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B	7
x	--	1
….	….	….

RIP: example



routing table in router D

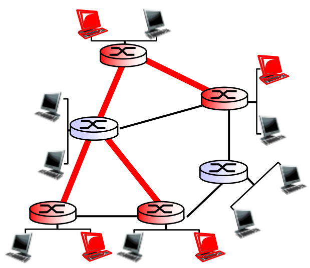
destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
….	….	….

5\. a. Describe multicast routing.

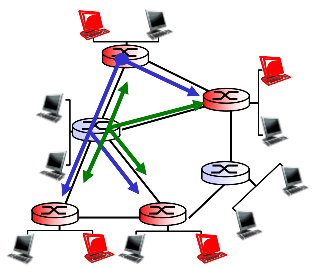
Multicast routing: problem statement

goal: find a tree (or trees) connecting routers having local mcast group members

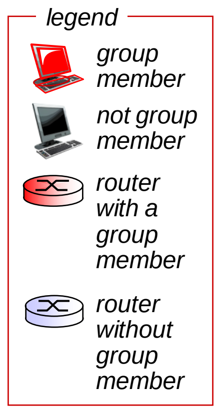
- *tree:* not all paths between routers used
- *shared-tree:* same tree used by all group members
- *source-based:* different tree from each sender to rcvrs



shared tree



source-based trees



Approaches for building mcast trees

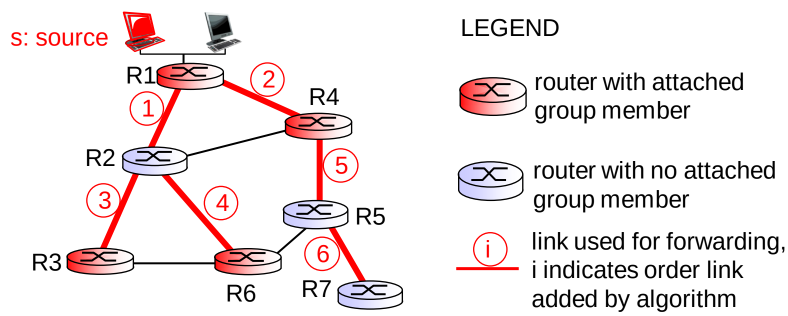
approaches:

- *source-based tree:* one tree per source
  - shortest path trees
  - reverse path forwarding
- *group-shared tree:* group uses one tree
  - minimal spanning (Steiner)
  - center-based trees

…we first look at basic approaches, then specific protocols adopting these approaches

Shortest path tree

- mcast forwarding tree: tree of shortest path routes from source to all receivers
  - Dijkstra's algorithm

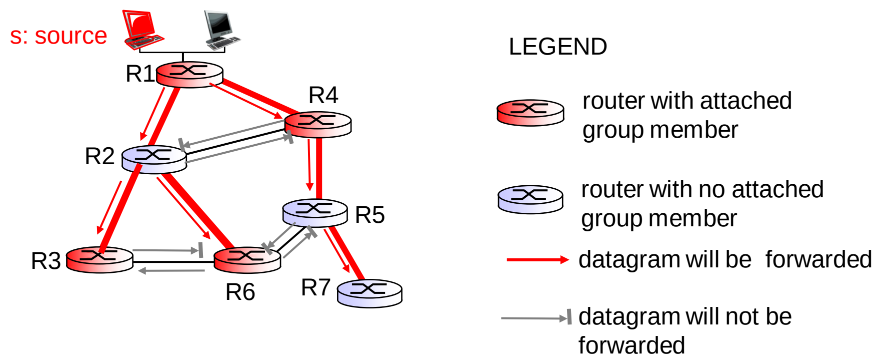


Reverse path forwarding

- rely on router's knowledge of unicast shortest path from it to sender
- each router has simple forwarding behavior:

if (mcast datagram received on incoming link on shortest path back to center)
then flood datagram onto all outgoing links
else ignore datagram

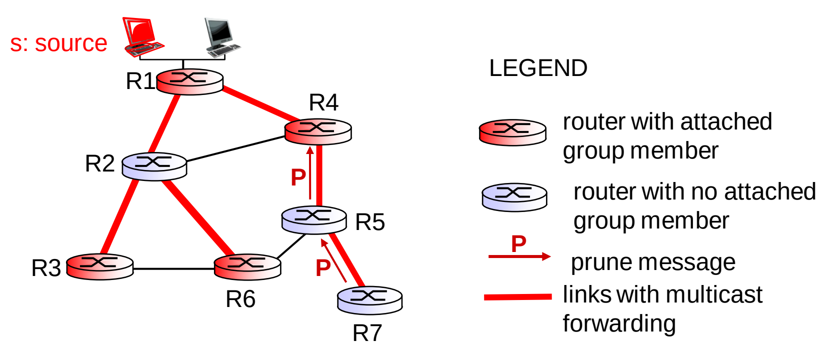
Reverse path forwarding: example



- result is a source-specific *reverse* SPT
  - may be a bad choice with asymmetric links

Reverse path forwarding: pruning

- forwarding tree contains subtrees with no mcast group members
  - no need to forward datagrams down subtree
  - "prune" msgs sent upstream by router with no downstream group members



Shared-tree: steiner tree

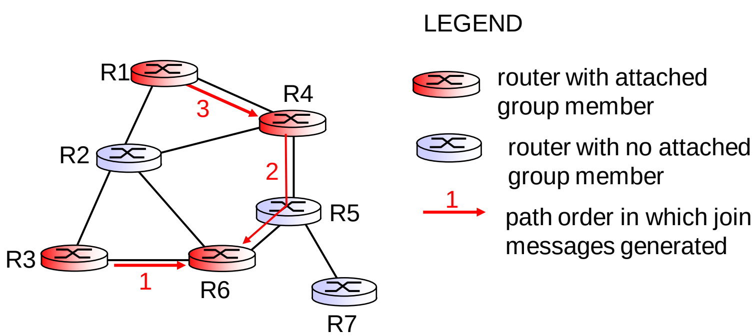
- *steiner tree:* minimum cost tree connecting all routers with attached group members
- problem is NP-complete
- excellent heuristics exists
- not used in practice:
  - computational complexity
  - information about entire network needed
  - monolithic: rerun whenever a router needs to join/leave

Center-based trees

- single delivery tree shared by all
- one router identified as *"center"* of tree
- to join:
  - edge router sends unicast *join-msg* addressed to center router
  - *join-msg* "processed" by intermediate routers and forwarded towards center
  - *join-msg* either hits existing tree branch for this center, or arrives at center
  - path taken by *join-msg* becomes new branch of tree for this router

Center-based trees: example

suppose R6 chosen as center:



Internet Multicasting Routing: DVMRP

- DVMRP: distance vector multicast routing protocol, RFC1075
- *flood and prune:* reverse path forwarding, source-based tree
  - RPF tree based on DVMRP's own routing tables constructed by communicating DVMRP routers
  - no assumptions about underlying unicast
  - initial datagram to mcast group flooded everywhere via RPF
  - routers not wanting group: send upstream prune msgs

DVMRP: continued…

- *soft state:* DVMRP router periodically (1 min.) "forgets" branches are pruned:
  - mcast data again flows down unpruned branch
  - downstream router: reprune or else continue to receive data
- routers can quickly regraft to tree
  - following IGMP join at leaf
- odds and ends
  - commonly implemented in commercial router

PIM: Protocol Independent Multicast

- not dependent on any specific underlying unicast routing algorithm (works with all)
- two different multicast distribution scenarios :

dense:

- group members densely packed, in "close" proximity.
- bandwidth more plentiful

sparse:

- # networks with group members small wrt # interconnected networks
- group members "widely dispersed"
- bandwidth not plentiful

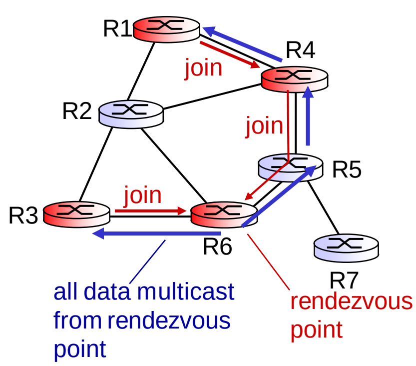
PIM- dense mode

flood-and-prune RPF: similar to DVMRP but…

- underlying unicast protocol provides RPF info for incoming datagram
- less complicated (less efficient) downstream flood than DVMRP reduces reliance on underlying routing algorithm
- has protocol mechanism for router to detect it is a leaf-node router

PIM - sparse mode

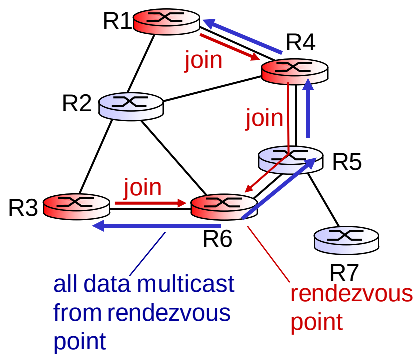
- center-based approach
- router sends *join* msg to rendezvous point (RP)
  - intermediate routers update state and forward *join*
- after joining via RP, router can switch to source-specific tree
  - increased performance: less concentration, shorter paths



PIM - sparse mode

sender(s):

- unicast data to RP, which distributes down RP-rooted tree
- RP can extend mcast tree upstream to source
- RP can send *stop* msg if no attached receivers
  - "no one is listening!"



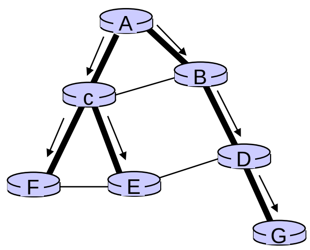
6.a. Summarize how controlled flooding is implemented in broadcast routing

In-network duplication

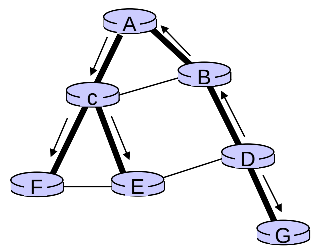
- *flooding:* when node receives broadcast packet, sends copy to all neighbors
  - problems: cycles & broadcast storm
- *controlled flooding:* node only broadcasts pkt if it hasn't broadcast same packet before
  - node keeps track of packet ids already broadcasted
  - or reverse path forwarding (RPF): only forward packet if it arrived on shortest path between node and source
- *spanning tree:*
  - no redundant packets received by any node

Spanning tree

- first construct a spanning tree
- nodes then forward/make copies only along spanning tree



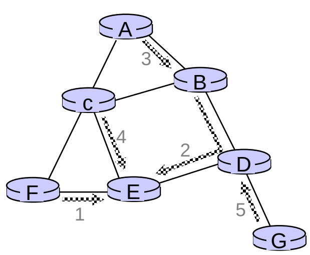
(a) broadcast initiated at A



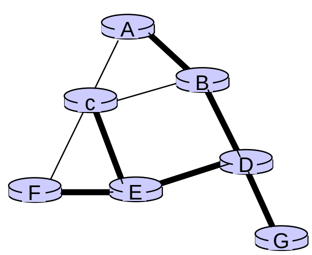
(b) broadcast initiated at D

Spanning tree: creation

- center node
- each node sends unicast join message to center node
  - message forwarded until it arrives at a node already belonging to spanning tree



(a) stepwise construction of spanning tree (center: E)

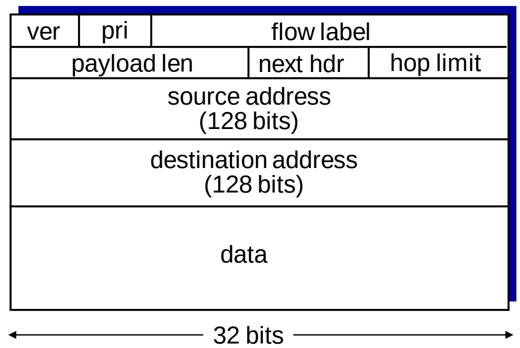


(b) constructed spanning tree

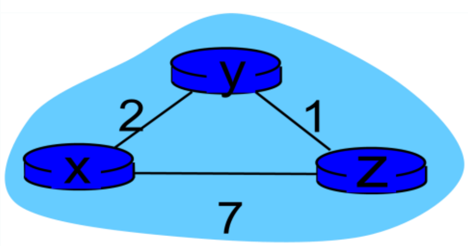
6. b. Label IPv6 packet format.

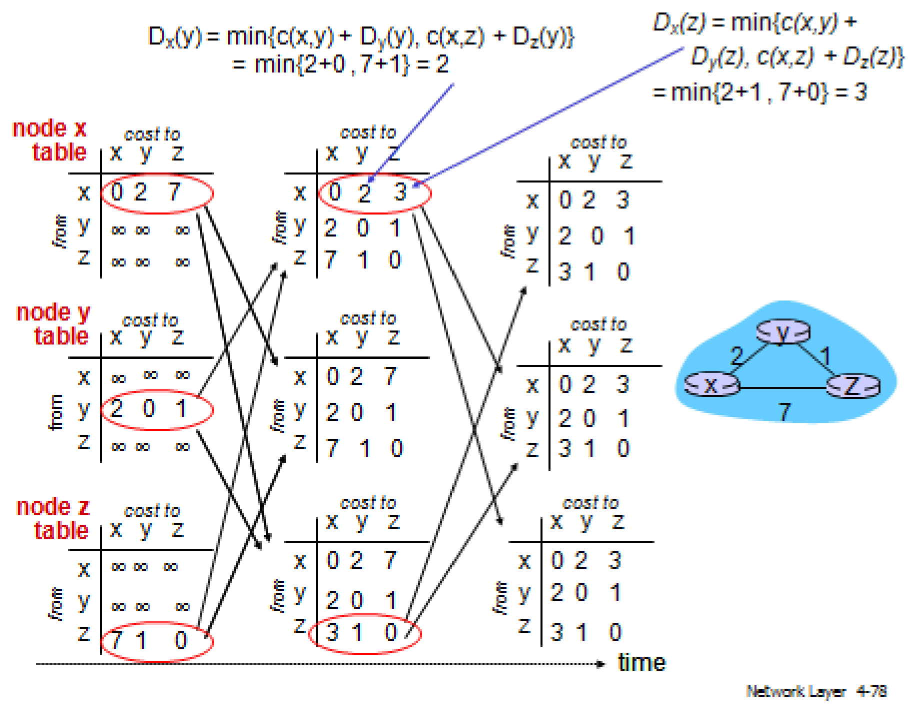
IPv6 datagram format

priority: identify priority among datagrams in flow
flow Label: identify datagrams in same "flow."
(concept of "flow" not well defined).
next header: identify upper layer protocol for data



7. a. Consider the following network. With the indicated link costs, use distance vector algorithm to compute the shortest path between all network nodes.



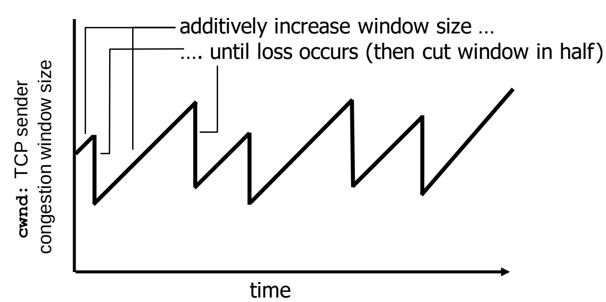


8.a. TCP Congestion control

TCP congestion control: additive increase multiplicative decrease

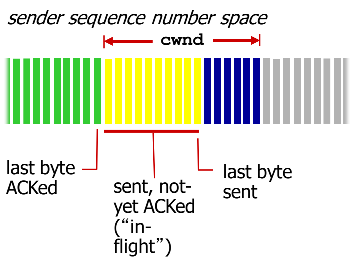
- *approach:* sender increases transmission rate (window size), probing for usable bandwidth, until loss occurs
  - *additive increase:* increase **cwnd** by 1 MSS every RTT until loss detected
  - *multiplicative decrease:* cut **cwnd** in half after loss

AIMD saw tooth behavior: probing for bandwidth



Transport Layer 3-99

TCP Congestion Control: details



- sender limits transmission:

```
LastByteSent-
  LastByteAcked   ≤  cwnd
```

- **cwnd** is dynamic, function of perceived network congestion

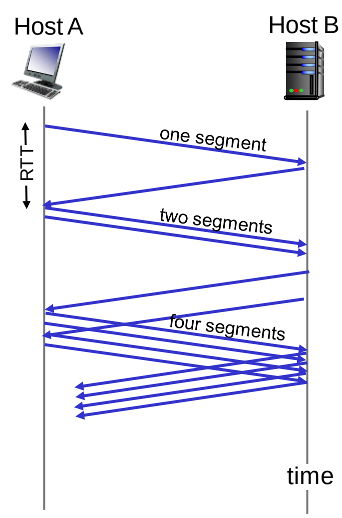
TCP sending rate:

- *roughly:* send cwnd bytes, wait RTT for ACKS, then send more bytes

$$\text{rate} \approx \frac{\text{cwnd}}{\text{RTT}} \text{ bytes/sec}$$

TCP Slow Start

- when connection begins, increase rate exponentially until first loss event:
  - initially **cwnd** = 1 MSS
  - double **cwnd** every RTT
  - done by incrementing **cwnd** for every ACK received
- *summary:* initial rate is slow but ramps up exponentially fast



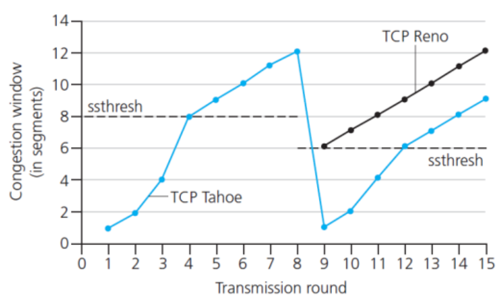
TCP: detecting, reacting to loss

- loss indicated by timeout:
  - **cwnd** set to 1 MSS;
  - window then grows exponentially (as in slow start) to threshold, then grows linearly
- loss indicated by 3 duplicate ACKs: TCP RENO
  - dup ACKs indicate network capable of delivering some segments
  - **cwnd** is cut in half window then grows linearly
- TCP Tahoe always sets **cwnd** to 1 (timeout or 3 duplicate acks)

TCP: switching from slow start to CA

Q: when should the exponential increase switch to linear?

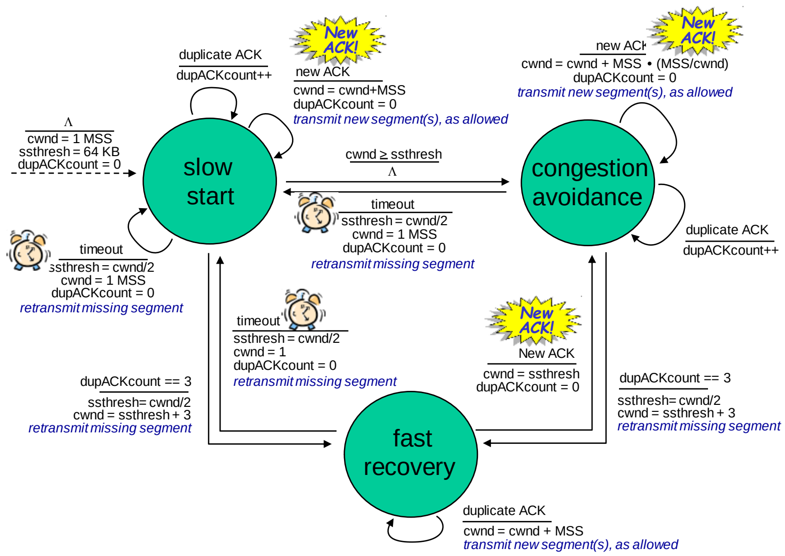
A: when **cwnd** gets to 1/2 of its value before timeout.



Implementation:

- variable **ssthresh**
- on loss event, **ssthresh** is set to 1/2 of **cwnd** just before loss event

Summary: TCP Congestion Control



b). If the classes 5a & 5b, want to be accomodated in a single classroom, then the class room will be congested. In order to overcome this, TCP congestion control mechanism can be followed.

At first. the students are asked to stand outside the door, at first then 5 students are asked to enter the classroom (slow start), if there is no congestion, then 10 students, then 20 students, 40 students are allowed upto (80 students ≈ threshold). Then students are allowed one by one linearly until congestion occurs.

If congestion occurs then the students count before congestion is taken as threshold value & again slow start starts until congestion. The remaining students can be accomodated in some laboratory if needed.