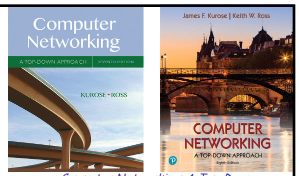
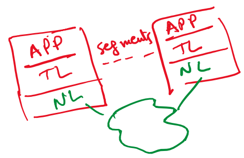
CS 655
Computer Networks

Abraham Matta
Computer Science
Boston University

Chapters 4 & 5
Routing Services



Computer Networking: A Top-Down Approach, 7th or 8th editions. Jim Kurose, Keith Ross. Pearson.

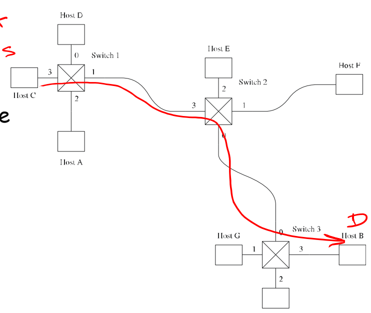


1

Wide Area Networks

network packet
segment S

- A WAN covers large geographic area
- Switches are connected by long-haul point-to-point links
- Hosts connect to switches



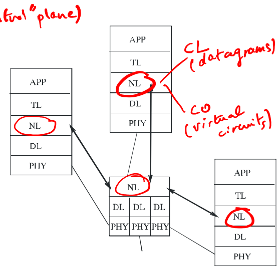
2

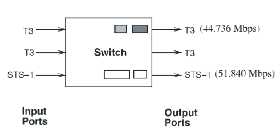
WANs

1) ROUTING ("Control" plane)
2) FORWARDING ("data" plane)

- Network layer deals with packet routing: the process of selecting paths/routes (path = sequence of switches and links) for the packets to take to reach their destination

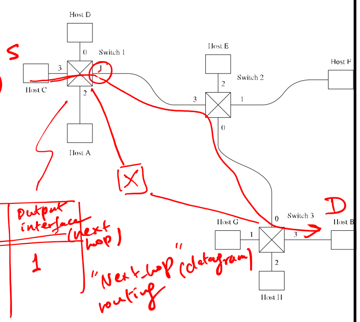
- A WAN has a maximum packet size to bound packet delays; may need fragmentation/reassembly at the higher layer





3

Datagrams



packet (datagram)

| segment

header

destination address

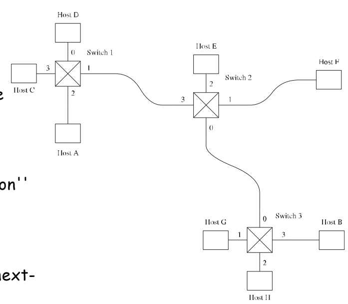
Forwarding Table

Dest Address	Output interface (next hop)
D	1

"Next-hop" (datagram) routing

4

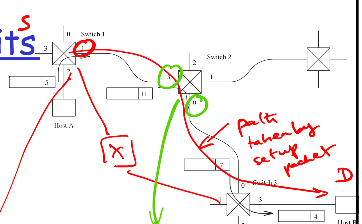
Datagrams



- No connection setup phase
- Each packet forwarded independently based on destination address
- Each packet in ``connection'' may follow different path
- Analogy: postal system
- Sometimes called *connectionless* model (*or* next-hop routing)
- Each switch maintains a forwarding (routing) table: {destination address, output port (or next-hop)}

5

Virtual Circuits



destination address

d

set-up packet

next-hop routed

VC Tables

data packet

VCI #

VCI	output interface
~~X~~	1

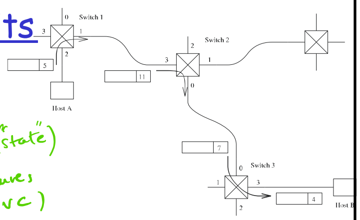
global vc id

vc in	input interface	vc out	o/p
x	3	y	0

local (link-level) VCI ids

6

Virtual Circuits

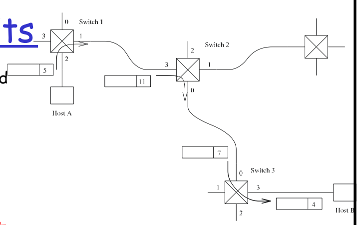


- wait for RTT
- extra VC tables ("state")
- not resilient to failures (re-establish VC)

+ in-order delivery
+ performance guarantees

7

Virtual Circuits



- Explicit connection setup (and tear-down) phase
- Setup packet is routed using next-hop routing
- Subsequent packets follow same route, thus arrive in order
- Analogy: phone call
- Sometimes called ***connection-oriented*** model (*or* VC routing)
- Each switch maintains a VC table: {input VCI, input port, output VCI, output port}
- VCI in header is interchanged by nodes as they forward data packet

8

Virtual Circuits versus Datagrams

Virtual Circuit Model:

- Typically wait full RTT for connection setup before sending first data packet
- While the connection request contains the full address of destination, each data packet contains only a small identifier, making the per-packet header overhead small
- Need per-connection state at switches
- If a switch or a link in a connection fails, the connection is broken and a new one needs to be established
- Connection setup provides an opportunity to reserve resources (to guarantee QoS)

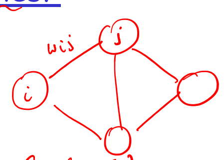
9

Virtual Circuits versus Datagrams (cont'd)

Datagram Model:

- There is no round trip time delay waiting for connection setup; a host can send data as soon as it is ready
- Since every packet must carry the full address of the destination, the overhead per packet is higher than for the connection-oriented model
- No connection state in switches
- Since packets are treated independently, it is easier to route around link and node failures
- Source host has no way of knowing if the network is capable of delivering a packet or if the destination host is even up

How do Switches Maintain Routing Tables?



$G = (V, E)$

$|V| = N$

Dest address	next hop
?	⋮

Find P s.t. $W(P) = \sum_{ij \in P} w_{ij}$ is min

$$w_{ij} = \begin{cases} 1 & \text{min\_hop routing} \\ delay_{ij} & \text{min\_delay routing} \\ cost_{ij} & \text{min\_cost routing} \\ \vdots & \end{cases}$$

How do Switches Maintain Routing Tables? (cont'd)

Link-state (SPF -- Shortest Path First)

Link State (Dijkstra's) — Distance Vector (Bellman-Ford)

$D_i \triangleq$ shortest distance from s to i

$M = \{s\}$

$D_s = 0$

$D_{i \neq s} = w_{si}$ [$w_{sk} = \infty$ if k is not neighbor of s]

$O(N^2)$

① Pick node with min distance

$D_k = \min_{i \notin M} \{D_i\}$

$M = M \cup \{k\}$ ← if $|M| = N$, stop

② Update distances

$D_i = \min\{D_i, D_k + w_{ki}\}$ $\forall i \notin M$

goto ①

How do Switches Maintain Routing Tables? (cont'd)

Link-state (SPF -- Shortest Path First)

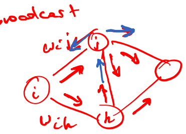
periodically broadcast

seq no

i's LSP

| w_{ij} | w_{ik} |

LSP looping



13

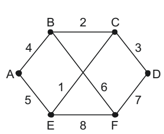
How do Switches Maintain Routing Tables? (cont'd)

Link-state (SPF -- Shortest Path First)

- Each node maintains a complete view of the topology: each node periodically broadcasts to all nodes (not just neighbors) the costs/status of its outgoing links (not entire routing table)
  - Link State Packet (LSP)
    - id of the node that created the LSP
    - cost of link to each directly connected neighbor
    - sequence number (SEQNO)
    - age or time-to-live (TTL) for this packet
- Reliable Flooding
  - store most recent LSP from each node
  - forward LSP to all neighbor 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
- When a link state message arrives and the view changes accordingly, the node recomputes routes by applying ``**Dijkstra's shortest path algorithm**''

14

LS Propagation Protocol



(a)

Link State Packets

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

(b)

15

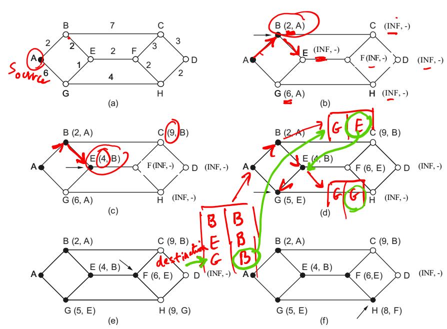
Route Calculation in Link State

- Dijkstra's shortest path algorithm
- N denotes set of nodes in the graph
- l(i,j) denotes non-negative cost (weight) for edge (i,j)
- s in N 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 (M != N)
        M = M U {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))
```


16

Dijkstra's Algorithm in action



17