Lecture - 1

04-06-23

Chapter - 1

Introduction

Book: Computer Networks Tanenbaum 6th edition

Uses
- → Business
- → Home
- → Mobile
- → Social

Client Server Model

Client ↔ Network ↔ Server

Client ↔ Network

Home network
- → Access to remote
- → person " person / peer to peer (P2P)

~~Access to~~

Wireless but not mobile → Directed network to a enclosed area

Not wireless but mobile → Notebook computer / Laptops using LAN port in hotel rooms

The Internet → World wide Web

internet → connection between multiple networks

Lecture - 2 05-06-23

LAN:

Two broadcast networks

a) Bus

b) Ring

Metropolitan Area Network:

Wide Area Network:

Routers creates routes

Network of routers → subnet

If connection path btwn. two host is fixed → connection oriented network

" " " " " " is not fixed → connectionless network

* Synchronization done using metadata in packets
* Logical connection done using code in routers

Wireless networks:

a) Bluetooh Configuration

b) Wireless LAN

Flying LAN

a) Individual mobile connectivity

b) One outworld connectivity

Home Network Categories:

Network Software:

Protocol Hierarcies

Design Issues for the Layers

Connection Oriented and connectionless Services

** Protocols help modularize the process

Each layer provides certain service

Layer to layer communicates through interface

Sender and Reciever have same amount of layers

Same layer P2P connection done through protocols.

* Higher layer protocols are indirect communication
* Only base layer protocols are direct "

Protocol Hierarchies:

Data chunks can be broken into layer capacity.

Design Issues for layers:

Addressing
Error control
Flow control
Multiplexing
Routing

Reliable / Acknowledged → Feedback of reception
Unreliable → No " " "

* Reliable and acknowledged are similar not synanomous
If ~~sender~~ receiver does not recieve and sender keeps resending until successful acknowledgement ⇒ reliable.

Lecture - 3 06-06-23

Protocol Hierarchy → Protocol stack

Multi layer format → Increase modularity

Each layer in sender adds headers/trailers..

" " " receiver strips " "

Eeach layer has a fixed data width → So data bits might be split to accomodate.

Service Primitives:

(i) Listen
(ii) Connect
(iii) Receive
(iv) Send
(v) Disconnect

* Baseline raw coding is in OS
* We use system calls to access OS kernel
* We have protocol stack to accomodate layer stack convention.

Services to Protocol Relation:

(i) Service → what a layer does

(ii) Interface → how a layer ~~come~~ communicates with its immediate upper/lower layer

(iii) Protocol → how a layer communicates with similar layer of differenet host

Reference Models:

(1) Open System Intercommunication (OSI):

Medium give/take data (Physical)
Immediate পরের জন receive করবে (Datalink)
কয়েকজন পরের receive এর route (Network)
প্রেরক থেকে শেষ পর্যন্ত data পাঠানো (Transport).
application data পাঠাবে (Application)

} 5 টা কাজ

* Presentation and Session কম use হয়

* Bottom most layer of end to end info → Network

7 layers:

Application → Presentation → Session → Transport → Network → Data link → Physical

* Routers have only network, datalink and physical layers

* Only end devices have all 7 layers.

TCP/IP Model :

- **Application**
  - → Data final source/sink
  - → TELNET, FTP, SMTP, DNS (Protocol)
- **Transport**
  - → End to end communication overall
  - → TCP, UDP (Protocol)
- **Internet**
  - → Route from source to end
  - → IP (Protocol)
- **Host to Network**
  - → Medium + Datalink
  - → ARPANET, SATNET, Packet radio, LAN (Network)

* Each model step has multiple protocols/networks in use.

* Common Steps (Central Concepts)
  - → Service
  - → Interface
  - → Protocols

A critique of OSI Model

- → Bad timing
- → " technology
- → " implementation
- → " politics

~~Critique of TCP/IP → Not a gener~~ reference model

What we use:

*We use a hybrid model → 5 base layers of OSI model

- → Application layer 5
- → Transport " 4
- → Network " 3
- → Data link " 2
- → Physical " 1

* ARPANET

a) Telephone switch architecture → If switch node fails, all fails

b) Baran's proposed distributed switching system

Critique of TCP/IP

- → common concepts not distinguished
- → Not a general model
- → Host to network is not an actual layer
- → Physical and data link layers non-existent
- → Minor protocols deeply entrenched, hard to replace

Example networks

- → The Internet
- → X.25, Frame Relay, ATM (Connection oriented)
- → Ethernet
- → wireless LAN: 802.11

Connection oriented → Fixed logical/virtual data route

" less → Not " " " " " ,

{each data packet can take any route it wishes.

Lecture 4 11-06-23

ARPANET, NSFNET,
↳ Intercontinental
↳ Interuniversity

ATM virtual circuits:

Asynchronous Transmit Mode (ATM) ATM cell
↳ No need for timing synchronization

ATM Reference Model
→ Upper layer
→ ATM Adaption Layer → CS (uniform interface)
→ SAR (segmentation and reassembly)
→ ATM layer (creating cell headers)
→ Physical Layer → TC (creating checksum and final cell)
→ PMD (Physical layer access)

Ethernet

Wireless LAN
→ Wireless networking with a base station
→ Adhoc networking

Network Standardization → Who's who?
→ Telecommunication World
→ International Standards World
→ Internet Standards World

ITU:
(International Telecommunication Union)

Main sectors
- → Radiocommunications
- → Telecommunication Standardization
- → Development

Classes of Members
- → National Governments
- → Sector ← Members
- → Associate ← "
- → Regulatory agencies (BTRC in BD)

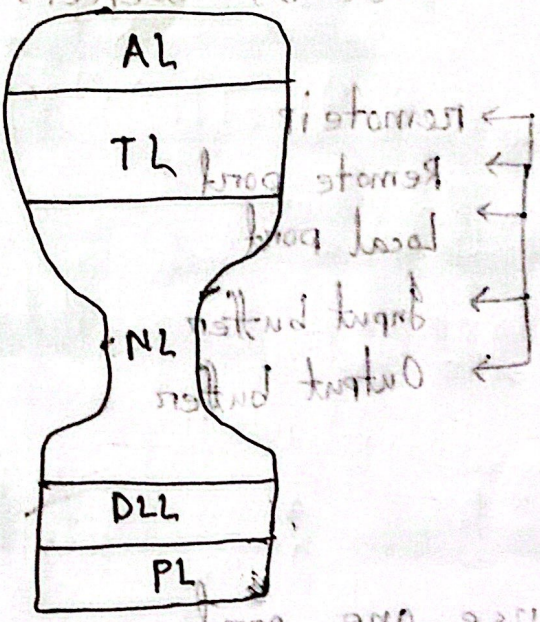
IEEE 802 standard table

Metric Unit table

Lecture-4 (contd.) 11-06-23

Chapter-5

The Network Layer



Network Layer implementation must be very simple.

Design Issues
- → Store and forward packet switch
- → Services provided to transport layer
- → Implementation of connationless design
- → " " connection-oriented "
- → Comparison of virtual-circuit and datagram subnets.

Connectionless:

Routers only know elements ~~wh~~ within its network.

Not the final destination if it needs multiple router subnet. ~~And the~~ Source → Router → Router ..... → Dest

When routers overload → The routings tables may change ~~or~~ in real time

Connection Oriented:

* id based

* each packet is tagged when sent from source. — This tag is distiguished and acknowledged in selecting route in each router.

* Comparison between virtual circuit (connection oriented) and datagram (connectionless) in slide

Lecture-5 12-06-23

Desired Properties of a routing Algorithm:

- Correctness
- Simplicity
- Stability (i) should converge quickly
  ii) should work over a long period of time)
- Robustness (should be able to adapt to all types of topology changes)
- Fairness
- Optimality

* Conflicting metrics : Fairness vs Optimality (slide)
* Even in case of optimality, metrics can be conflicting in some cases. Example: Delay vs Throughput

Throughput maximization through full network utilization
↓
Network queues are close to their capacity
↓
Switching delay increased
↓
Time delay increased

Conflicting!

* However in some cases, throughput and delay might not be conflicting.

Ex: # of hops ↓ (delay ↓ BW consumption ↓) Throughput ↑

Not conflicting

Routing Algos:

- Adaptive : Dynamic decision based on current traffic and topology
- Static/ Non-adaptive : Do not use estimates/measurements of current traffic/ topology
  - → Shortest Path Routing (Ex: Dijkastra)
  - → Flooding

Flooding: → Every incoming packet is sent out on every outgoing line except the one it arrived on.

→ Disadv.: Generates vast # of (∞, if there is no measure) duplicate packets.

⇒ Avoidance :

- Hop count (decremented at each hop, ~~discarded at~~ discard the packet for zero hop count)
- Seq # (list of seq# for each src is stored for discarding duplicates)
  use packet identifier (sourceid + seq number)
- limit growing list : Augmenting by a counter k. summarizing up to k seq# are seen.
  if get 1 2 3 7 8 packets sequence → store 3 7 8 → reduces seq # storage

⇒ Selective Flooding :

Sent to only those lines that are approximately in the right direction

Application : Military, Distributed DB, wireless Network, Benchmarking . . .

Routing Algorithms:

(I) Optimality Principle

(II) Shortest Path Routing

(III) Flooding

(IV)

Optimality Principle:

Fix sink and then find shortest paths to the sink from other nodes.

Distance Vector Routing:

Lecture-6 13/06/23

Distance Vector Routing (DVR):

- Each vector contains a routing table
- Each routing table contains one entry for each router in the subnet
- Each entry contains two info:
  (i) Preferred outgoing line/ next-hop node for corresponding destination router
  (ii) Estimate of the distance to that destination (the best known distance)
- Different metrics used for estimating distance
  - # of hops (1 for neighbour)
  - Time delay (by sending echo packets (special purpose packets), which initiate timestamping and sending back from neighbours)
  - Queue length (by simply examining the queues of other nodes)

Algo:

① Each router exchanges own routing table info with its neighbors.

↳ only distance part (excluding the preferred next hop) suffices

② After receiving an info from a neighbor, a router updates its own routing table accordingly.

A — B — C — D — E

A

Dest.	Dist.	Nexthop
A	0	A
B	1	B
C	∞	—
D	∞	—
E	∞	—

B

A	1
B	0
C	1
D	∞
E	∞

C

A	∞
B	1
C	0
D	1
E	∞

D

A	∞
B	∞
C	1
D	0
E	1

E

A	∞
B	∞
C	∞
D	1
E	0

Full routing tables
Initially (only by echo exchange)

Now,

Update in C after getting routing table of B

C

A	2	B
B	1	B
C	0	C
D	1	D
E	∞	—

Update in C after getting routing table ~~from~~ of D.

C

A	2	B
B	1	B
C	0	C
D	1	D
E	2	D

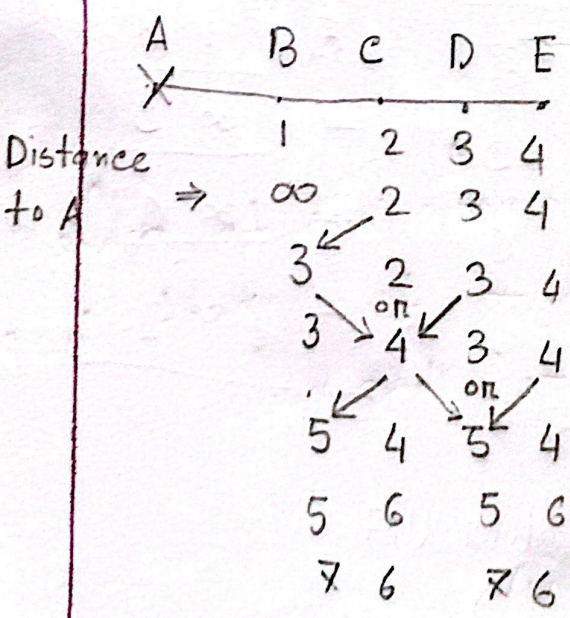
* Good news is spreading at the rate of one hop per exchange

- N-hop length subnet → Everyone will know good news in N exchanges
(presence of a router in the subnet)

Longest path in N-hop subnet
Diametre of the graph

Ques: what about bad news?

Let A fails in the linear topology (A mayb down or A-B link broken)



Initially after B detects it has no direct connection to A anymore, B thinks it can go to A through C. C has to update its distance to A as both of its neighbors have 3 hop paths to A

⇒ Bad news travels slowly
Count to infinity problem (longest path+1)

Reason: No idea about whether I am the next-hop node in neighbor path (only know its own next hop)

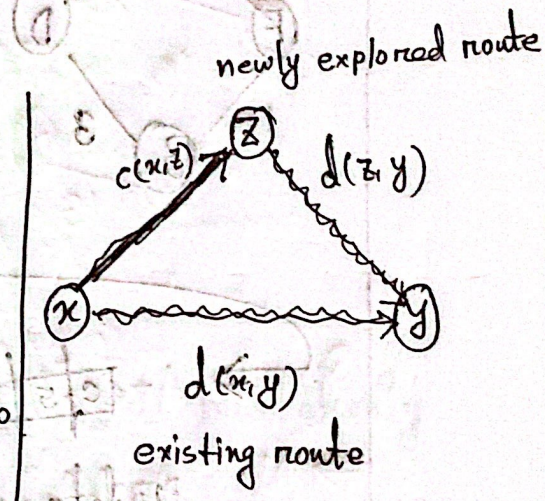
⇒ Elaboration and solution from algorithmic perspective (book from Petterson)

Lecture-7 18/06/23

Algorithmic Perspective of DVR (from Patterson)

Initially routing table of each node will have finite distance for only directly connected nodes.

Algorithm for update:



Update (x, y, z)

$d \leftarrow c(x,z) + d(z,y)$ # cost of x to y with first hop z

if $d < d(x,y)$

found better route

return d, z # update cost, next hop

else return $d(x,y)$ # existing cost next hop

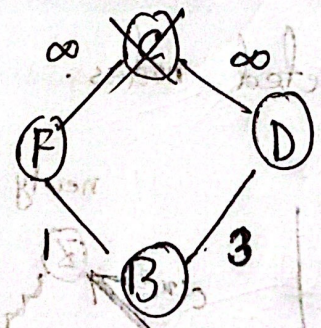
next hop y

⇒ Only direct neighbours are explored here

⇒ Repeat the update until convergence (Bellman-Ford Algo.)

Time of convergence: Diameter of the graph

⇒ Impact of node failure :



⇒

C	∞	-

⇒

C	2	F

B's table arrive at D and F

→

C	5	B

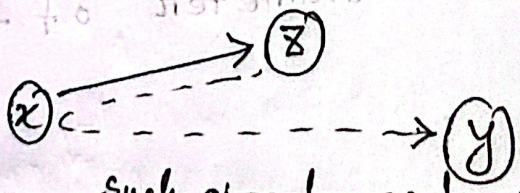
Update entry at D's table

C	3	B

Updated entry in F's table

(WRONG!)

Solution : while updating a node has to make sure that, it is NOT the next hop on the path from its first hop node.



Such circular route shall not be explored

Update (x, y, z)

$d \leftarrow c(x,z) + d(z,y)$

if $d < d(x,y)$ and $x \neq nexthop(z,y)$ } "Split horizon rule"

found better route

return d, z

else

return $d(x,y)$, $nexthop(x,y)$

⇒ Now fully ok?

Ans:- No, as the entry in D's table still mistakenly gets updated. Reason: B's state table entry to C updates table in D only once, as the algo only decreases the cost/distance, however does not increases it.

Solution: Always accomodate the update from next hop node.

Update (x, y, z)

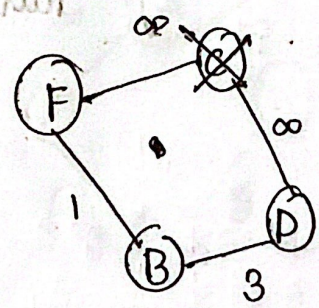
$d \leftarrow c(x,z) + d(z,y)$

if $\underbrace{nexthop(x,y) = z}_{\text{Forced update}}$ or $(d < d(x,y)$ and $x \neq nexthop(z,y))$

return d, z

else

return d(x, y), nexthop (x, y)



⇒ | C | 5 | B | ⇒ | C | ∞ | - | ⇒ | C | ∞ | - |

in D in B in D

Here we will only consider update from the nexthop neighbour which will first set wrong values but then fix after one more iteration

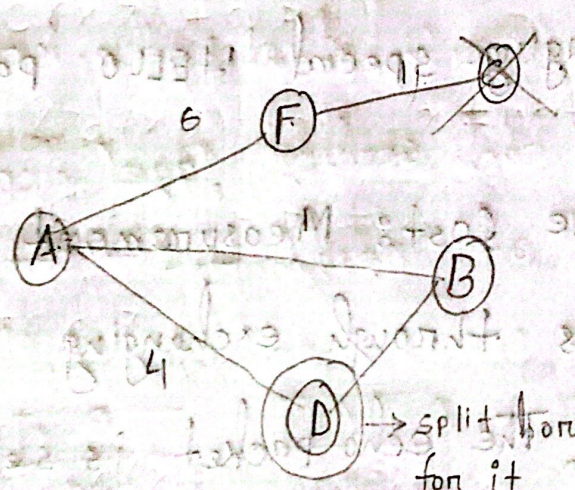
Ques: Are all problems solved?

⇒ No problem solved in next lecture

*Check Shattik's solve in Farhan's dm

Lecture - 8

Problem:



(Details in handout)

→ split horizon rule not applicable for it

~~Link Routes~~

Link State Routing (LSR):

Basic difference with DVR

DVR – sends info of all nodes to neighbors

LSR – " " " neighbors to all nodes

Five operational steps in LSR:

(i) Discover neighbors → (ii) Measure cost to each neighbor

↓

(iii) Construct a packet incorporating cost to neighbors

← (iv) Send the info to all othe routers

↓

(v) Compute shortest path to every other router

(i) Discover neighbors: Identification of neighbors through exchanging special HELLO packets.

(ii) Measuring Line Cost: Measurement of delay/cost to neighbors through exchanging special echo packets. The echo packet is sent back immediately. (delay = $\frac{RTT}{2}$) *RTT = Round trip time

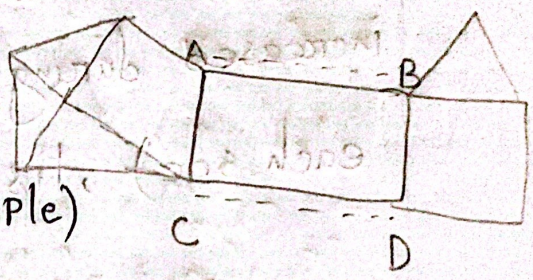
⇒ Two different techniques to start the RTT timer:

a) when the echo packet is queued: considers load balancing

b) when the echo packet arrives infront of Q: Does not consider load balancing

Load Balancing:

A-B and C-D are of same BW, however experiences different traffic loads. (A-B > C-D for example)



Then the route with less traffic (C-D) will be considered for balancing traffic load.

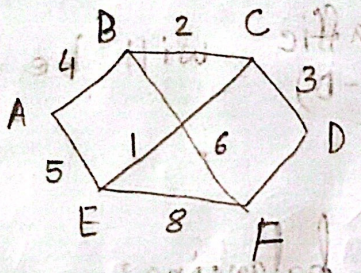
⇒ Problem of considering load balancing:

(A-B > C-D) ⇒ Choose C-D in new update ⇒ (C-D > A-B) ⇒ Choose A-B in new update ⇒ :-
traffic

⇒ Solⁿ: Traffic is splitted over multiple lines in fractions. Router choosing only a single line (such split inhibits ~~direct~~ drastic change in traffic load over a link just after one update)

(III) Constructing Link State Packets:

An LSP contains: own identity, a seq # (number increases during each sending), age (decreases in each sec), list of neighbors. ↳ (storage time)



A	
Seq	
Age	
B	4
E	5

LSP of A

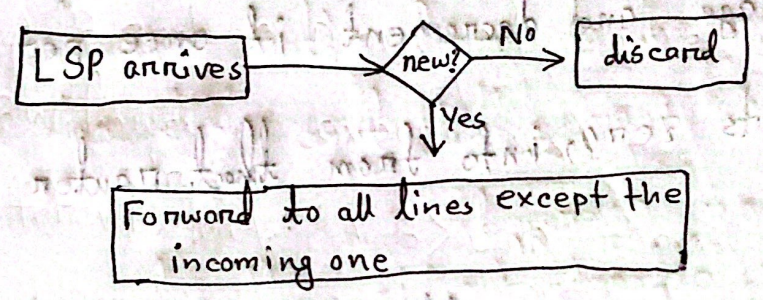
B	
Seq	
Age	
A	4
C	2
F	6

LSP of B

(IV) Distributing LSP's (the trickiest part of the algo):

Lecture-9

Distributing LSPs:



→ similar to flooding
→ Extent is kept checked using seq#

⇒ Problems with the algo (mainly with seq#)

① Wrapping up of the seq#: (0, 1, ... max#, 0, 1, ...)
↑ Seems to be obsolete

→ 32 bit seq#; 1 LSP per sec ⇒ 137 years to wrap around

② Router crash: (0, 1, ... n crash 0, 1, ...)
↑ Seems to be obsolete

→ seq# starts from 0 after enabling a crashed router. So, all LSPs generated from this router (up to previously sent max #) will be discarded

③ Corrupted seq#: (Huge effect if it happens in MSB side)
→ Suppose the 16th bit gets corrupted (0 to 1) and 65540 is received instead of 4. All valid LSPs with seq# 5 to 65540 will be rejected in this case.

Solution:

Include "age" and decrement it once per second. When the "age" gets zero, info from that router is discarded.

[So, if any of the previous problems would happen, we will end up with only a few wrong discarding events, (up to time age > 0)]

→ "Age" is also decremented during initial flooding process (to reduce the extent of flooding)

⇒ Increase robustness of the algo:

→ An LSP after being received, is not immediately queued for transmission. Instead, it is first put in a holding area to wait for a short while.

→ The info in holding area helps in discarding duplicate LSPs, determining where to acknowledge (each LSP is acknowledged) and where to send (excluding the incoming line). The holding area is periodically scanned in

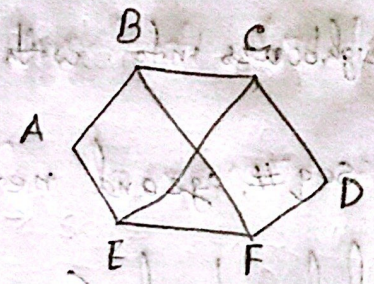
a round-robin order to select a packet or acknowledgement to send.

Src	Seq#	Age	Send flag A	Send flag C	Send flag F	ACK flag A	ACK flag C	ACK flag F
A	21	60	0	1	1	1	0	0
F	21	60	1	1	0	0	0	1
E	21	59	0	1	0	1	0	1
C	20	60	1	0	1	0	1	0
D	21	59	1	0	0	0	1	1

Holding Area of B

Arrived directly from A, so, ACK A
Send to rest
" " " F, " " F.
" " "

⇒ If a copy of C arrives through F before performing corresponding forwarding, this entry will become 1 0 0 0 1 1



⇒ If we would store only the send flag (assuming ACK flag as its compliment), then we will have trouble at the beginning (beginning of ~~each~~ each period), when al SEND and ACK flags should be 0 (not complement of each other)

⑤ Compute Shortest Paths to every other routers:

→ Construct entire subnet graph after accumulating all LSPs. Each link is represented twice, which can be used seperately or averaged.

→ Shortest Path determination in each node using Dijkastra's algo.

Example of LSR → IS-IS system, OSPF system

DVR vs LSR comparison → in handout + slide

DVR	LSR
1. Shows everyone's info with neighbours	1. Shares neighbours info with everyone
2. Routing table contains dist. and next hop	2. Contains seq #, age and next hop
3. Stores localized network, less memory	3. Stores whole network. So, more memory needed.
4. Less complex	4. Computation more complex
5. Convergence slow	5. Convergence faster
6. Time and complexity less	6. Time and complexity more
7. More distributive	7. Less distributive
8. Multiple path not possible	8. Multiple path possible.