Prelim Review

CS 4450 (Spring 2018)

Vishal Shrivastav, Cornell University
How to send data from point A to point B?
Characterizing a network link

- **Bandwidth** – Number of bits you can send per unit time
- **Latency** – Time it takes for a bit to reach the other end
  - Transmission delay: \( \frac{Packet\ size}{Link\ bandwidth} \)
  - Propagation delay: \( \frac{Link\ length}{Speed\ of\ propagation} \)
A little bit closer to reality...

- What happens if A sends faster than C could transmit?

- Queuing delay
  - Time to transmit all the packets in front in the queue

Congestion!!!
Packets get queued
In reality...

Can be really really complex!!!
Let’s break things down a bit...
Two ways to share the network...

**Circuit switching**
- Reserve the resources before sending the data

**Pros**
- Predictable performance
- Reliable delivery
- Simple forwarding

**Cons**
- Resource underutilization
- Connection set up overheads

**Packet switching**
- On-demand resource reservation
- Packets carry data + header

**Pros**
- Efficient resource utilization
- No set-up cost

**Cons**
- Unpredictable performance
- Best effort delivery
- Header overhead
- Complex forwarding
Implementing network sharing: Layering

- Divide the tasks of a network into separate modules
  - Bits on wire (Physical)
  - Deliver packets to hosts across local network (Datalink)
  - Deliver packets to host across networks (Network)
  - Deliver packets reliably, to correct process (Transport)
  - Do something with the data (Application)
How to exchange data within a local network?
Datalink Layer

• Point-to-Point: **dedicated** pairwise communication
  – E.g., long distance fiber link
  – E.g., Point-to-Point link between two routers

• Broadcast: **shared** wire or medium
  – A HUB (traditional Ethernet)
  – 802.11 wireless LAN

• Each end host has a unique “name” (MAC address)
Broadcast Medium

• A simple broadcast medium
  – Encode the name of destination in each packet
  – Send packet to every end host
  – On receiving a packet, if host name matches the packet destination then accept, else discard

• **Problem:** Collision !!
CSMA/CD

- **Carrier Sense: continuously listen to the channel**
  - If idle: start transmitting
  - If busy: wait until idle

- **Collision Detection: listen while transmitting**
  - No collision: transmission complete
  - Collision: abort transmission; send jam signal

- **Random access: exponential back off**
  - After collision, transmit after “waiting time”
  - After k collisions, choose “waiting time” from \{0, ..., 2k-1\}
Drawback of CSMA/CD

• Imposes restrictions on frame size and network length

  \[ \text{Transmission time} > 2 \times \text{propagation delay} \]

• As link bandwidth increases, transmission time decreases
• Thus propagation delay needs to be very small
• Network length can become extremely small!!
Point-to-Point Medium

- Enables concurrent communication
  - A can talk to C, while B talks to D
  - No collisions → no need for CSMA/CD
  - No constraints on link lengths and frame size
Switched Ethernet

- Each switch maintains a forwarding table

- On receiving a packet
  - Check the forwarding table for packet’s destination address
  - If entry present, forward via corresponding port
  - Else broadcast
  - Add an entry in the forwarding table (if not already present)
    - <Packet source address, Port on which packet was received>
    - E.g. <A, 3>
But there is a problem...

• **Loops in the topology !!!**

• **Build a spanning tree**
  – Contains all the vertices but no cycles
  – Links not in the spanning tree not used for forwarding packets
Spanning Tree Protocol

• Initially each switch proposes itself as the root
  • that is, switch X announces (X,0,X) to its neighbors

• Switches update their view
  • Upon receiving message (Y,d,Y) from Z, check Y’s id
  • If Y’s id < current root: set root = Y

• Switches compute their distance from the root
  • Add 1 to the shortest distance received from a neighbor

• If root or shortest distance to it changed, send neighbors updated message (Y,d+1,X)

• Each switch determines if a link is on its shortest path to the root, excludes it from the tree if not
Network Layer

How to exchange data across local networks?
Routing

- Each router maintains a routing table

<table>
<thead>
<tr>
<th>Destination address</th>
<th>Port to forward on</th>
</tr>
</thead>
</table>

- How to construct the routing table?
  - Link State Protocol
    - “Tell about your neighbours to everyone”
  - Distance Vector Protocol
    - “Tell your neighbours about everyone”
Link State Protocol

- **Every router knows its local “link state”**
  - Knows state of links to neighbors
  - Up/down, and associated cost

- **A router floods its link state to all other routers**
  - Uses a special packet — Link State Announcements (LSA)
  - Announcement is delivered to all nodes (Flooding)
  - Hence, every router learns the entire network graph

- **Runs route computation locally**
  - Computing least cost paths from them to all other nodes
  - E.g., using Dijkstra’s algorithm
Distance Vector Protocol

• Assume link between nodes x,y has cost c(x,y)

• Node u finds minimal cost paths using the following

  1. Neighbors tell me their distance to all nodes v
     • Each neighbor w gives me a “distance vector” d(w,v) for all v

  2. Node u’s cost to a given destination v is then:
     • \( d(u,v) = \text{Min}_{\text{nbrs } w} [c(u,w) + d(w,v)] \)

  3. Node u tells neighbors about d(u,v)... and process repeats
Count-to-Infinity Problem

B: 1, C: Inf
B: 1, C: 2

C’s shortest distance to A is via B
So, C should not have exchanged this information with B

A: 3, C: 1
A: 5, C: 1
A: 7, C: 1
A: 9, C: 1
A: 3, C: 1
A: 5, C: 1
A: 7, C: 1
A: 9, C: 1

B: 1, A: 2
B: 1, A: 4
B: 1, A: 6
B: 1, A: 8
B: 1, A: 10

...
IP Addressing

- 128.0.0.0/8
- 128.64.0.0/16
- 128.64.15.0/24
- 128.64.15.1
Questions?
Thank you!