Computer Communications and Networks 2017 Summary

Traiko Dinev <traiko.dinev@gmail.com>

July 1, 2019

NOTE: Some images were taken from the COMN course (Edinburgh). Everything is publicly accessible via the link at the end of the document. Credit goes to Myungjin Lee (`http://homepages.inf.ed.ac.uk/mlee23/`).
NOTE: Note this "summary" is NOT a reproduction of the course materials nor is it copied from the corresponding courses. It was entirely written and typeset from scratch.

1 Network Layers (Stack)

Different depending on who you ask. Sometimes contains *Presentation* and *Session* (think cookies) layers. See `https://en.wikipedia.org/wiki/OSI_model`

- Application layer - messages (**HTTP**, **P2P**)
- Transport layer - segments (**TCP**, **UDP**)
- Network layer - datagrams (**IP**)
- Link layer - frames (**Ethernet**)
- Physical layers - bits (Check out the Signals summary (TODO))

2 Capacity, delays

Bandwidth $\equiv$ link capacity a.k.a transmission rate (R [bits/s]).
Transmissions delay: L [bits]/R [bits/s] for an L-bit packet.

2.1 Transmission

- Store and forward: We store each packet and forward the first one in the queue
- Time-division multiplexing: Each receiver gets a slot in time. Slots rotate per a given time.
- Frequency-division multiplexing: Same as TDM, but in frequency domain (think Fourier transform)

2.2 Delays in transmission

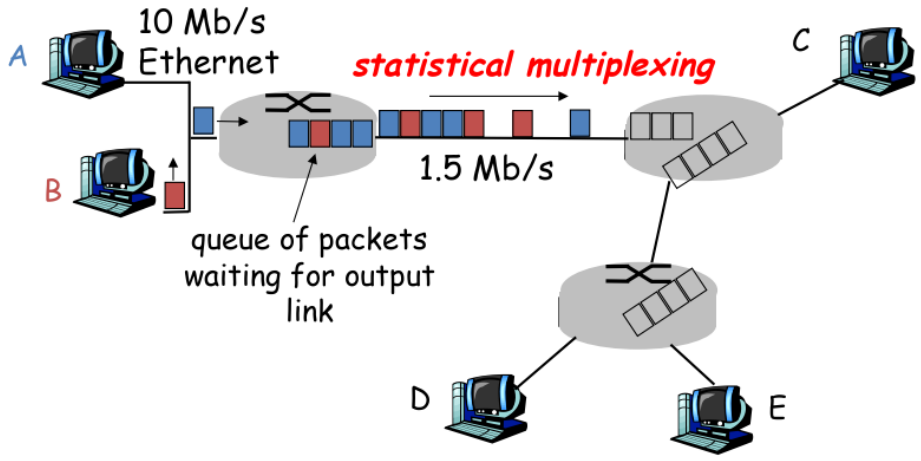


Figure 1: Time-division multiplexing (from COMN)

5 sources of delays

- Nodal processing delay
- Queuing delay (queue of packets)
- Transmission delay (above)
- Propagation delay: through the wire, dependent on the speed of light $\frac{d}{s}$, $s \approx 3x10^8 m/s$

Figure to the right, a is the average arrival of packets, L and R are defined above.

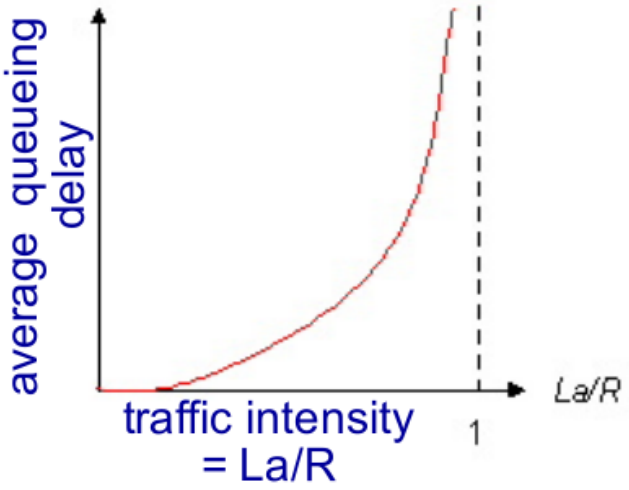


Figure 2: Time-division multiplexing (from COMN)

2.3 Throughput

Throughput is basically $\frac{\text{transferred data}}{\text{time}}$. The time is usually measured as

$$\text{time} = \text{Round-trip time} + \frac{\text{size}}{\text{bottleneck BW}} \quad (1)$$

BW-delay product: describes the "volume" of the channel (think pipe). Measured in bits/second.

3 Application layer

HTTP works over TCP (wiki's fine), sends "resource"
requests and receives the resources. Either persistent, where it can receive more than one resource (e.g. page, image, etc.) or non-persistent, where it establishes a new TCP connection for each request (see below). Persistent connections add +1 RTT per request, non-persistent +2 RTT. TCP has a 3-way handshake, hence the persistent one can ask for data on the third packet sent.
HTTP request types, but any can be used for anything:

- GET - get resource (common)
- POST - update resource (common)

```
GET /index.html HTTP/1.1
Host: www.website.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Cookie: MyName = MyValue
(blank line)
```

Figure 3: Example HTTP request

- PUT - create new resource
- DELETE - delete resource

Status codes:

- 200 - OK
- 301 - Permanently Moved
- 400 - Bad Request
- 404 - Not Found
- 505 - Version not supported
- 500 - Error

Cookies are stored in the actual request (via set-cookie:). Example request:

3.1 DNS

Domain Name Service, Website name → IP address.

- **Canonical** names are the most explicit/ longest (www.thisismywebsite.com)
- DNS servers store tuples (Name, Values, Type, TTL)

DNS Record types are:

Type	Name	Value
A	hostname	IP
NS	domain	authoritative DNS server
CNAME	domain	domain
MX	domain	mail server

Table 1: DNS record types

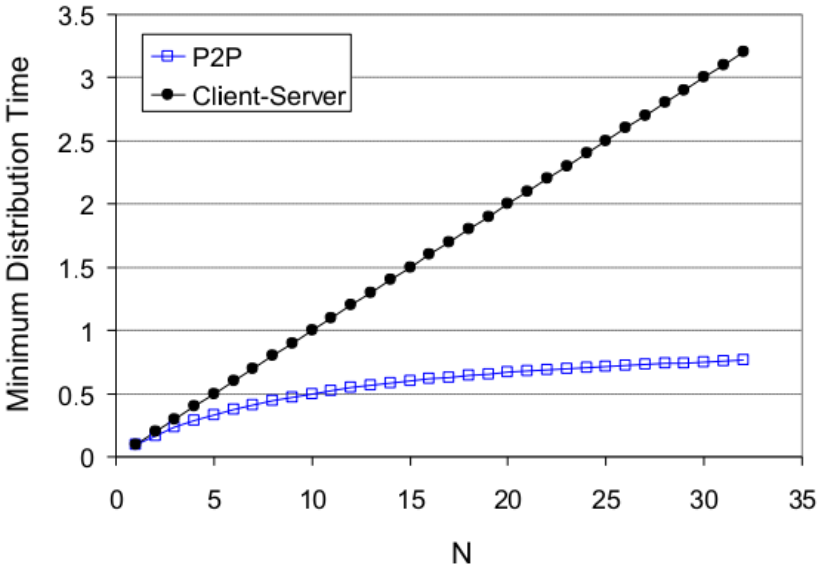


Figure 4: P2P vs. Server download times (from COMN)

3.2 Peer-to-peer (P2P)

Definitions: F - file size. d_{min} - minimum download capacity. u_s - server upload capacity. u_i - download capacity of peer.

Client-server is one server transmitting files to many clients. P2P is when one person uploads the file (as a server) and then everybody downloads it from each other. The server and the first person is always u_s.
Client-server delay until file transmission:

$$D_{\text{client-server}} = \max\{\frac{NF}{u_s}, \frac{F}{d_{\min}}\} \tag{2}$$

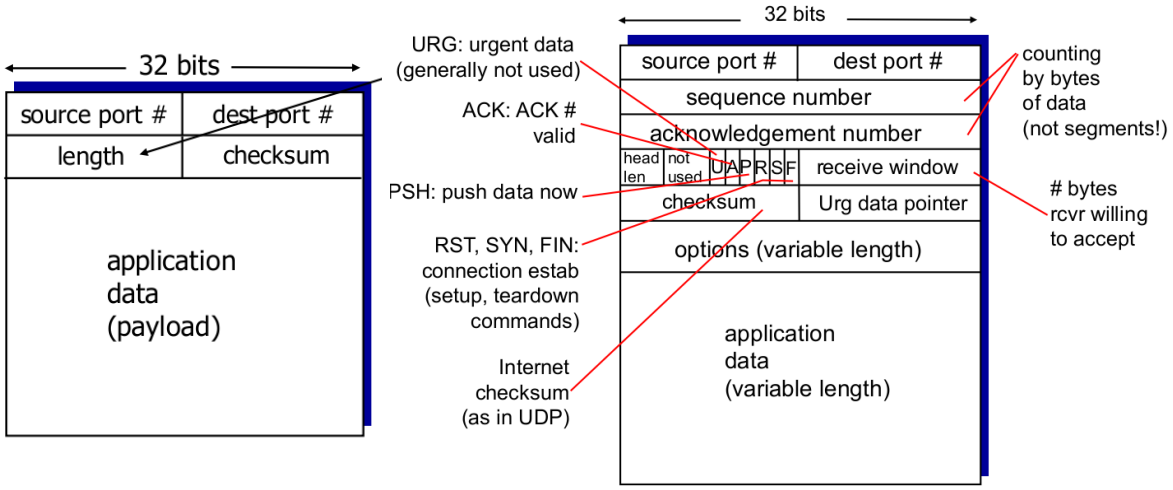
The first term is the total upload size by the server, the second is the slowest download time. P2P delay is bounded:

$$D_{\text{P2P}} \geq max\{\frac{F}{u_s}, \frac{F}{d_{\min}}, \frac{NF}{(u_s + \sum_i u_i)}\} \tag{3}$$

First term is the upload of the original file, then the max time for one peer to download it and then the sum total of the downloaded size over the sum total of the uploaded size.

3.3 Bittorrent

- One tracker has a list of torrents
- Torrent is P2P (see above)
- Peer churn - peers come and go
- Tit-for-tat - every client uploads to top 4 contributors + 1 random one with the hope that the random will become a top.
- Optimistically unchoke one peer every 10s (1 above)



(a) UDP segment (from COMN)

(b) TCP segment (from COMN)

3.4 Distributed Hash Table (DHT)

- (key, value) store in peers
- Key is converted to integer
- Put the key in the closest successor by ID
- Circular ordering makes it easier to index or we can have shortcuts to the addresses of peers

4 User Datagram Protocol (UDP)

- Connectionless, just sends packets to destination port
- Length is header + data
- Checksum is 1^s complement of sums of all 16-bit integers. So read the data as integers, sum and do 1^s complement.

5 Transmission Control Protocol (TCP)

A couple of iterations to get there. The following are principles of **Reliable Data Transfer (RDT)**

- **RDT 1.0** - perfect channel, just use UDP
- **RDT 2.0** - has bit errors, but we can do checksums to detect and send ACK (acknowledgment) and NACK messages for the client to re-send every packet.
- **RDT 2.1** - like 2.0, but ACK can also get garbled. We do sequence numbers for each packet, instead of 1/0 ACK/NACK.
- **TDT 2.2** - just has ACK, which is cumulative, so up to the last received packet
- **3.0** - 2.2 but with timeout for lost packets. Basically TCP.

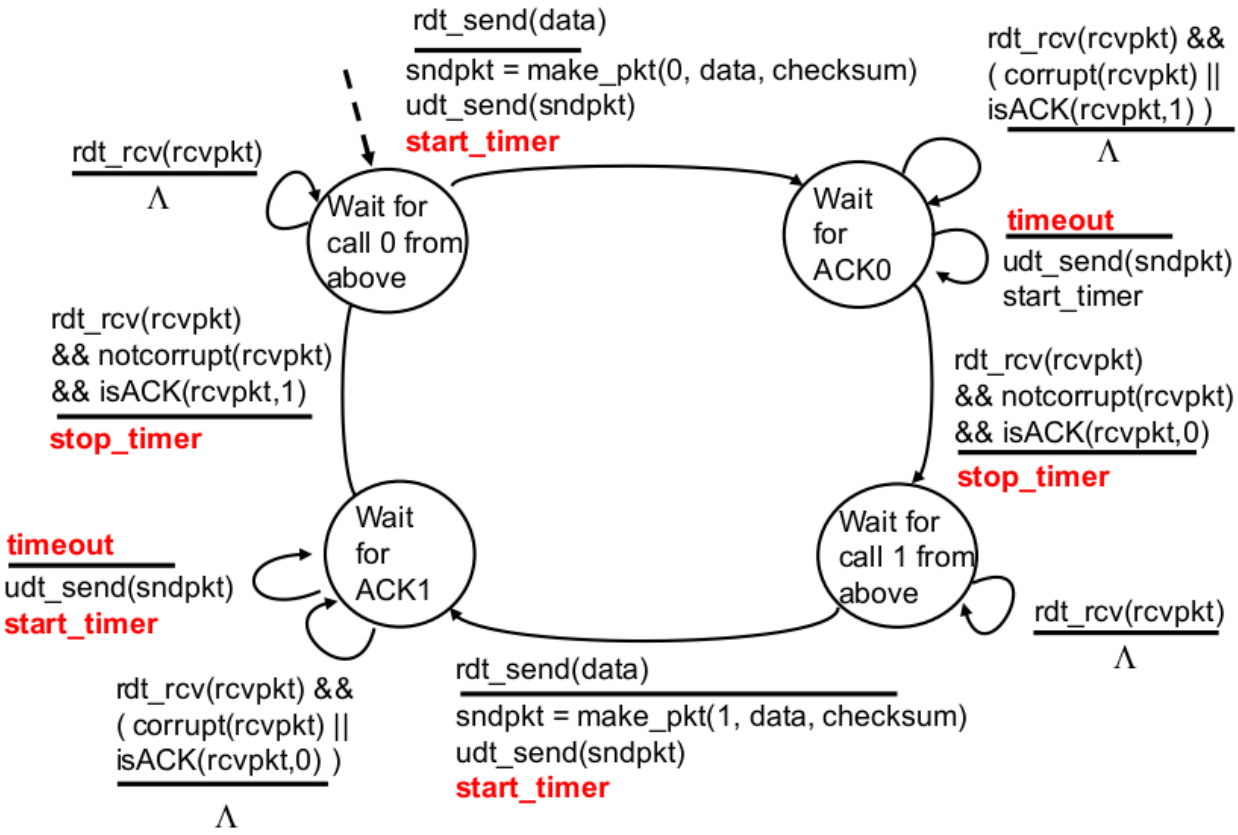
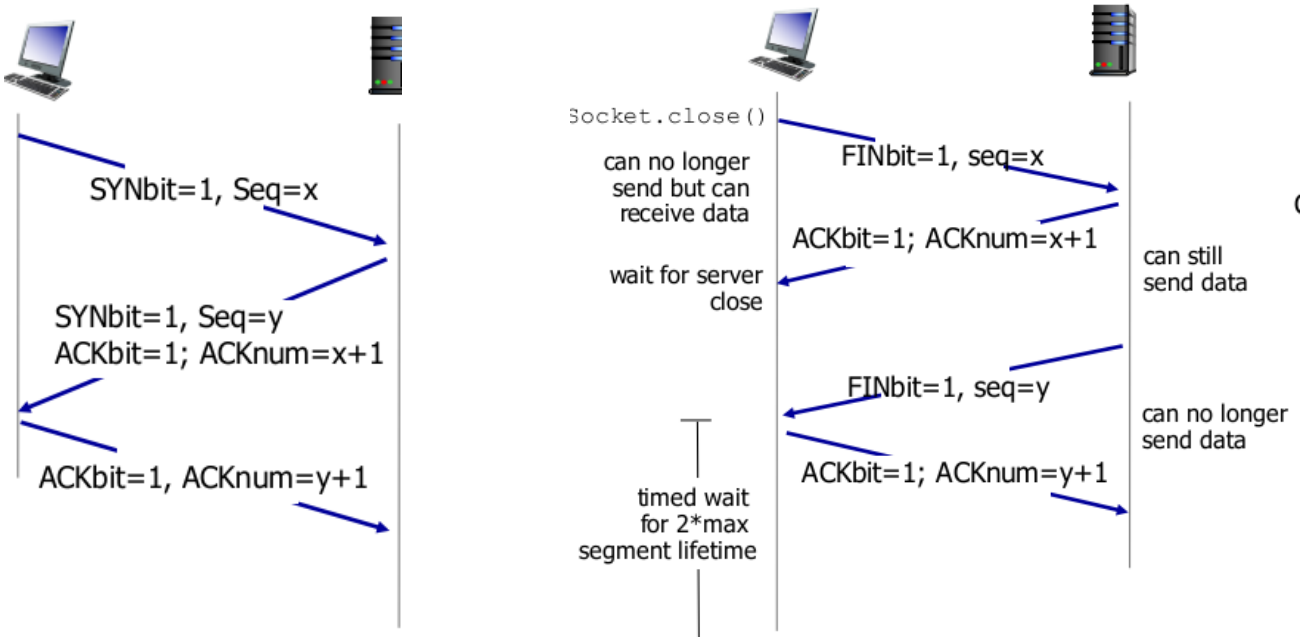


Figure 6: TCP (RDT 3.0) reliable transfer sender (kind of complicated) (from COMN)



(a) TCP connect (from COMN)

(b) TCP close (from COMN)

Figure 7: TCP connections

Stop and Wait - sends one packet and waits for response.

Go-back-N - Wait for cumulative ACKs, otherwise resend previous N packets.

Selective Repeat - Sliding window of packets, send repeat for each packet not received (timeout).
Check this for a cool visualization `http://www.ccs-labs.org/teaching/rn/animations/gbn_sr/`. TCP is Go-back-n (GBN). Speeds are:

$$D_{\text{transmission}} = \frac{L}{R} \tag{4}$$

$$U_{\text{sender}} = \frac{\frac{N\ L}{R}}{\frac{L}{R} + \text{RTT}} \tag{5}$$

5.1 Actual TCP

See figure 7 for the TCP handshake/connect and disconnect.

5.1.1 Flow Control

Do not overwhelm receiver by keeping track of their buffer (GBN):

$$\begin{aligned} \text{RTT} &= (1 - \alpha)\ \text{Estimated} + \alpha\ \text{SampleRTT} \\ \text{DevRTT} &= (1 - \beta)\ \text{DevRTT} + \beta\ |\text{SampleRTT} - \text{EstimatedRTT}| \\ \text{Timeout} &= \text{EstimatedRTT} + 4\ \text{DevRTT} \end{aligned}$$

Read Kurose and Ross about this, it's kind of complicated. Otherwise check figure 8. For the difference between Tahoe and Reno check out figure 9a.

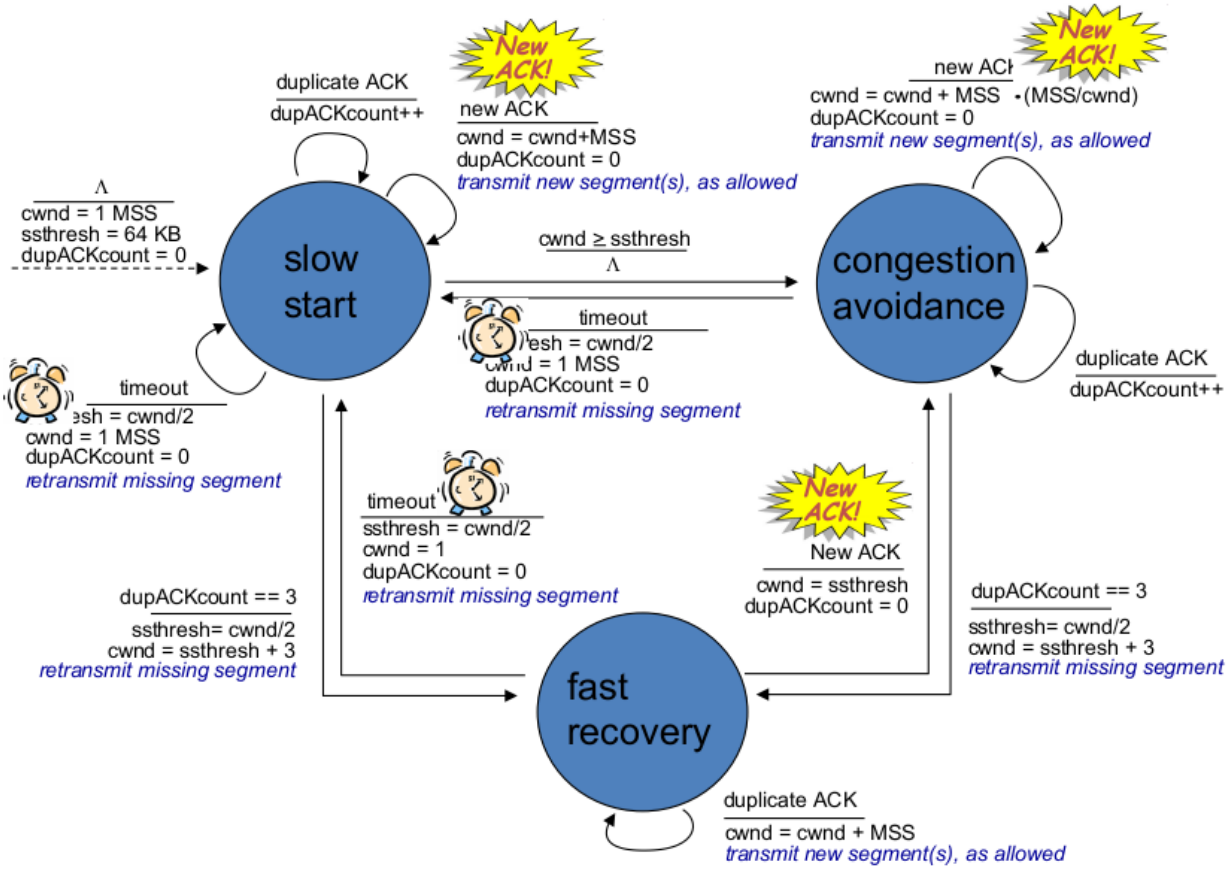
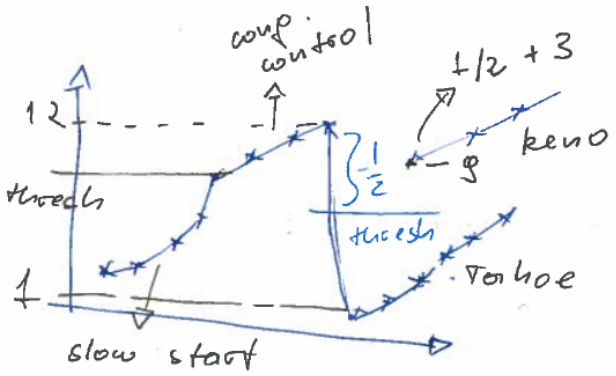
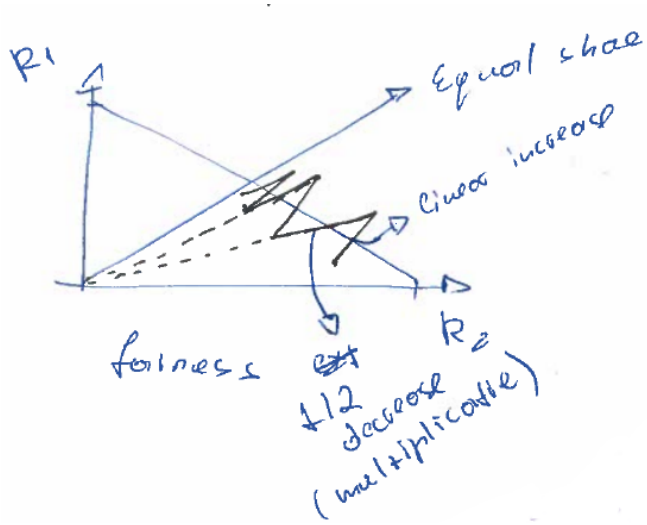


Figure 8: TCP Flow Control (kind of complicated) (from COMN)



(a) TCP Reno vs Tahoe (original work :D)



(b) TCP Fairness (original work)

Figure 9: TCP flow/ congestion control

Fairness is in figure 9b. In the beginning (slow start) CWND increases by 1 MSS (Maximum Segment Size)for both connections. This means they increase along a 45 degree line. Once the bandwidth becomes too high, they both halve it, meaning it's now halfway towards the origin. And so on.

6 Network Layer

6.1 Routers

Longest prefix matching in a table, routes to physical links connected to the router.

IP Range	Link
192.168.1.x	0
15.*.*.*	1
otherwise	2

Table 2: DNS record types

IP Mask: 192.168.0.32/24 bits means the mask is 192.168.0.x

- **CIDR** - classless interdomain routing (masks)
- **Subnet** - machines in it can communicate without routers
- **DHCP** - protocol for obtaining IP addresses
  - MSG type discover → sent to 255.255.255.255 (broadcast)
  - MSG with leased IP is sent back to 255.255.255.255 (see Ethernet below)
- **ICANN** - Internet Corporation for Assigned Names and Numbers (for DNS)
- **NAT** - Network Address Translation: Routers at the Network Layer bind connections to the outside to $\text{IP}_{\text{out}}/\text{PORT}_{\text{out}}$ to a local port **on the router**. Connections are then routed to that computer.
  - **DMZ** (Demilitarized Zone) - when one computer is exposed to the outside behind the router
- **ICMP** - Internet Control Message Protocol
  - Used to send signals via IP (i.e. without TCP or UDP on top)
  - Type 0, Code 0 → echo, ping
  - Type 11, Code 0 → TTL expired, used for *traceroute*
- **IPv6**
  - 40 byte header, 128 bit IP
  - no checksums
  - **ICMPv6** has more control messages
  - **IPv4** datagram can contain **IPv6** messages, meaning we can use v6 with v4 routers, so long as the **IPv6** routers are aware of the v4 protocol

The following are algorithms to route between routers.

6.2 Dijkstra

- $c(x, y) \rightarrow$ cost from x to y
- $D(v) \rightarrow$ cost of path to v
- $P(v) \rightarrow$ predecessor node
- $N \rightarrow$ visited node

Then just google Dijkstra..

6.3 Bellman-Ford

Uses this equation:

$$d_x(y) \ = \ \min_v\{c(x, v) \ + \ d_v(y)\} \tag{6}$$

Here $d_x(y) \rightarrow$ distance from x to y. We need to keep track of all the neighbors' distance vectors ($D_v \ = \ [D_v(y) : y \ \in \ N]$). They get synchronized every so often. The vectors are basically tables, computed with the equation above. If we iterate, then it eventually computes the correct values.

We have AS (autonomous systems). Think a collection of routers, usually in the same subnet. Some of the protocols are Inter-AS (between ASes, ha-ha..), some Intra-AS (inside Ases, um..).

6.4 Routing Information Protocol (RIP)

IntraAS

- Uses Bellman Ford (a.k.a Distance Vector)
- 30sec. advertisement from every router
- if something is 16 hops away, then it's ∞
- 180sec. = dead link
- uses UDP

6.5 OSPF - Open Shortest Path First

IntraAS

- Open source, uses Dijkstra
- Advertisement flooded to entire AS (autonomous system)
- Customizable security

6.6 BGP - Border Gateway Protocol

InterAS

- over semi-permanent TCP connections
- see illustration (figure 10)
  - Once a subnet (the outside world) becomes reachable

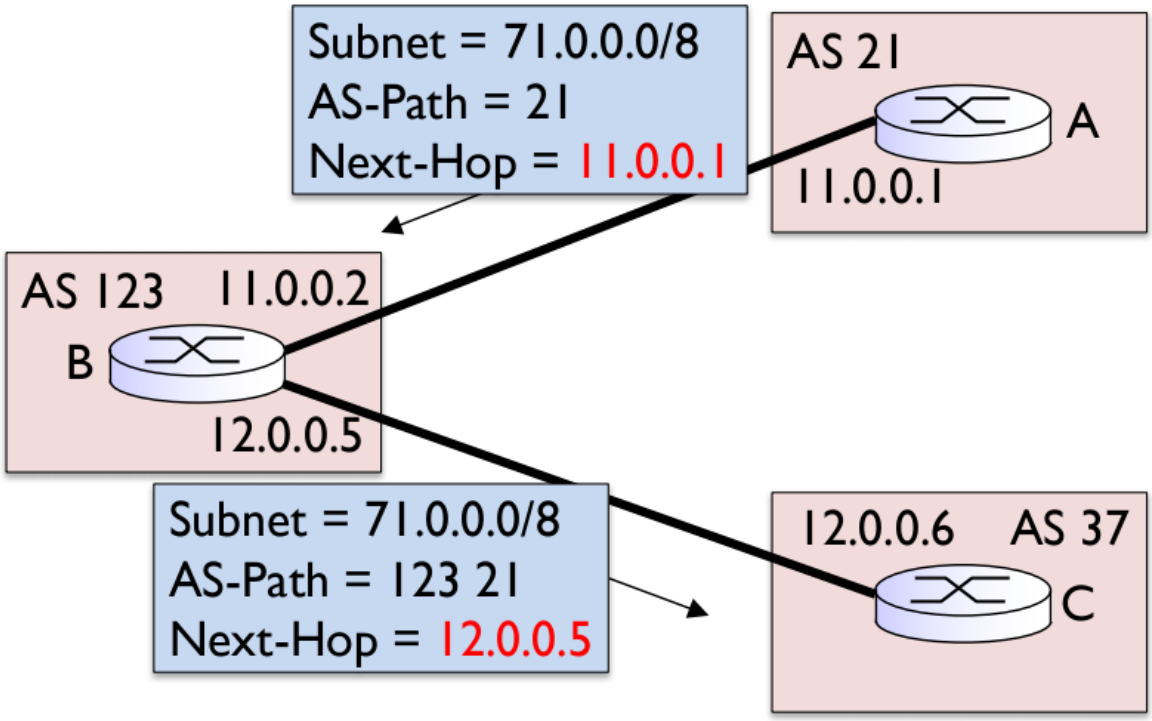


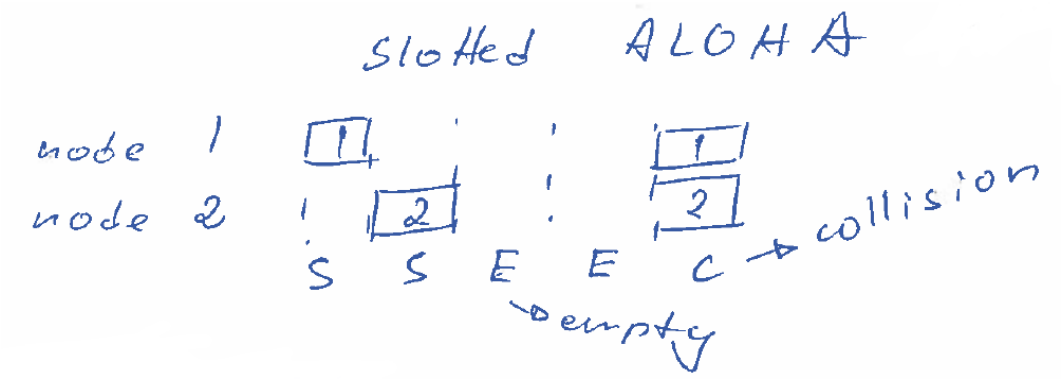
Figure 10: BGP operation (from COMN)

- The router advertises it and puts itself as the next hop
- AS-Path is the path taken to reach the target subnet
- And so routers know where to send packets to reach the subnet (71.0.0.0/8 in the diagram)

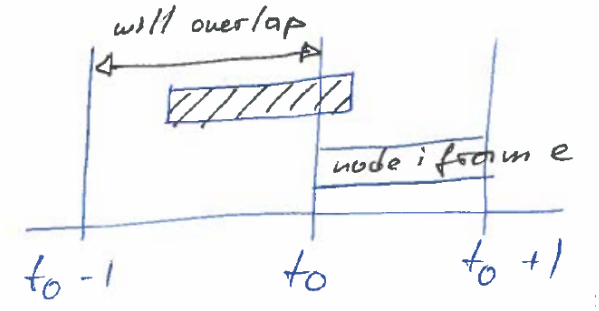
7 Link Layer

This is mostly low-level communication without knowledge about IPs. It's sending packets over physical Wires. See the scanned figures 11 for the details.

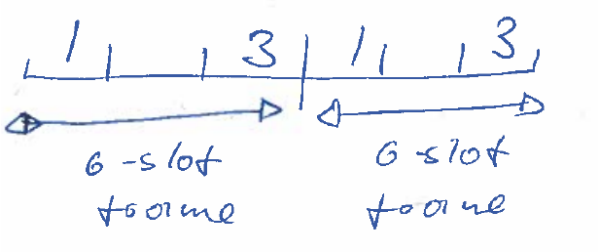
- **Time division multiple access**: each machine gets a time slot
- **FDMA**: Same with frequencies, uses Fourier transform to accomplish it
- **Slotted Aloha** - Each node transmits in slots. If there is a collision, then the node retransmits the frame in each subsequent slot with prob p until it succeeds.
  - The efficiency is $Np(1-p)^{N-1}$, i.e. each node must be the only one transmitting, where p is the probability of a node sending a frame and N is the number of nodes.
- **Unslotted Aloha** - Each node just transmits when it needs to. Need to consider the time interval $[t_0-1, t_0]$, $[t_0, t_0+1]$ for analysis.
  - The efficiency is $P(\text{success}) = P(\text{node transmits})\ P(\text{no other in}[t_0-1, t_0])P(\text{no other in}[t_0, t_0+1])$, which is $p(1-p)^{N-1}(1-p)^{N-1}$, which is even worse than the slotted version.



(a) Slotted Aloha



(b) Unslotted Aloha



(c) TDMA

Figure 11: Transmission protocols

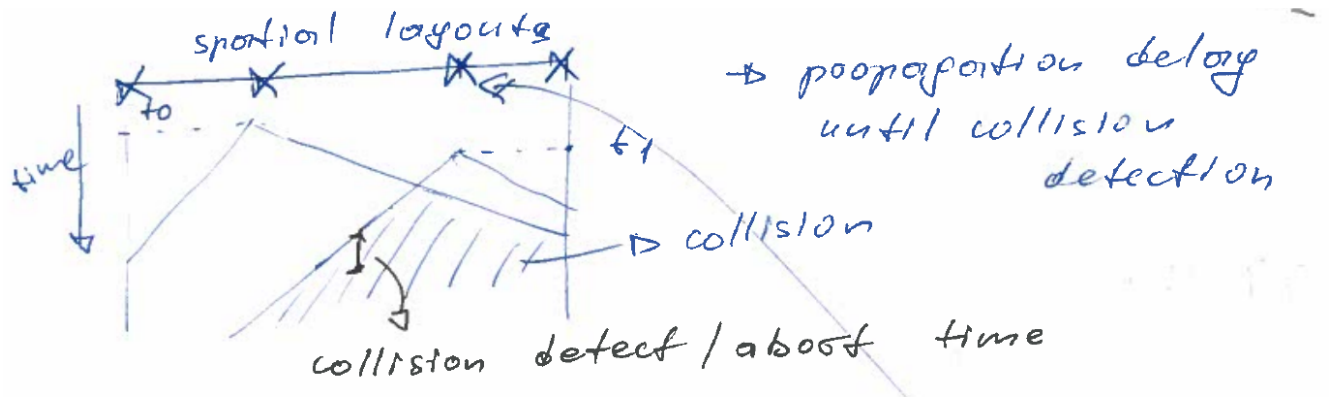


Figure 12: CSMA. The crosses are routers (say, Wi-Fi) in space. The signal propagates through time and collisions are detected by the router with some delay.

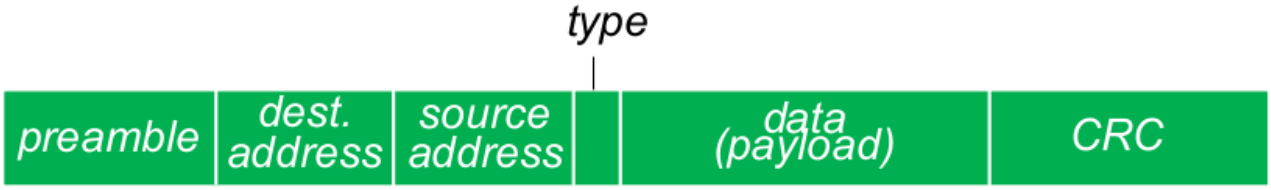


Figure 13: Ethernet Frame.

7.1 Carrier Sense Multiple Access (CSMA)

Used by Wi-Fi mostly, see figure 12. Once a node detects a collision, it does Binary exponential back-off. This means each node chooses a delay at random from the set $\{0, 1, ..., 2^{m-1}\}$ to wait. Efficiency is:

$$\frac{1}{1 + \frac{5\ t_{\text{prop}}}{t_{\text{trans}}}} \tag{7}$$

7.2 Address Resolution Protocol (ARP)

This translates IP addresses to MAC (Media Access Control) addresses, which Ethernet (below) uses for communication. Has a table of IP to MAC addresses.

IP	Mac
192.168.123.1	11:22:33:44:55:AA

Table 3: ARP Records

7.3 Ethernet

See figure 13. Low level communication based on MAC addresses. Uses Cyclic Redundancy Codes for error correction.

8 Overview

Here's an overview of the entire procedure of opening a webpage. Inspired by Kurose and Ross' Retrospective: A Day in the Life of a Web Page. I **highly recommend** you read that as well, since it's much more detailed and puts everything together nicely.

- *Scenario*: We want to open Google
- First, we need an IP. We create a **DHCP** request, put it in a **UDP** segment that goes into an **IP** datagram which goes into a **Ethernet** frame. The destination IP address is broadcast (255.255.255.255) and so is the destination MAC address (FF:FF:FF:FF:FF:FF).
- The router gets the frame, demultiplexes it back up to **DHCP** and allocates some IP with a lease. Then that is sent back via a **DHCP** ACK message back to broadcast.
- The router also put its address (192.168.0.1) as a default gateway, so that requests first go there.
- **DNS**: We now need google's IP. We create a DNS query via UDP that has a destination IP as obtained from the DHCP request.
- The frame needs to go to the gateway, but we do not know its MAC address. We create an **ARP** query with the target IP of the gateway (192.168.0.1). The Ethernet frame this goes in still has a broadcast address (FF:FF:FF:FF:FF:FF).
- The router sends back the **ARP** reply with its MAC address (BB:BB:BB:BB:BB:BB).
- We send the DNS query with the **DNS**'s server IP (8.8.8.8), but the Ethernet frame is addressed to the gateway router (BB:BB:BB:BB:BB:BB).
- **OSPF, BGP, etc.** protocols used by the routers tell them where to forward the DNS frame, which bounces around and goes back to our computer.
- **TCP** We create a TCP request to Google's IP and the rest is easy.

9 FAQ

- Why use MAC *and* IP addresses?
  - MAC addresses are hard-coded into network cards. They were the old way of communicating. IP came afterwards and groups things together (think subnets). Hence it works on top of Ethernet as described above.
- What happens to MAC addresses between router hops?
  - They get overridden. When you go through a router, it will put its interface MAC as the source MAC, since it's the only machine reachable from the outside.
- Can you describe this better?
  - Check this out

    Bit late but still here is my answer :) ... To send data you need two address, the MAC address and the IP address. Basically the sending host will ARP for a MAC address, this occurs when the local host doesn't know the MAC address of the host it has an IP address for or it will ARP for the default gateway MAC address (if it doesn't already know it) if the IP address in on a different subnet/ network. Once it

obtains a MAC address the IP packet is encapsulated in a L2 frame and sent across the media. If the IP packet is meant for a host on a different subnet/ network, it will be sent to the default gateway, this router will de-encapsulate the L2 frame (remove and discard it) check the IP address and will forward it. For the router to do this it needs a MAC address to send it over the media, It will look up the next hop in it's routing table, encapsulate the IP packet with the same source and destination IP address that was sent from the original host into a new L2 frame. This time the MAC address for the source address will be that of the forwarding interface of the router, and the receiving interface of the next hop will be the destination MAC address. This will continue from hop to hop until it reaches the final host, each time the MAC addresses will change, but the original IP address will remain the same.

Last paragraph taken from
`https://stackoverflow.com/questions/23935095/how-are-mac-addresses-used-in-routing-packets`.

10 References and Links

- `https://www.inf.ed.ac.uk/teaching/courses/comn/` - Edinburgh course on computer networks. What most of this note is based on.
- Jim Kurose, Computer Networking: A Top-Down Approach