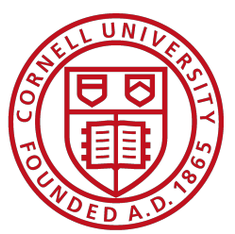
CS4410/11: Operating Systems

CPU Scheduling (Recap)

Networking

Rachit Agarwal
Anne Bracy

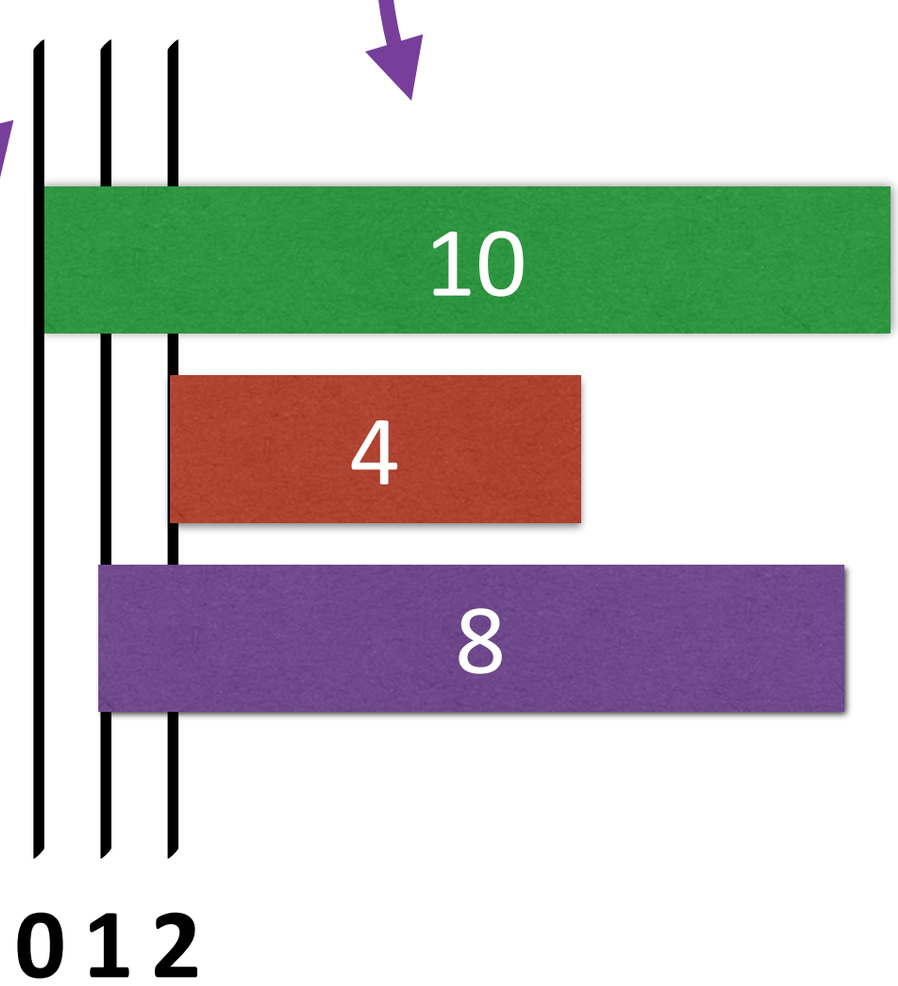
Slides based on material from Sirer, Rennesse, Rexford (Princeton)



CPU Scheduling — Example

Job Length
(e.g., #CPU cycles)

Arrival Time



- FIFO
- LIFO
- SJF
- SRTF
- RR
- Priority

Networking — What is it about?

So far: focused on what happens on a “machine”!

- **Networking**
    - How do machines communicate?
- **Lets start with a simple analogy**
    - How to move stuff from München to Ithaca?

Networking — Key Concepts

Four “concepts”!

- **Layering**
    - Abstraction is the key to manage complexity
- **Naming**
    - A name for each computer, protocol, ..
- **Protocols**
    - Computers, network devices speaking the same language
- **Resource Allocation**
    - Share resources (bandwidth, wireless spectrum, paths, …)

Networking — A Stack of Protocol Layers

Five “layers”!

- **Modularity**
    - Each layer relies on services from layer below
    - Each layer exports services to layer above
- **Interfaces**
    - Hide implementation details
    - Layers can change without disturbing other layers

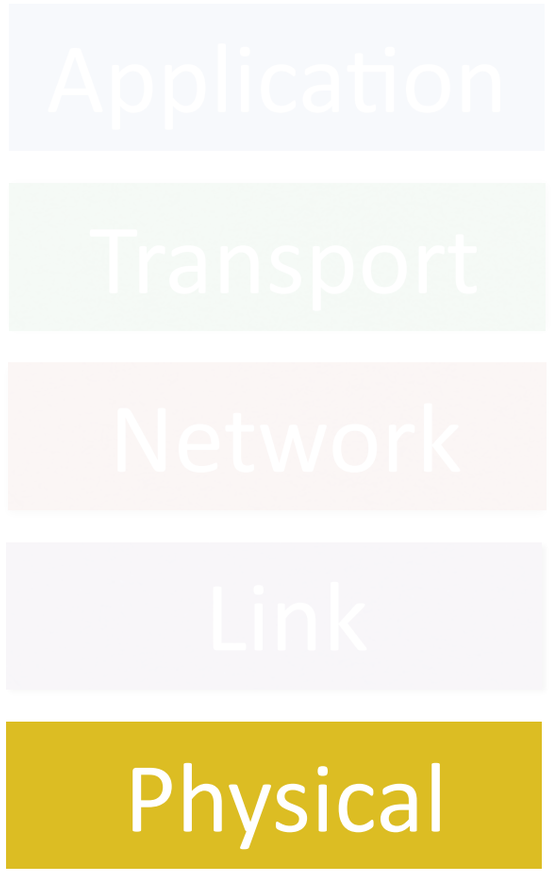
Networking — A Stack of Protocol Layers

Five “layers”!

Application	You
Transport	Post office
Network	Airplane/rail
Link layer	Postman
Physical layer	Transfer “signals”

Networking — Physical layer

- **Transfer of bits**
  - 0s and 1s
  - Not concerned with protocols



Networking — Link layer

Link = Medium + Adapters

- **Communication Medium**



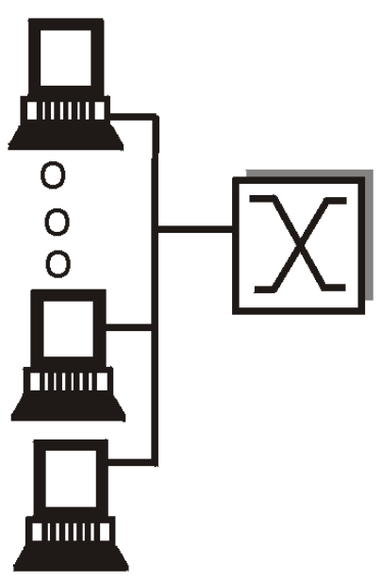

- **Network Adapters (e.g., NIC — network interface card)**



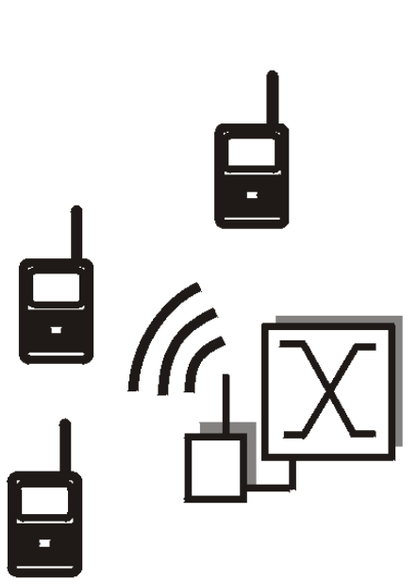
Networking — Link layer

Broadcast links = Shared Medium

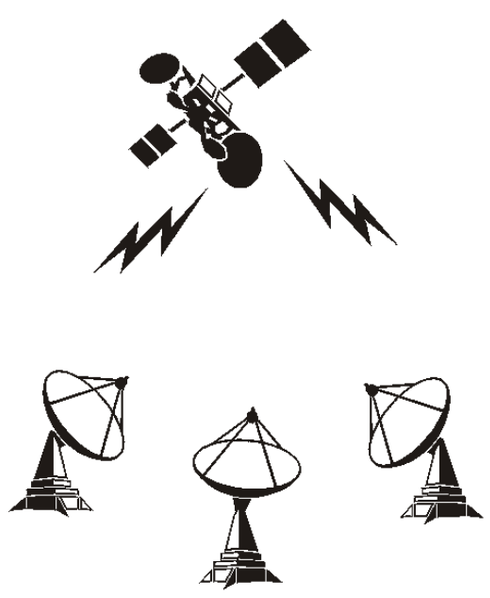
- Everyone listens to everybody



shared wire
(e.g. Ethernet)



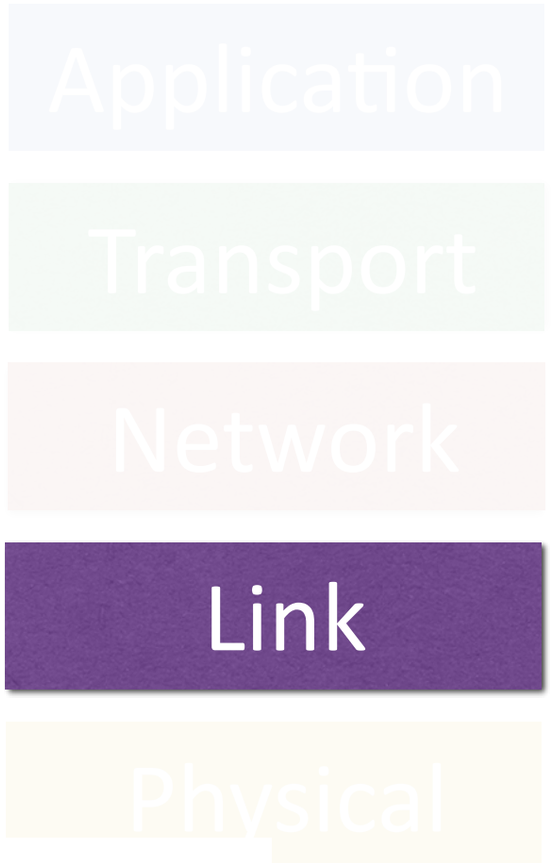
shared wireless
(e.g. Wavelan)



satellite

cocktail party

Networking — Link layer



Broadcast links = Shared Medium

- Everyone listens to everybody

source

destination

link-layer “protocol”

Networking — Link layer

Five “services”!

- **Encoding data**
    - Represented as a collection of 0s and 1s
- **Framing**
    - Put data packet into a frame; add receiver address
- **Error detection and correction**
    - Detect and (optionally) correct errors
- **Flow control**
    - When to send/receive frames
    - Depends on the protocol

Networking — Link layer

Addresses

- **Unique identifiers for sources and destinations**
  - "Hard-coded" in the adapter
  - MAC address (e.g., 00-15-C5-49-04-A9)
  - Hierarchical allocation
    - Blocks: assigned to vendors (e.g., Dell) from IEEE
    - Adapters: assigned by the vendor from its block
- **What if I want to send to everybody?**
  - Special (broadcast) address: FF-FF-FF-FF-FF-FF

Networking — Link layer

Sharing a medium

- **Ever been to a party?**
    - Tried to have an interesting discussion?
- **Collisions**

Networking — Link layer

Lets try to come up with a protocol to avoid collisions!

- **Attempt 1: Time sharing**
    - Everybody gets a turn to speak
- **Goods**
    - Never have a collision
- **Problem**
    - Wasted resources
        - During my turn, I may have nothing to speak
        - When I have something to speak, I wait for my turn

Networking — Link layer

Lets try another protocol to avoid collisions

- **Attempt 2: Frequency sharing**
    - Each person is assigned a particular frequency
    - E.g., Divide into groups; each group talks among themselves
- **Problem**
    - What if I want to talk to others?
    - E.g., one person wants to announce something …

Networking — Link layer

Attempt 3: Carrier sense, Collision detection, Random access

- **Carrier Sense**
    - Listen before speaking
    - …. and don't interrupt
- **Collision detection**
    - Detect simultaneous speaking
    - …. and shut up!
- **Random access**
    - Wait for a random period of time
    - …. before trying to talk again

Networking — Link layer

Comparing the three approaches

- **Time division**
    - No collisions
    - Wasted resources!
    - What if token is lost?
- **Frequency division**
    - Efficient and fair at high load
    - Inefficient at low load!
- **Random access**
    - Efficient at low load, inefficient at high load (collisions)

Networking — Link layer (Ethernet)

Ethernet uses 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 $\{0, \ldots, 2^k-1)$
  - (Exponentially increasing waiting times)

Networking — Link layer (Ethernet)

Interesting Properties

- **Distributed**
    - **No** Central arbitrer
    - Why is that good?
- **Inexpensive**
    - No state in the network
    - Cheap physical links

Networking — Link layer (Ethernet)

Connection-less, unreliable service

- **Connection less**
    - E.g., I am going to talk to you without getting permission first
    - Networking terminology: No “handshaking”
- **Unreliable**
    - Destination adapter does not acknowledge
        - Did you listen to what I said?
    - Adversarial behavior could bring the connections down
        - I am going to ignore the protocol
    - Untrusted data access
        - I want to listen to what others are talking