CS4450

Computer Networks:
Architecture and Protocols

Lecture 6
Data Link Layer

Spring 2018
Rachit Agarwal

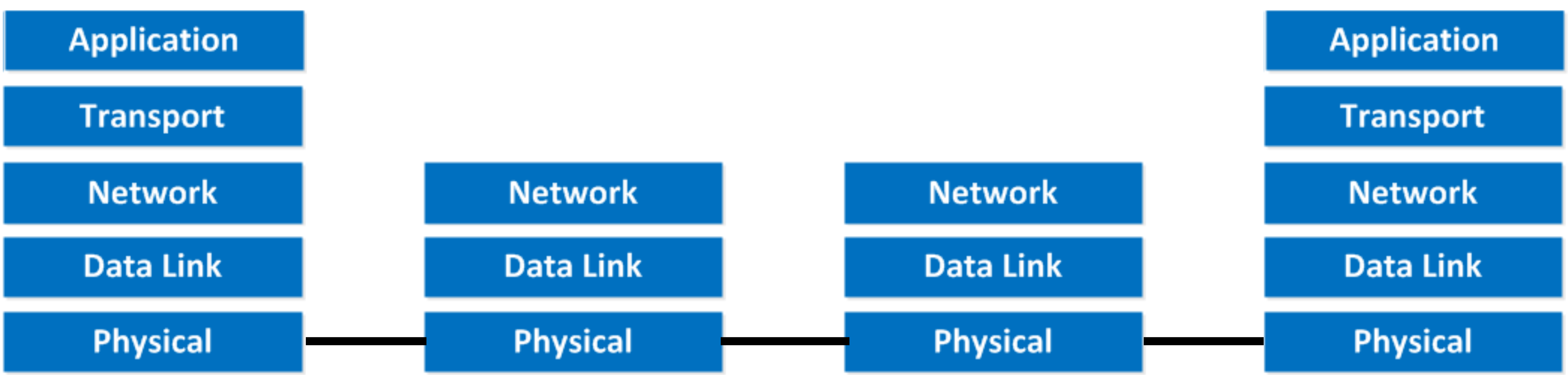
Announcements

- We will have a "live" coding class on Thursday next week (02/22)
    - **Please bring your laptops**
    - Its gonna be a lot of fun!
    - We will learn how to implement sockets, etc.
- Please read Chapter 1 of the textbook!
- And try to solve problems at the end of Chapter 1 of the textbook
    - For extra practice
    - Ask us questions on Piazza
- Remember: **in-class quizzes can happen at any time**

Quick recap from last lecture

Three design principles

- How to break system into modules
  - **Layering**
- Where are modules implemented
  - **End-to-End Principle**
- Where is state stored?
  - **Fate-Sharing**



Internet Design Goals

- **Build something that works**
- Connect existing networks
- Robust in face of failures
- Support multiple types of delivery service
- Accommodate a variety of networks
- Allow distributed management
- Easy host attachment
- Cost effective
- Allow resource accountability

Context for Today's Lecture

- You now understand
  - Network sharing (in depth)
  - Architectural principles and design goals (in depth)
  - End-to-end working of the Internet (at a high-level)

- Now its time to dive deep:
  - Link Layer (~1 week)
  - Network Layer (~4 weeks)
  - Transport Layer (~3 weeks)

- **Today: Link layer**

Goals for Today's Lecture

- **Link layer:**
  - Broadcast medium
  - Sharing broadcast medium
  - CSMA/CD

Data Link Layer

- **Communication Medium**
  - Point-to-point
    - The high-level ideas discussed so far were for point-to-point
  - Broadcast
    - Original design of Link layer protocols
    - More recent versions have moved to point-to-point
      - We will discuss why so!

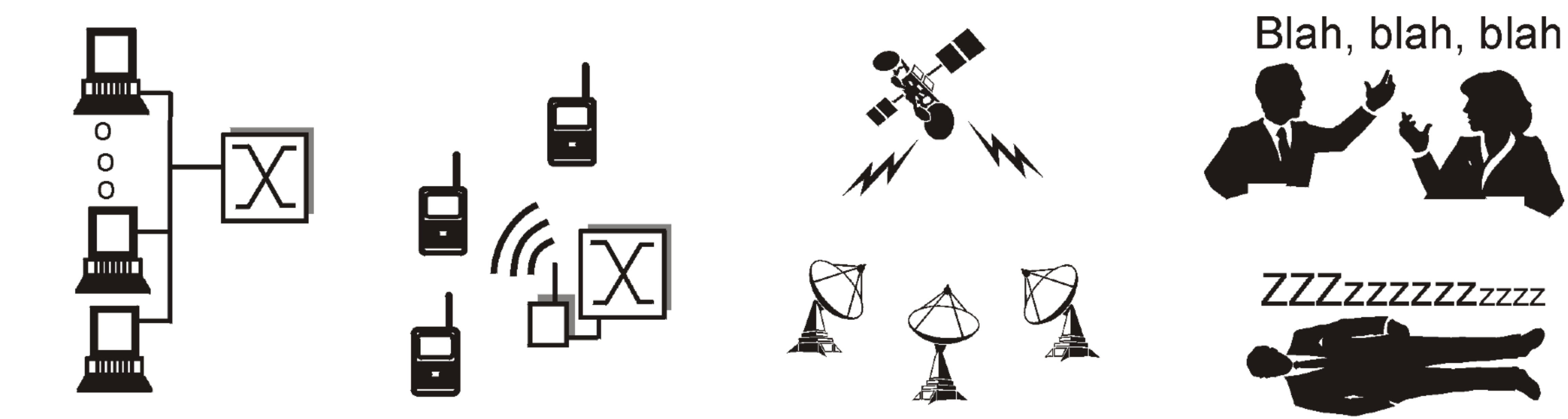
- **Network Adapters (e.g., NIC — network interface card)**
  - The hardware that “connects” a machine to the network
  - Has an “address” — MAC (Medium access control) address



Point-to-Point vs. Broadcast Medium

- 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
  - Traditional Ethernet
  - 802.11 wireless LAN



shared wire (e.g. Ethernet) | shared wireless (e.g. Wavelan) | satellite | cocktail party

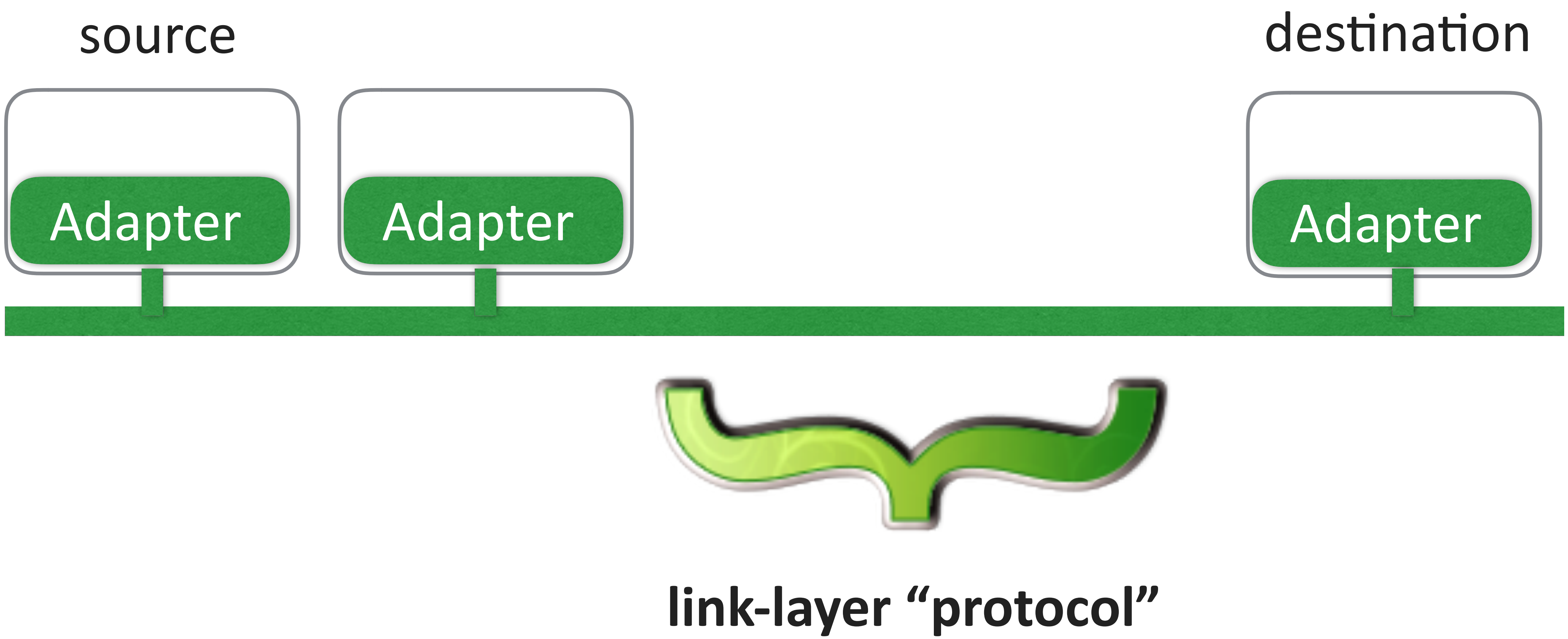
Data Link Layer: Broadcast (until ~2000s)

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

Broadcast Medium: Desirable properties

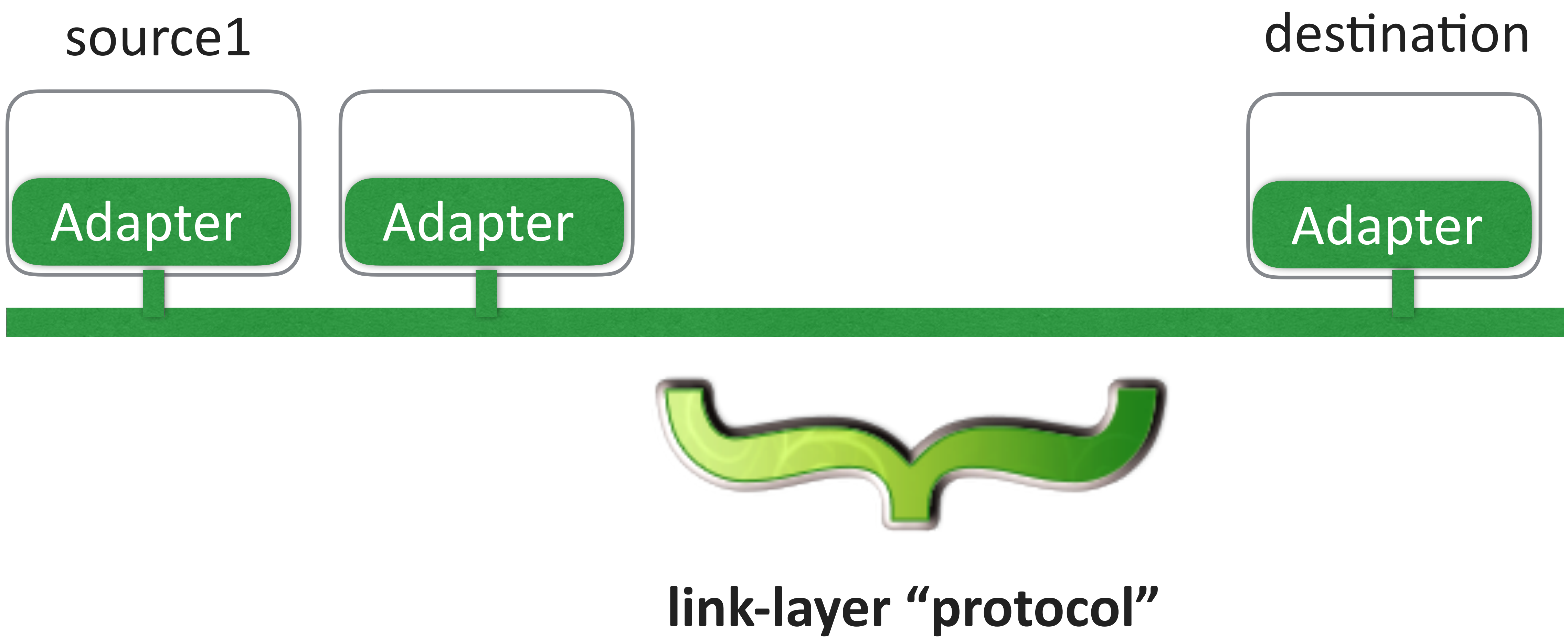
- **One and only one: data delivery**

- How do we design a link-layer protocol that ensures data delivery?



Group Exercise:

Multiple source-destination pairs
Design a protocol that allows sharing the broadcast medium

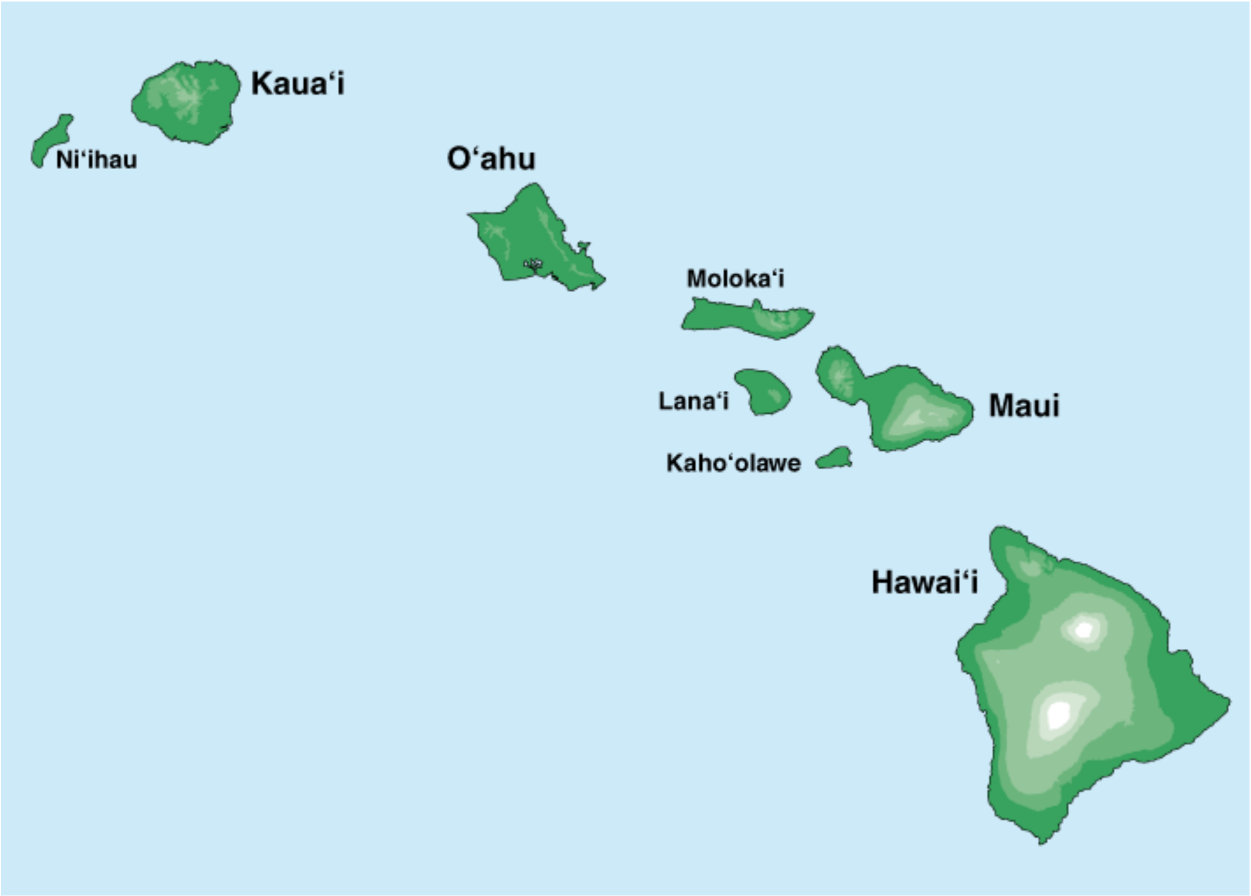


Attempt 1: Time Sharing

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

Where it all Started: AlohaNet

- **Norm Abramson:**
    - Left Stanford in 1970
        - So he could SURF
    - Set up first data communication system for Hawaiian islands
    - Central hub at University of Hawaii, Oahu



Aloha Signaling

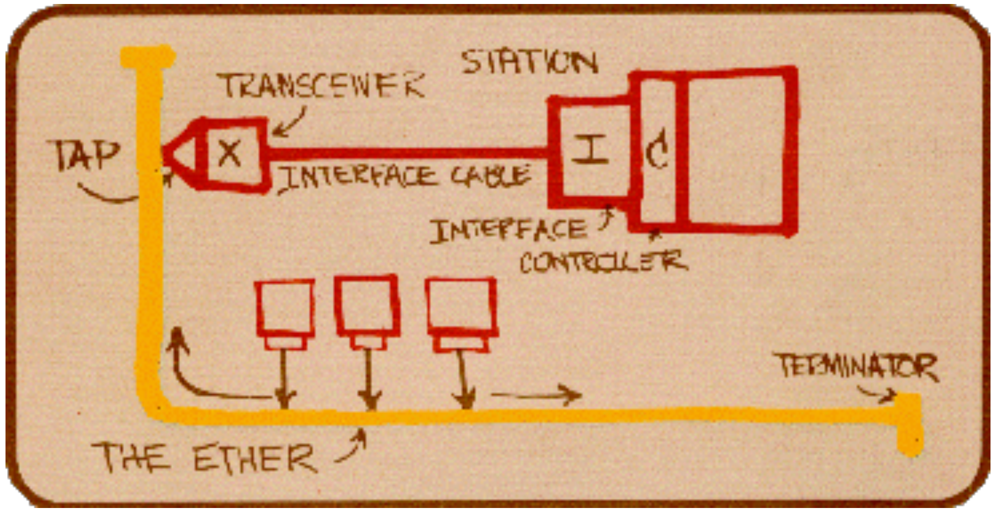
- Two channels: random access, broadcast

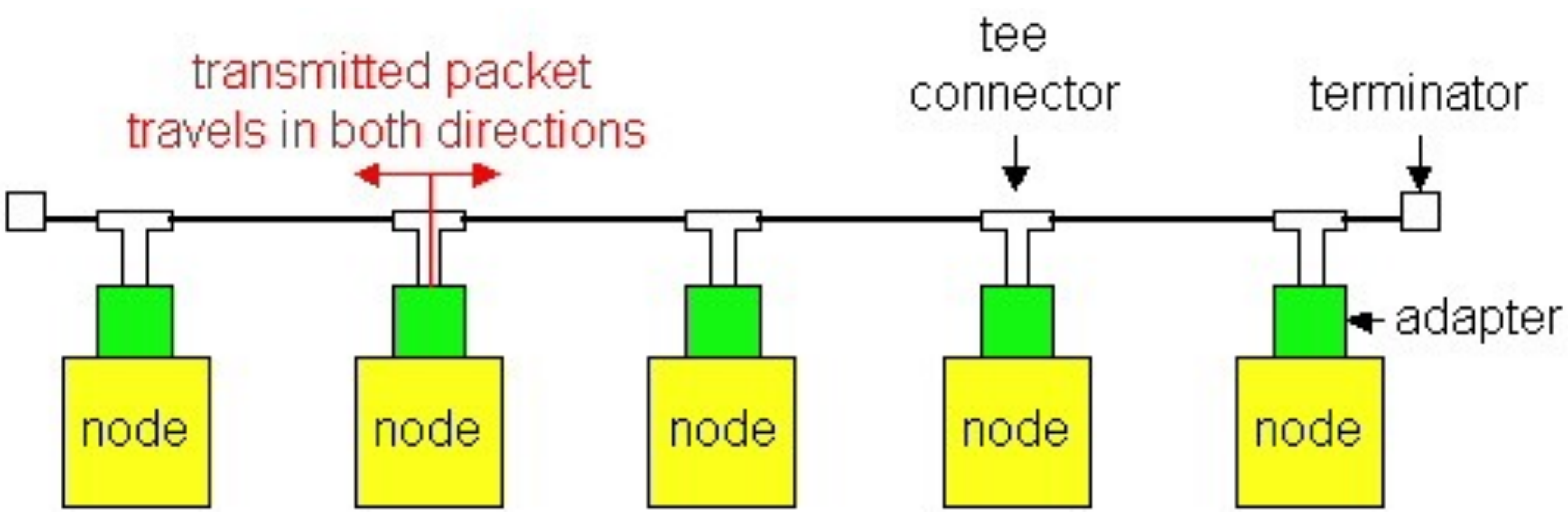
- Sites send packets to hub
    - Random access channel
    - Each site transmits packets at “random” times
    - If a packet not received (due to collision), site resends

- Hub sends packets to all sites
    - Broadcast channel
    - Sites can receive even if they are also sending

Ethernet:

- **Bob Metcalfe:**
  - Xerox PARC
  - Visits Hawaii, and gets the idea
  - Shared wired medium





Lesson:

If you want to invent great things,
go to Hawaii :-)

Sharing a broadcast channel

- **Context: a shared broadcast channel**
  - Must avoid having multiple nodes speaking at once
  - Otherwise collisions lead to garbled data
  - Need distributed algorithm for sharing channel
  - Algorithm determines which node can transmit

- **Three classes of techniques**
  - **Channel partitioning**: divide channel into pieces
  - **Taking turns**: scheme for trading off who gets to transmit
  - **Random access**: allow collisions, and then recover
    - More in the Internet style!

Link Layer (MAC) Protocol

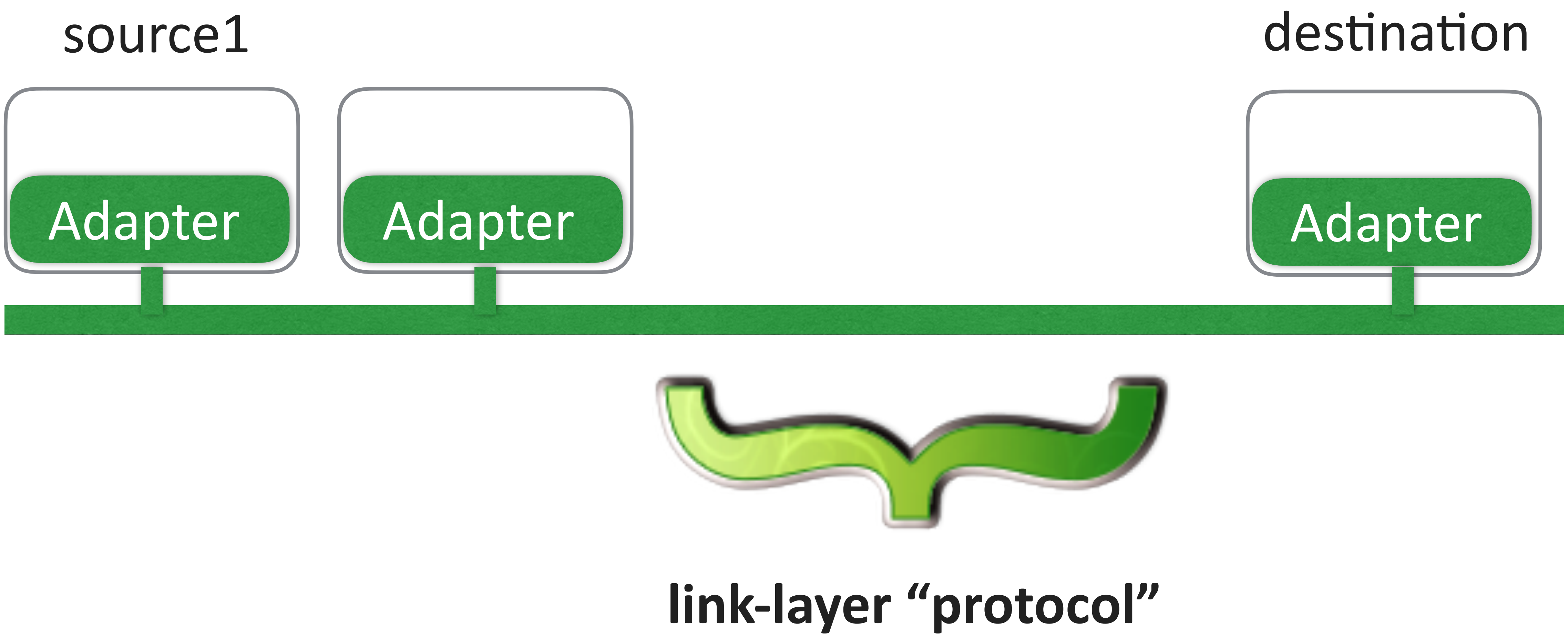
- **When node has packet to send**
  - Transmit at full channel data rate
  - No a priori coordination among nodes
- **Two or more transmitting nodes => collision**
  - Data lost
- **Link-layer protocol specifies:**
  - How to detect collision
  - How to recover from collisions

LETS TRY AGAIN!

Group Exercise:

Multiple source-destination pairs

Design a protocol that allows sharing the broadcast medium



CSMA (Carrier Sense Multiple Access)

- CSMA: **listen** before transmit
    - If channel sensed idle: transmit entire frame
    - If channel sensed busy: defer transmission
- Human analogy: don't interrupt others!
- Does this eliminate all collisions?
    - **No**, because of nonzero propagation delay

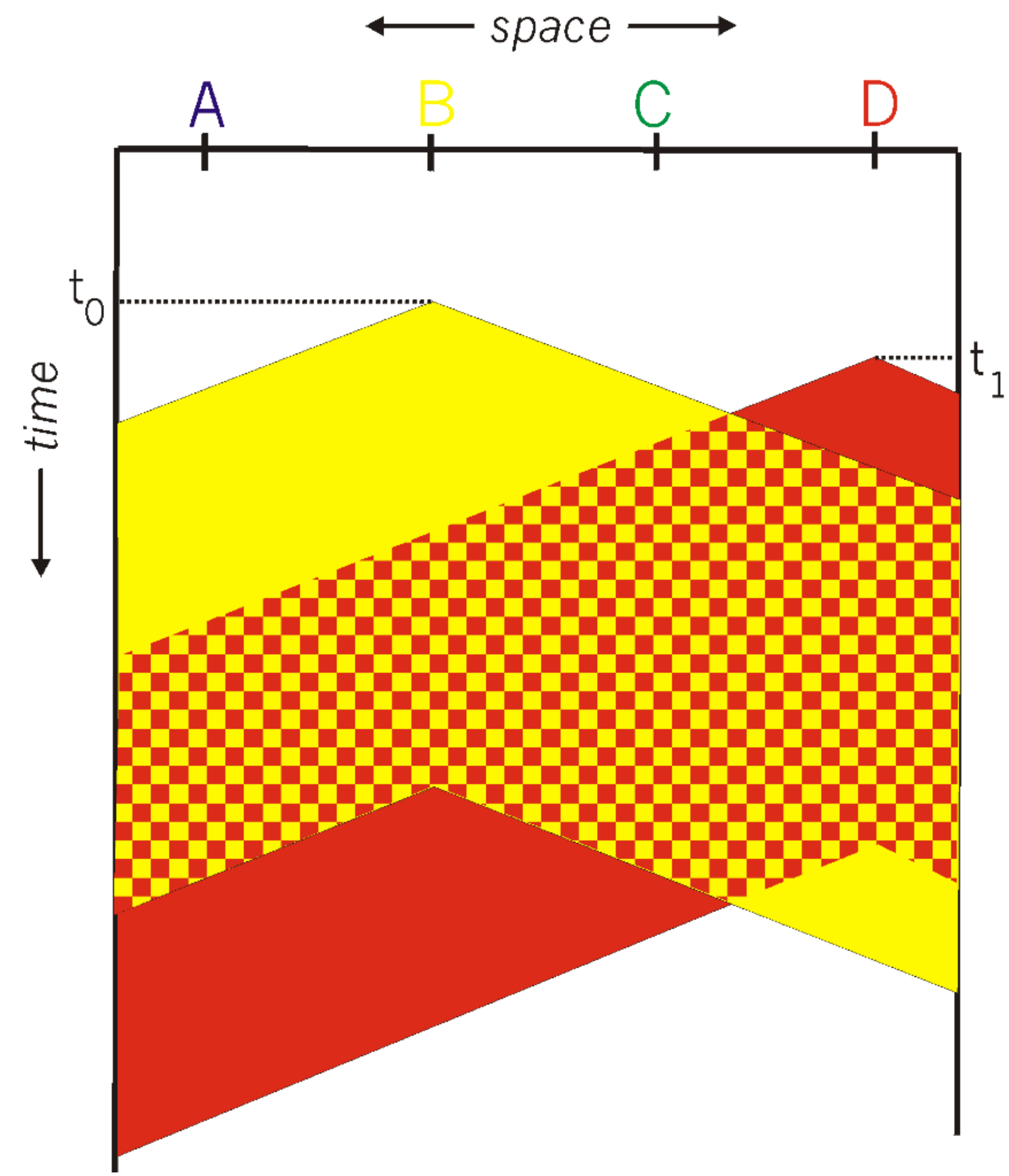
CSMA Collisions

Propagation delay: two nodes may not hear each other's before sending.

CSMA reduces but does not eliminate collisions.

Biggest remaining problem?

Collisions still take the full transmission slot!



CSMA/CD (Collision Detection)

- CSMA/CD: carrier sensing, defering as in CSMA
    - **Collisions detected within short tine**
    - Colliding transmissions aborted, reducing wastage

- Collision detection easy in wired (broadcast) LANs
    - Compared transmitted, received signals

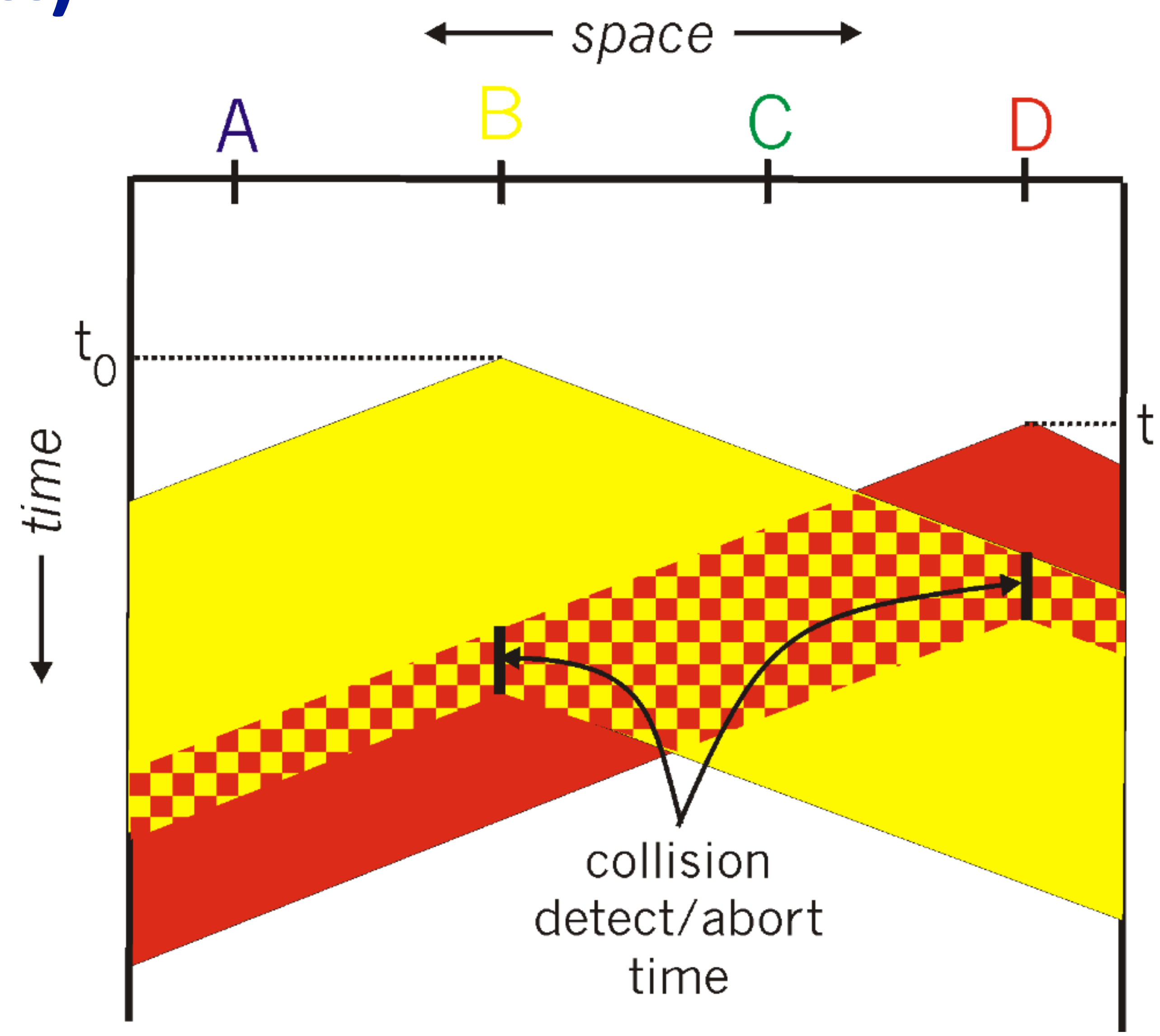
- Collision detection difficult in wireless LANs

CSMA/CD (Collision Detection)

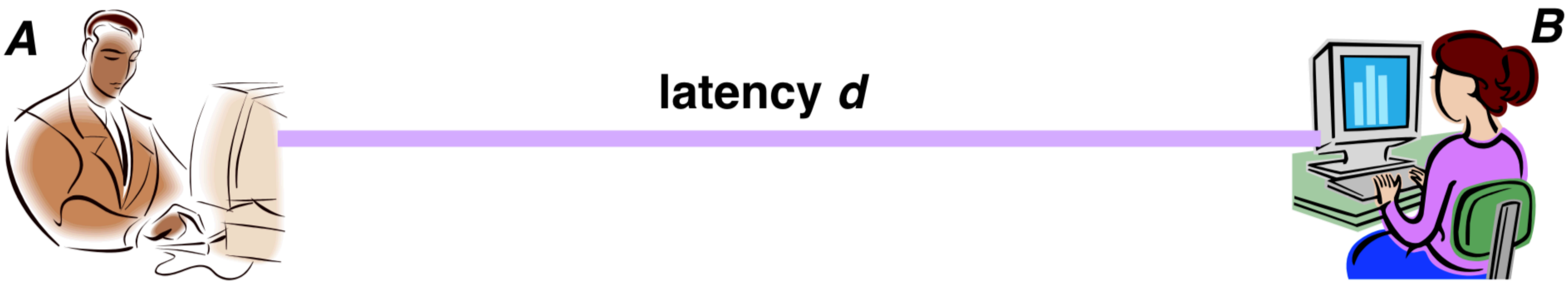
B and **D** can tell that collision occurred.

Note: for this to work, need restriction on **minimum frame size** and **maximum distance**.

Why?



Limits on CSMA/CD Network Length

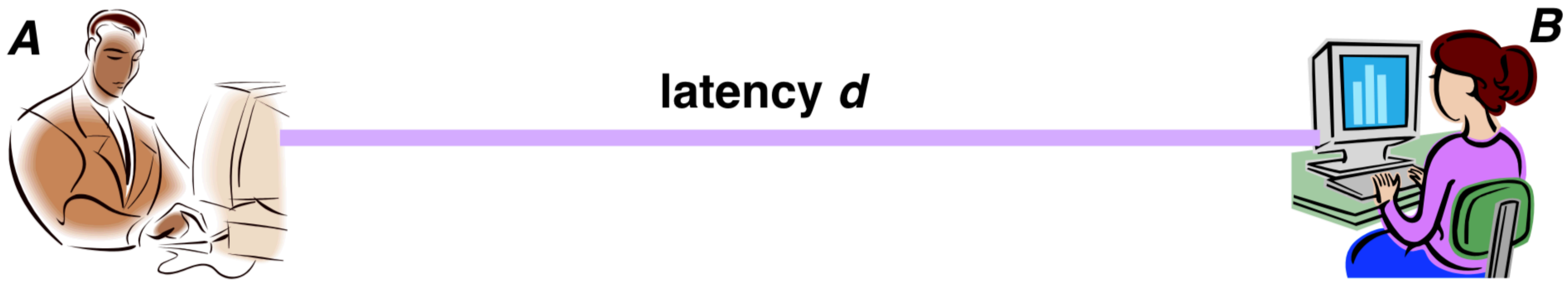


- **Latency depends on physical length of link**
  - Time to propagate a packet from one end to the other

- **Suppose *A* sends a packet at time *t***
  - And *B* sees an idle line at a time just before ***t+d***
  - … so *B* happily starts transmitting a packet

- ***B* detects a collision, and sends jamming signal**
  - But A can’t see collision until ***t+2d***

Limits on CSMA/CD Network Length



- **A needs to wait for time 2d to detect collision**
  - So, A should keep transmitting during this period
  - … and keep an eye for a possible collision
- **Imposes restrictions. E.g., for 10 Mbps Ethernet:**
  - Maximum length of the wire: 2500 meters
  - Minimum length of a frame: 512 bits (64 bytes)
    - 512 bits = 51.2 μsec (at 10 Mbit/sec)
    - For light in vacuum, 51.2 μsec ~ 15,000 meters
      - vs 5,000 meters “round trip” to wait for collision
- **What about 10Gbps Ethernet?**

How long should you wait?

- After collision when should you resend?
- Should it be immediate
- Should it be a random number with a fixed distribution?

CSMA/CD (Collision Detection)

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, ..., 2^k-1)
  - (Exponentially increasing waiting times)

Performance of CSMA/CD

- Time wasted in collisions
  - Proportional to distance d

- Time spent transmitting a packet
  - Packet length p divided by bandwidth b

- Rough estimate for efficiency (K some constant)

$$E \sim \frac{\frac{p}{b}}{\frac{p}{b} + Kd}$$

- Note:
  - For large packets, small distances, E ~ 1
  - As bandwidth increases, E decreases
    - That is why high-speed LANs are switched

Evolution

- **Ethernet was invented as a broadcast technology**
    - Hosts share channel
    - Each packet received by all attached hosts
    - CSMA/CD for media access control

- **Current Ethernets are "switched" (<u>next lecture</u>)**
    - Point-to-point medium between switches;
    - Point-to-point medium between each host and switch
    - No sharing, no CSMA/CD
    - Uses "self-learning" and "spanning tree" algorithms for routing