CS 3640: Introduction to Networks and Their Applications

Fall 2023, Lecture 7: Medium Access Control Protocols

Instructor: Rishab Nithyanand
Teaching Assistant: Manisha Keim

Announcements

- Assignment 2 has been released. Due on 9/28.

Today’s class

1. Recap: Error detection in the link layer

2. Medium access control protocols

Error detection and correction in the link layer

- **What approaches have been used to detect errors in frames? What are the limitations of each?**
  - Send multiple copies
  - 1-d parity bits
  - 2-d parity bits
  - Compute mathematical functions of the frame payload.
    - Checksums and Cyclic Redundancy Checks (CRC)

- **How does a receiver communicate that a frame was error-free?**
  - The “ACK” frame. How is it sent?
    - Stop and wait.
    - Sliding windows.

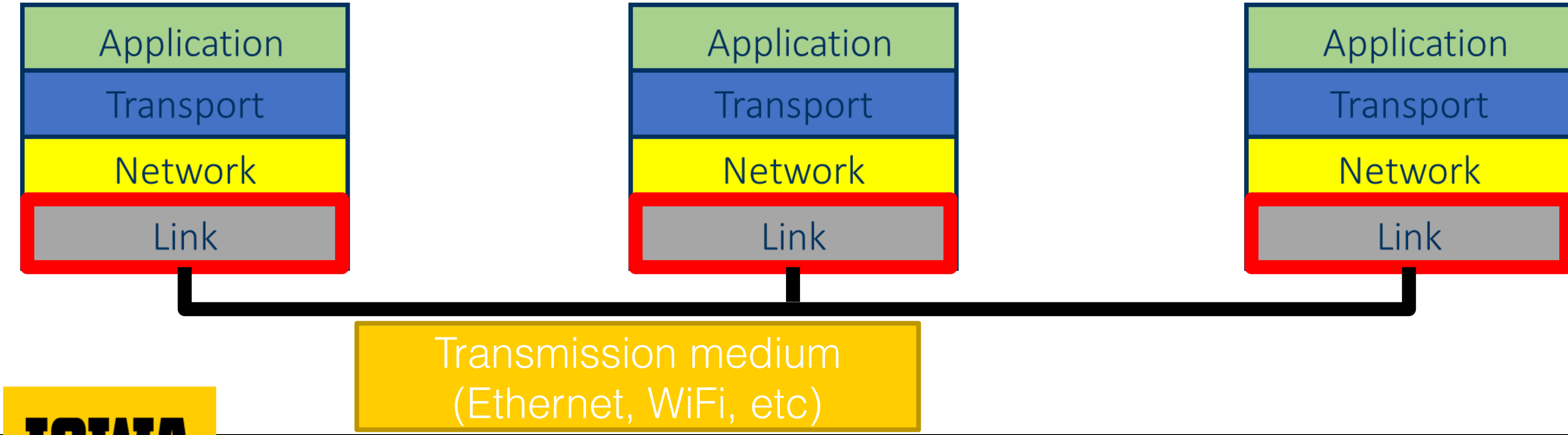
Today's class

1. Recap: Error detection in the link layer

2. Medium access control protocols

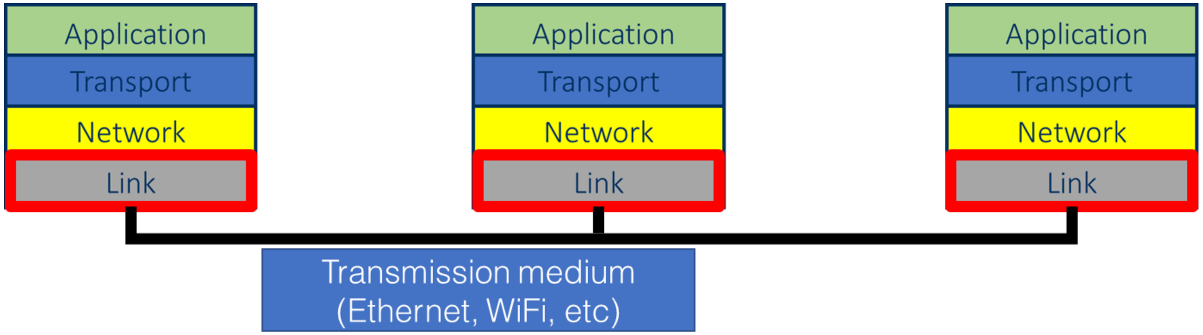
Medium Access Control (MAC) protocols

- **What is a MAC protocol?**
  - A transmission medium can be shared by many devices.
  - If everyone talks at the same time, we have “collisions” and unintelligible data.
  - MAC: Rules for sharing a common transmission medium.



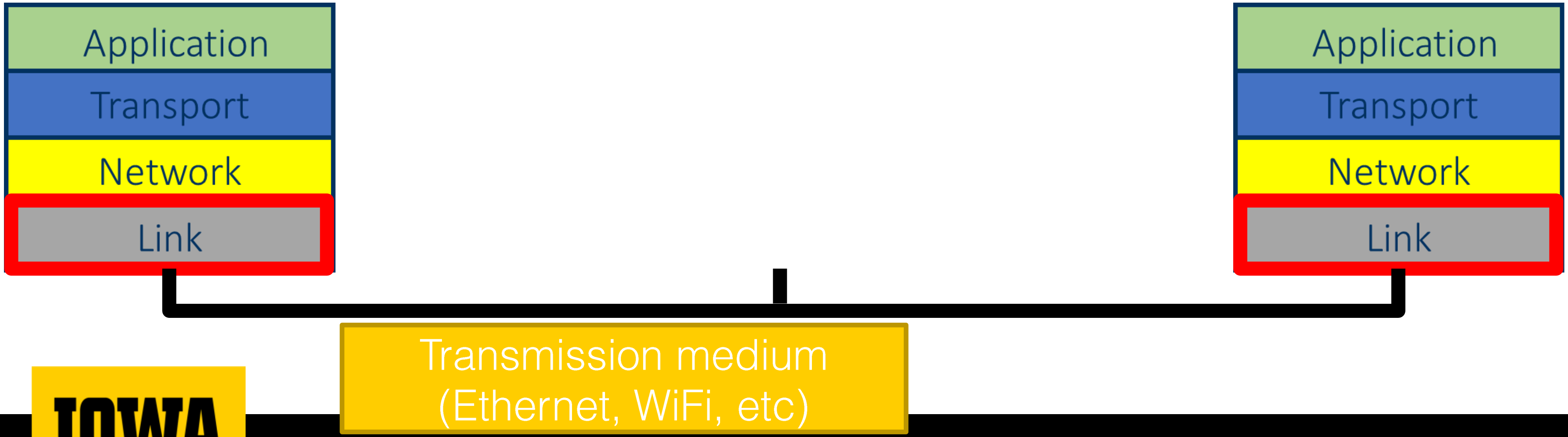
Medium Access Control (MAC) protocols

- **General strategies for MAC protocols**
  - Idea 1: Partition the transmission channel so each host has its share.
    - We briefly saw this – Time and Frequency division. Each host has a fixed share (time or frequency band) in the medium.
    - What if a host has nothing to send?
  - Idea 2: Pass a "transmit now" token to hosts.
    - Like me cold-calling someone to answer a question. (I'm giving you the token).
    - What if multiple people really want to give an answer?
  - **Problem**: Transmission channel utilization isn't great.



Medium Access Control (MAC) protocols

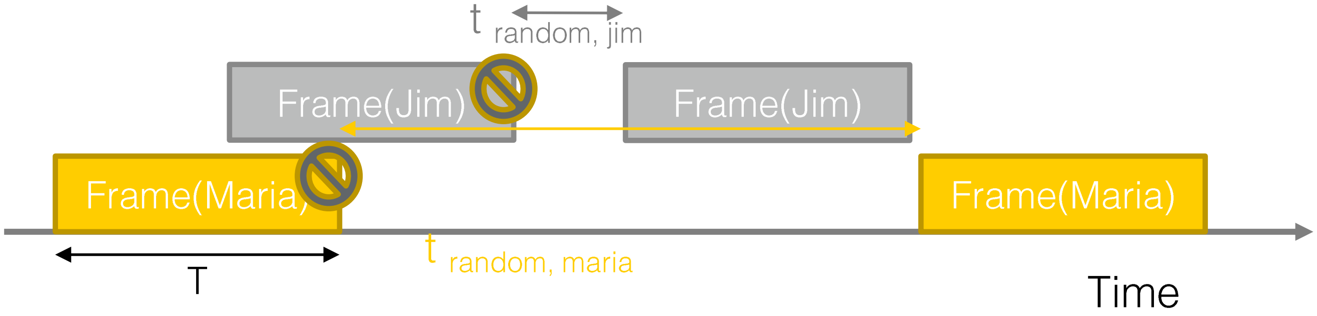
- **General strategies for MAC protocols**
  - Idea 3: Allow collisions, we'll figure out how to recover data.
    - Allows much higher utilization.
    - Now we have new problems:
      - How to identify when a collision has occurred.
      - How to recover from a collision.
    - This strategy is called "Random access MAC" or "Contention-based MAC".
    - Used by Ethernet, mobile transmission protocols, and others.



The ALOHA MAC protocol (circa 1970)

ALOHA: Additive Links On-line Hawaii Area (developed at UHawaii)

- **Idea: Send a frame as soon as you need to.**
  - If you receive a frame while transmitting, a collision has occurred.
    - Remember everyone using the same medium will receive all transmissions on that medium.
  - Wait for some random time and transmit again.
  - Eventually, a frame will get transmitted without collisions.
- **Assumptions:**
  - All frames are equally sized.
  - Errors (collisions) are detectable.
- **Problem**: For successful transmissions, no other frame from any other host should start within T time before or after you.
  - Sensitive transmission period: 2T for each frame.
  - Scales terribly. (Theoretical maximum throughput for large number of hosts with random transmission times: 18%).

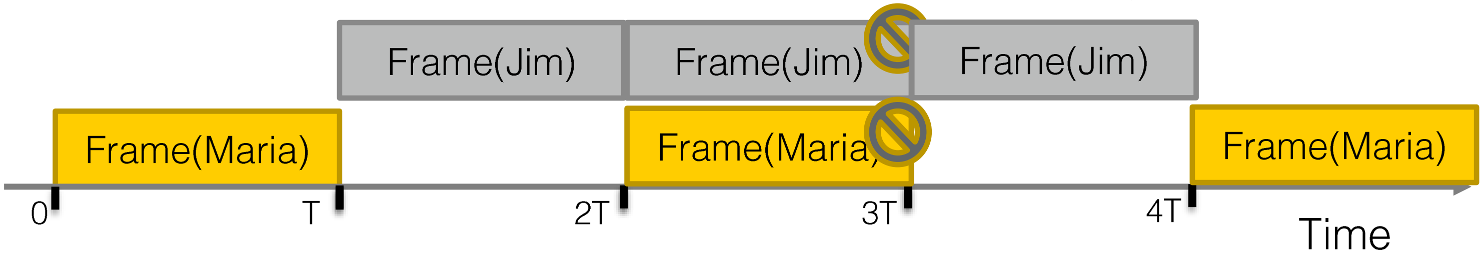


The Slotted ALOHA MAC protocol (circa 1970)

- **Idea: If we allow transmissions only at certain time points, the "sensitive" period for transmissions reduces.**
  - Transmissions on the channel can occur only every T seconds.
    - The channel is divided into slots, but anyone can send in any slot.
  - Sensitive transmission period: T for each frame.
  - If a collision occurs, wait some random number of time slots and retransmit.
- **Assumptions:**
  - All frames are equally sized.
  - All host clocks are synchronized (!!!)
  - Errors (collisions) are detectable.
- Sensitive period reduced from 2T to T.
  - Theoretical maximum throughput is doubled to 36%.
  - Still not great.

Slots $_{\text{random, jim}}$ = 0

Slots $_{\text{random, maria}}$ = 1



The CSMA MAC protocol (circa 1990)

- **Carrier Sense Multiple Access (CSMA).**

- **Idea: Check to see if the medium is being used by someone else before you try to use it. Start transmission only if the medium is idle.**
  - Basically, be polite and don’t try to talk over someone else.
  - If someone is talking, wait for some time and check again.

- Removes the need for synchronized clocks.

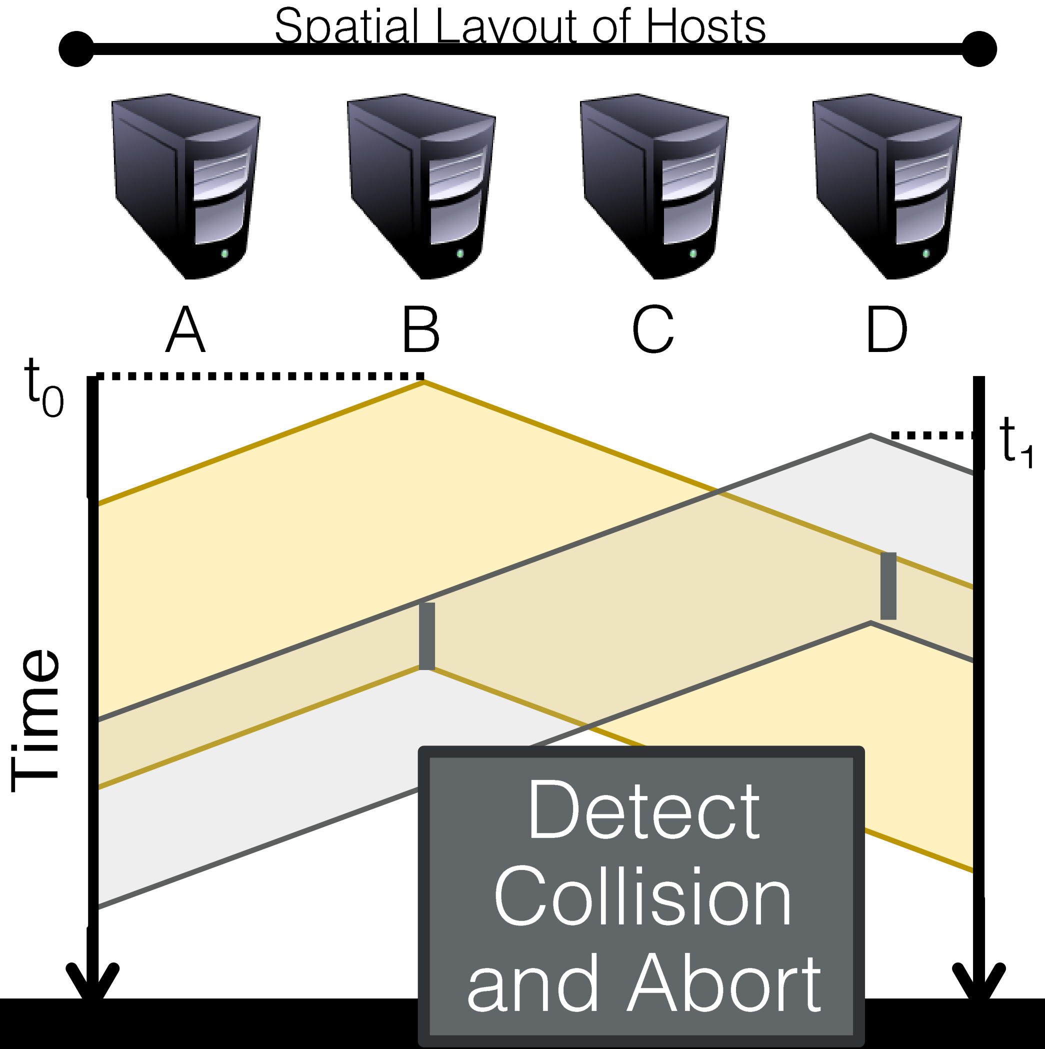
- Collisions can still occur.
  - The propagation delay may mean you don’t know that someone was talking until after you started talking.

The CSMA/CD MAC protocol (circa 1990)

- **Carrier Sense Multiple Access with Collision Detection (CSMA/CD).**
  - Currently used in Ethernet (802.3) and other wired networks.
  - Why wired?
    - Carrier sensing is much easier than in wireless networks.
- **Idea: While sending a frame, sense the medium for a collision. If a collision occurs, then abort immediately and notify the others. Retry after some time.**
  - Why keep sending when you know its corrupted.
- Collisions can occur, but we can reduce the cost of one by quickly detecting it and stopping transmission.

The CSMA/CD MAC protocol (circa 1990)

- Collisions can occur
- Collisions are quickly detected, aborted, and reported (using a Jam sequence).
- Note the role of distance, propagation delay, and frame length



Deterministic vs. randomized back-off

- **I’ve been saying “wait for some time before trying again”.**
  - What is “some time”?

- Two general approaches:
  - **Deterministic**: I’ll always wait “*t*” seconds.
  - **Randomized**: I’ll wait for some random time between 0 and t seconds.

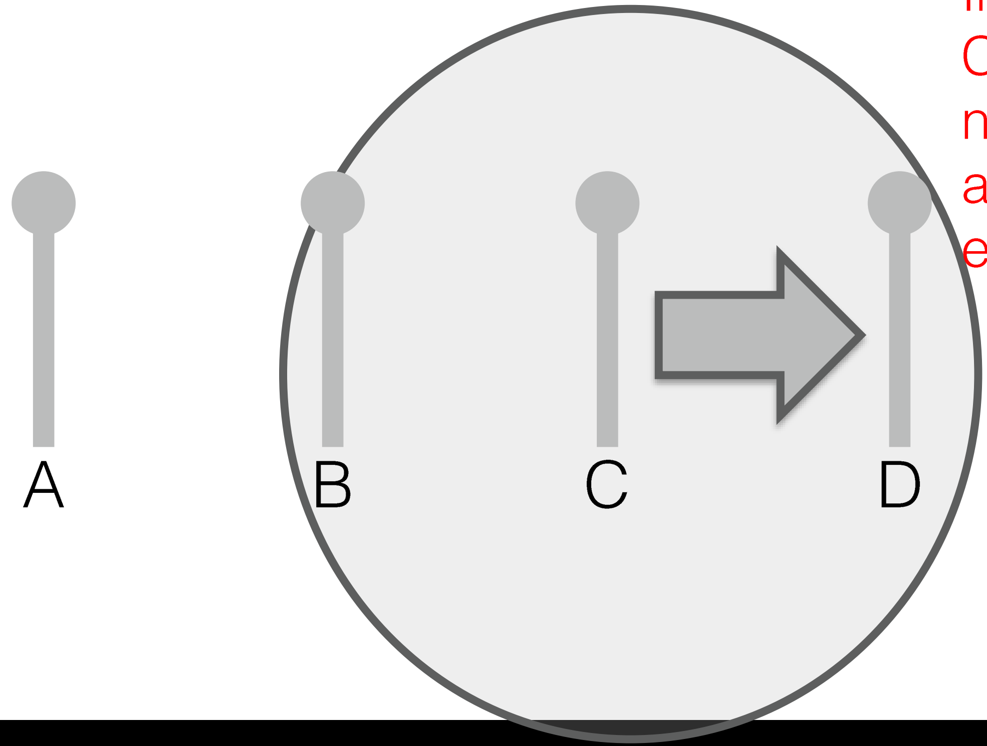
- **Discuss: Which is better?**
  - Randomized is better.
  - Usually, if you’re trying again, it means a collision was detected. If it was detected by you, it was also detected by everyone else. If two hosts have the same “*t*” and collide once, then they will always collide.

Randomized back-off

- **Randomized back-offs**
  - If you need to retransmit, select a random time *t* in [0, *T*].
  - Retry after waiting for *t* (milli/micro) seconds.
  - **Discuss**: How should you change *t* when you have collisions occurring even on the retry?
- **Exponential randomized back-off**
  - **Key idea**: If collisions keep occurring, it means that the channel is really busy. Trying again only makes the problem worse. Let's back-off exponentially.
    - Keep doubling *T* for each successive collision.
    - If you try to send a frame the first time and it collides, select *t* from [0, *T*].
    - If you have c successive collisions on your retries, select *t* from [$0, T, \ldots, (2^c\text{-}1)T$].

Challenges with the wireless medium

- C is transmitting to D.
- **Discuss: What happens when C tries to sense for collisions?**
  - Its own transmission dominates any signal it can sense. This means it cannot sense the carrier while transmitting.

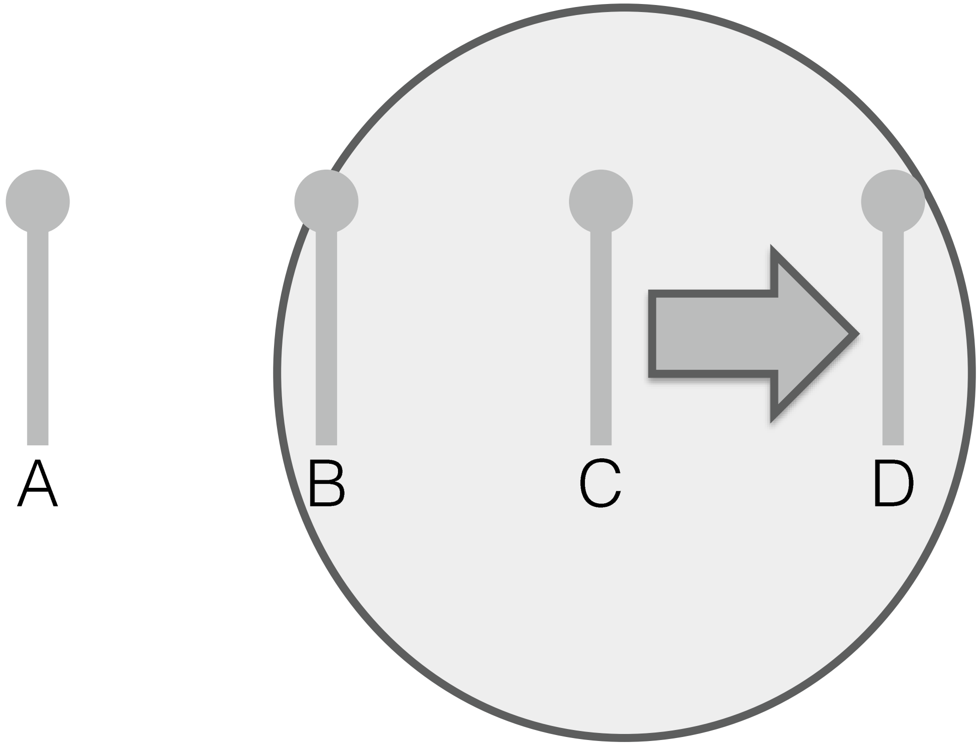


Implication: Collision detection needs to happen at the receiver end!

Challenges with the wireless medium

- C is transmitting to D.
- **Discuss: What happens when A senses the carrier?**

Implication: Carrier sensing is not always accurate because of connectivity issues.



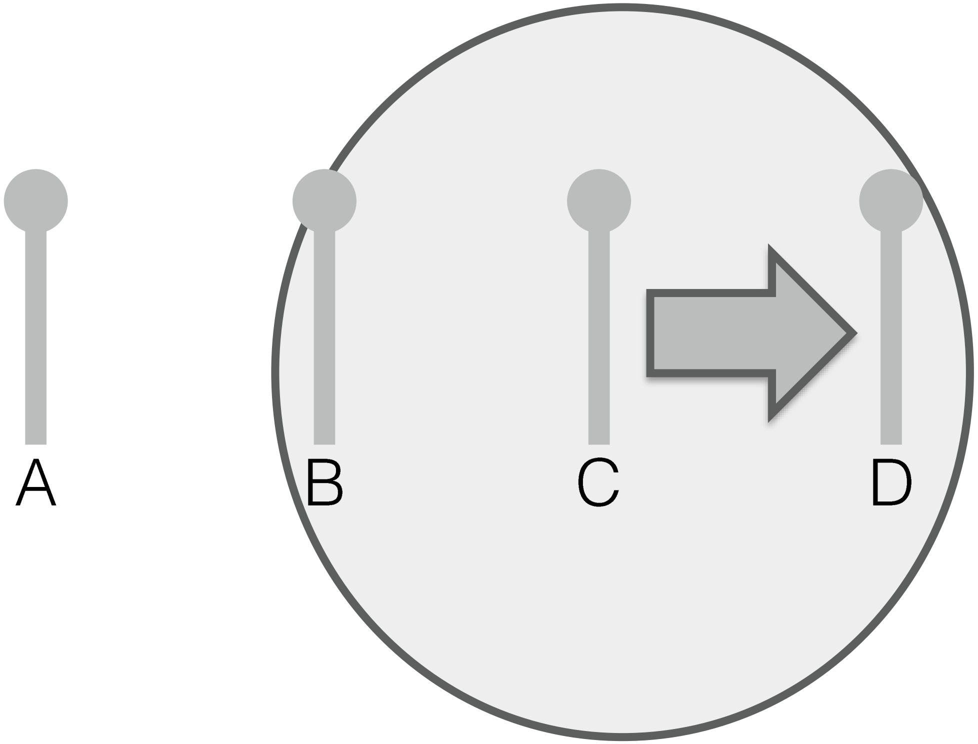
The CSMA/CA MAC protocol (circa 1990)

- **Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA).**
  - Currently used in wireless transmission protocols (WiFi 802.11).

- In wireless networks:
  - Carrier sensing while transmitting is not feasible, so collisions can only be detected at the receiver.
  - Detecting collisions is harder because accurately sensing the medium is tough – wireless signals may not carry to far away hosts.
    - Connectivity is not transitive: If A can reach B and B can reach C, it doesn’t mean that A can reach C.
  - Instead, most wireless networks just try to avoid collisions instead of detecting and retransmitting. They use the CSMA/CA protocol to do this.

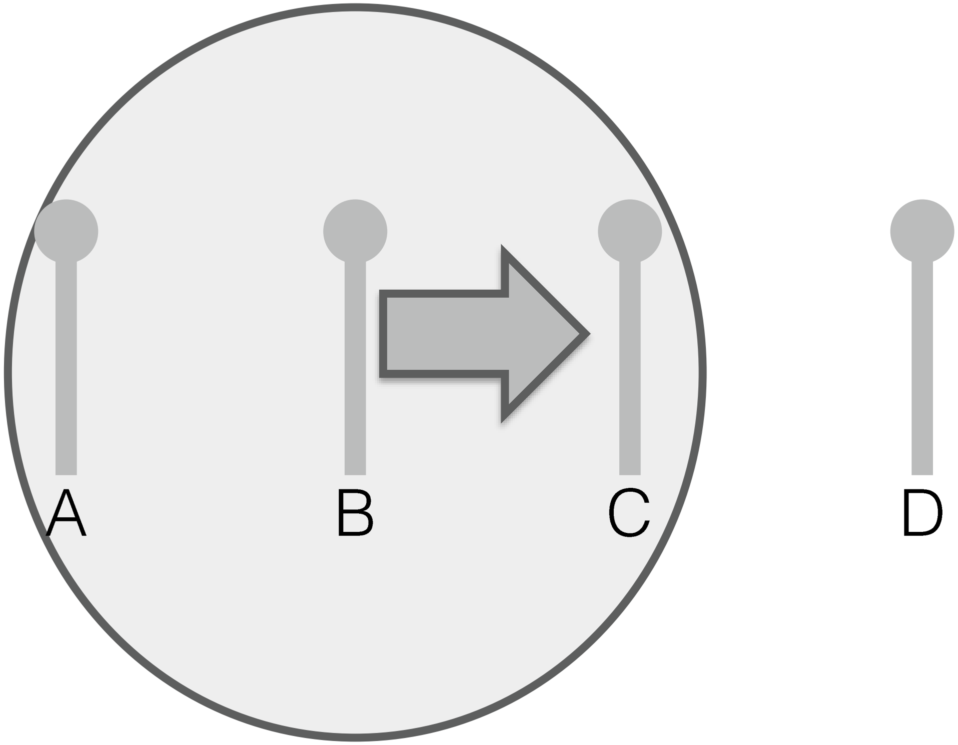
The CSMA/CA MAC protocol (circa 1990)

- B wants to transmit to C.
  - Sense the carrier. Is another node transmitting?
    - Yes. So do an exponential back-off before trying again.



The CSMA/CA MAC protocol (circa 1990)

- B wants to transmit to C.
  - Sense the carrier. Is another node transmitting?
    - No. Send frames. If I don’t get an ACK in reasonable time, start again.

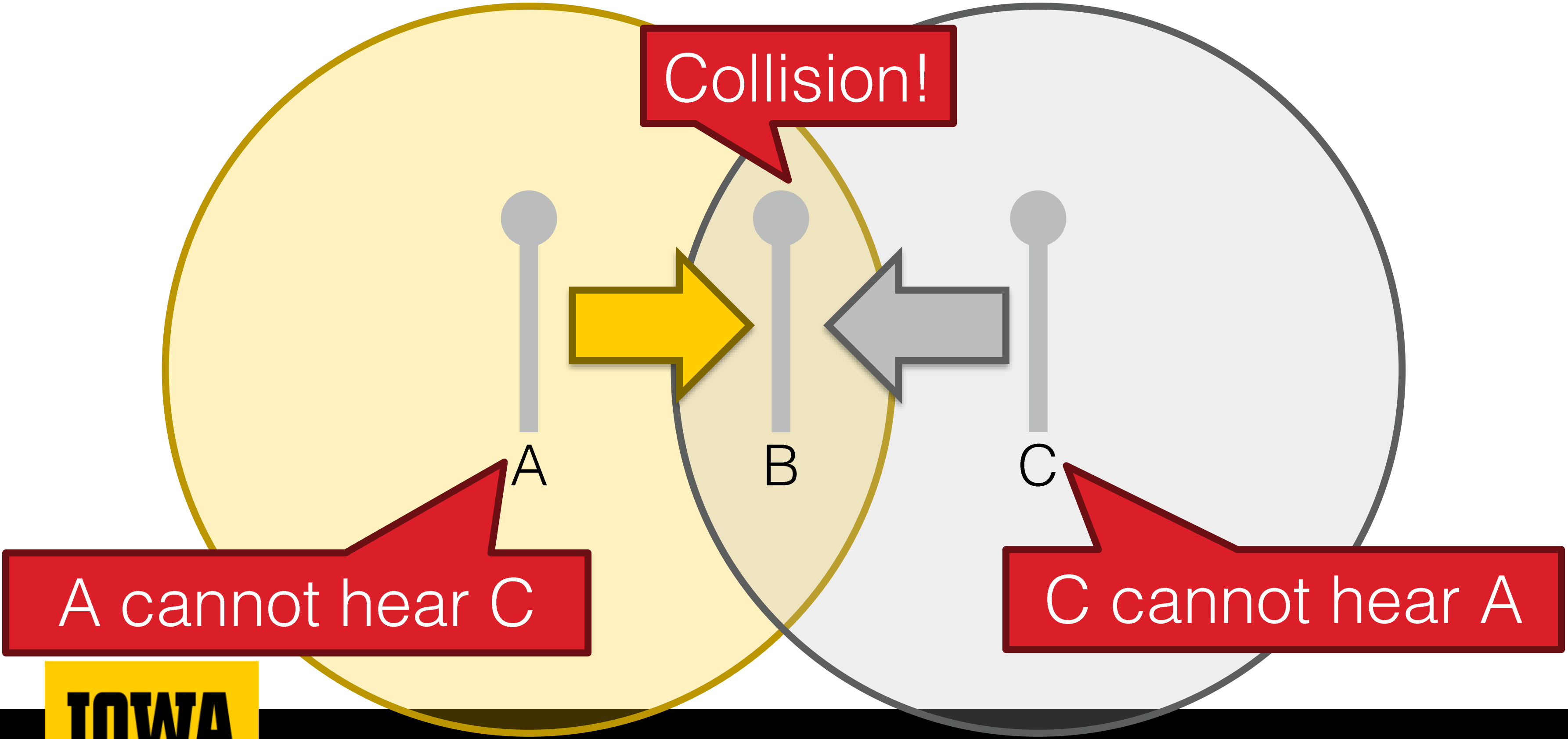


The CSMA/CA MAC protocol (circa 1990)

- Step 1:
  - Sense the carrier. Is another node transmitting?
- Step 2:
  - If the carrier is not busy: Send the frames.
  - If the carrier is busy: Do an exponential back-off and go to step 1.
- Step 3:
  - Wait for an acknowledgement. If it doesn’t arrive after “timeout” seconds, go to step 1 and try again.
- **Discuss: What scenarios might result in the CSMA/CA algorithm having more false-positives (thinking that the channel is busy when it isn’t) or false-negatives (thinking that the channel is free when it isn’t)?**
  - Hint: Think about the non-transitivity of connectivity.

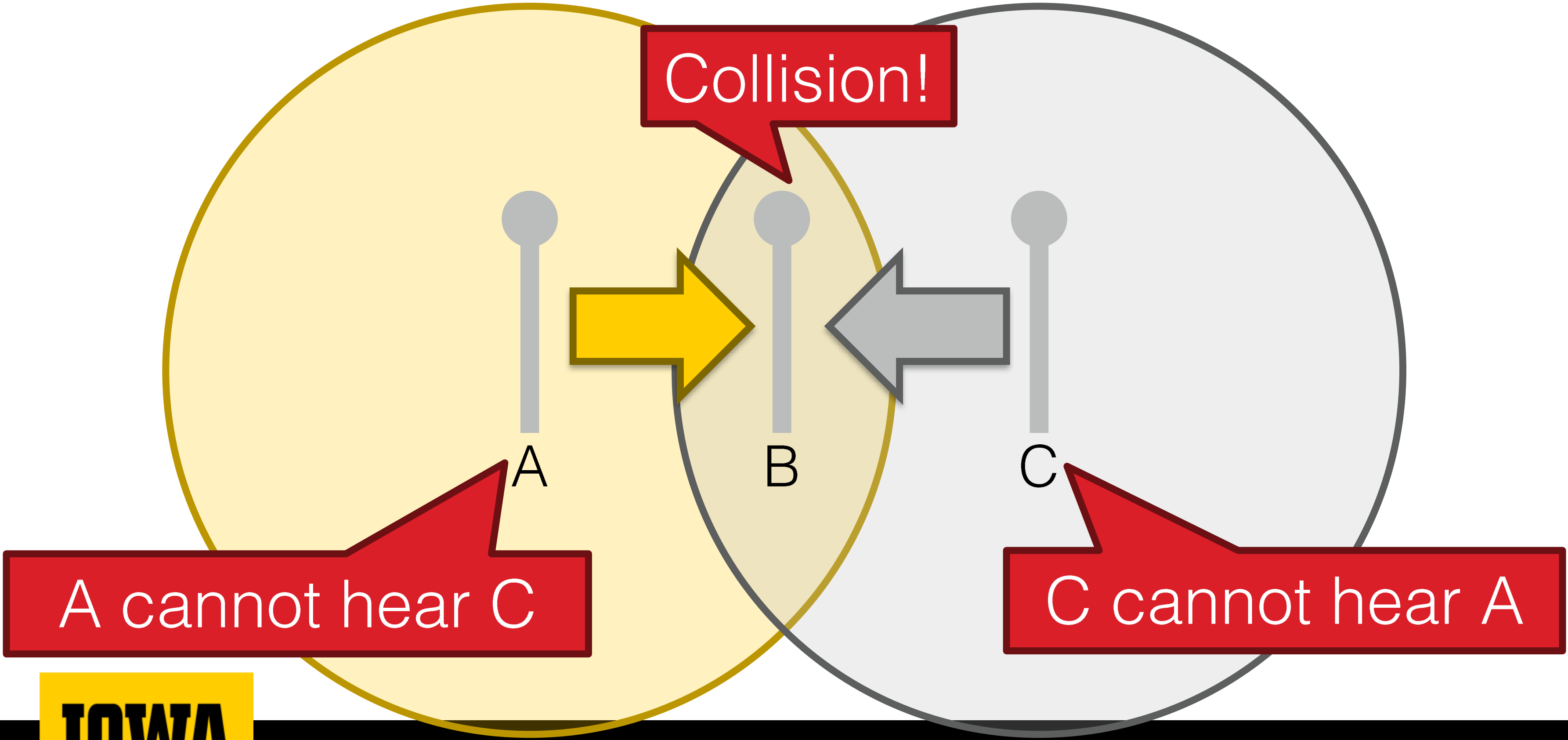
CSMA/CA and the hidden terminal problem

- Step 1: Sense the carrier. Is another node transmitting?
  - Hidden nodes can increase false-negatives (thinking the carrier is free when it is actually busy).

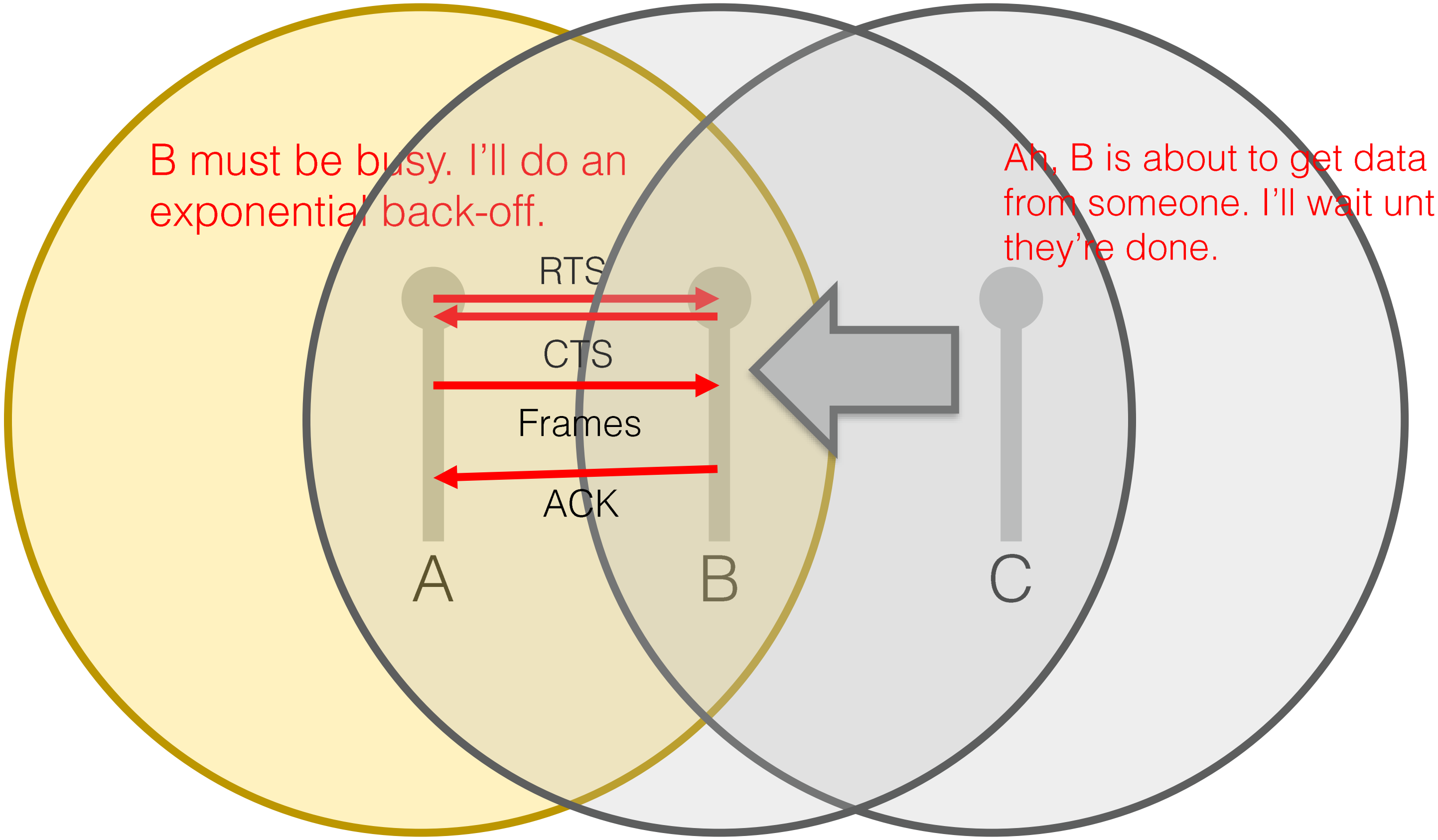


CSMA/CA and the hidden terminal problem

- **Discuss: Would asking for permission from the receiver help?**



CSMA/CA with RTS/CTS



B must be busy. I'll do an exponential back-off.

Ah, B is about to get data from someone. I'll wait until they're done.

CSMA/CA with RTS/CTS

- Step 1: Sense the carrier. Is another node transmitting?

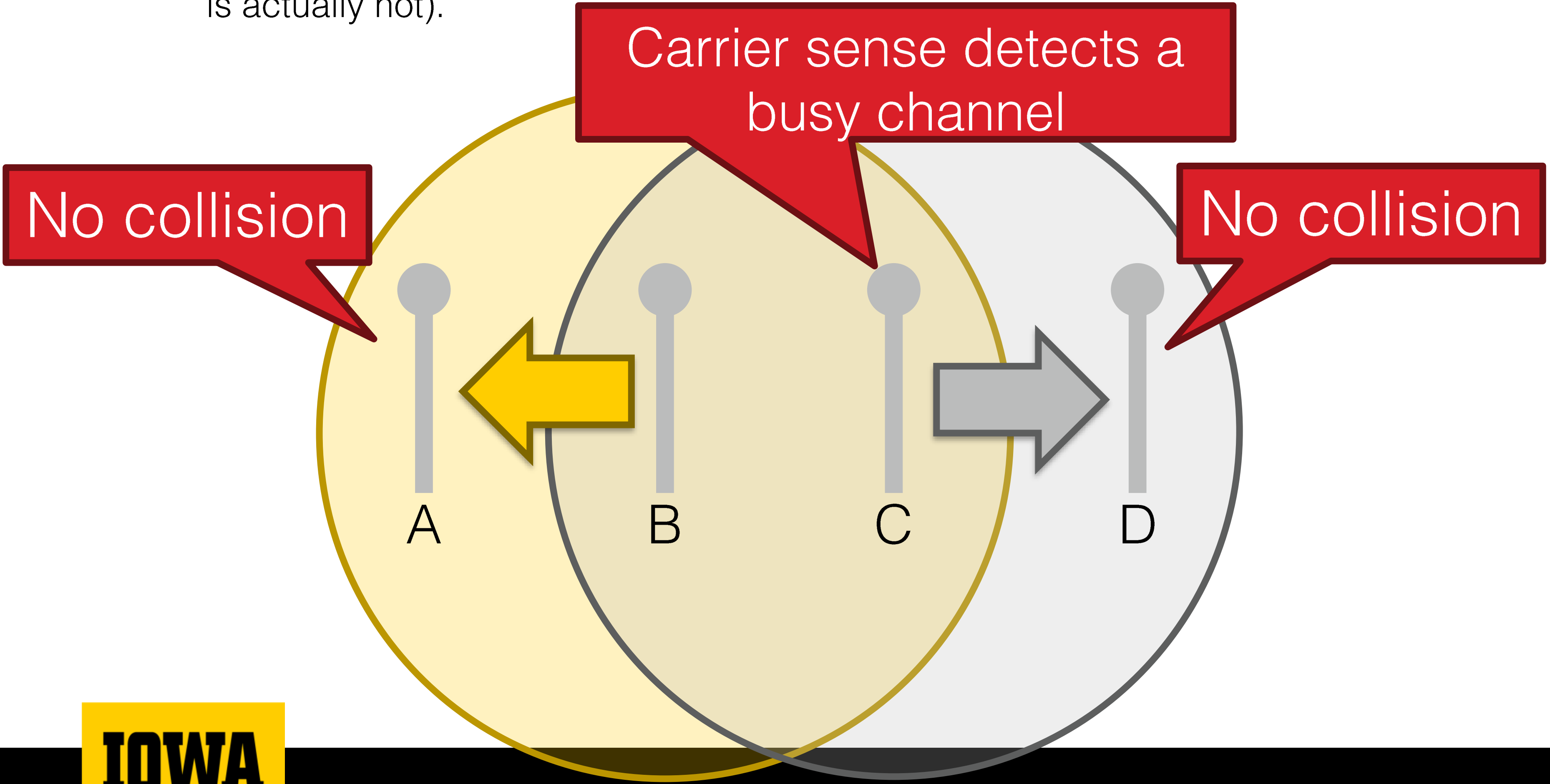
- Step 2:
    - If the carrier is not busy: Tell the receiver you have something to send via a “Request To Send” (RTS) message. Wait for “timeout” seconds for a “Clear To Send” (CTS) message from the receiver.
        - If CTS arrives: send the frames.
        - If CTS doesn’t arrive: do an exponential back-off and go to step 1.
    - If the carrier is busy: Do an exponential back-off and go to step 1.

- Step 3: Wait for an acknowledgement. If it doesn’t arrive after “timeout” seconds, go to step 1 and try again.

- **Solves the hidden terminal problem, doesn’t help with exposed terminals.**

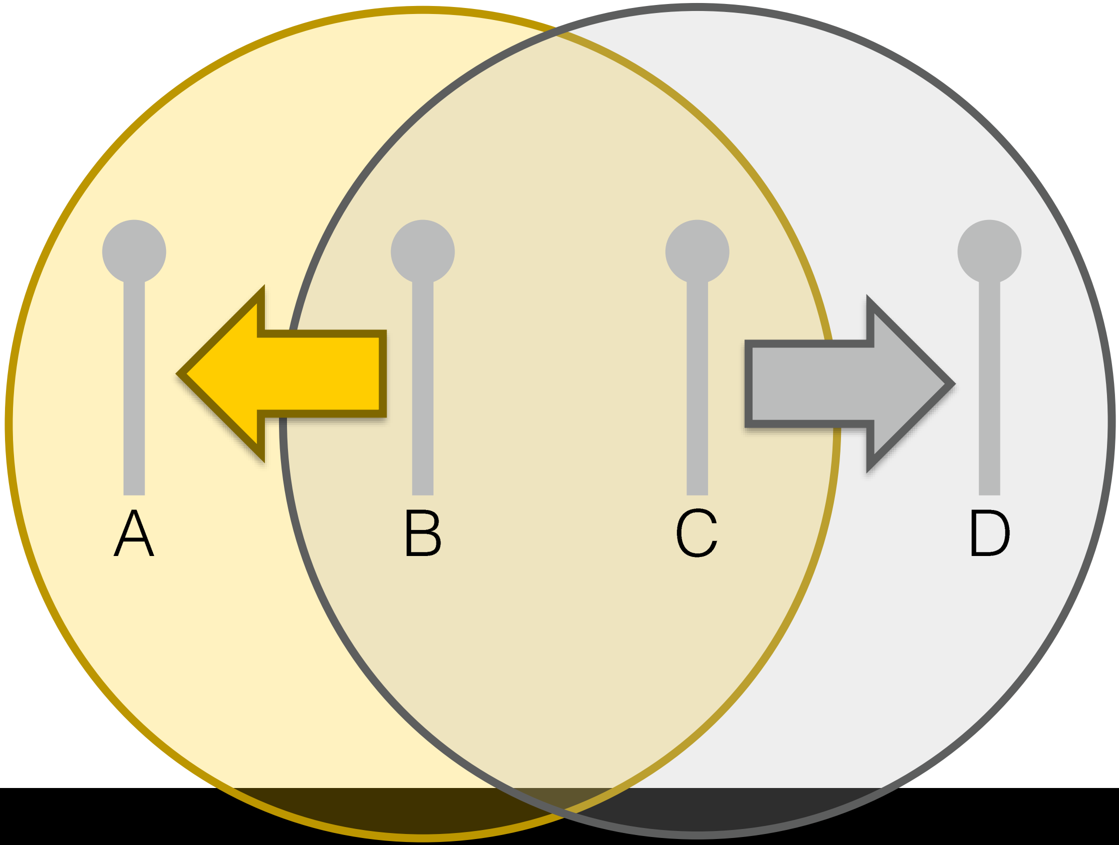
CSMA/CA and the exposed terminal problem

- Step 1: Sense the carrier. Is another node transmitting?
  - Exposed terminals can increase false-positives (thinking the carrier is busy when it is actually not).



CSMA/CA and the exposed terminal problem

- **Discuss: Why doesn’t RTS/CTS messaging work here?**
  - RTS/CTS helps reduce false-negatives (thinking that the medium is free when it isn’t) but doesn’t prevent false-positives (thinking the medium is busy when it isn’t) because we never actually reach the RTS stage when we have a false-positive.



CSMA/CA and the exposed terminal problem

- **Discuss: Why not skip sensing and just start with an RTS message if it solves the exposed terminal problem?**
    - Overhead!
    - RTS/CTS frames don't have any useful data in them. As your network gets faster, you send more RTS/CTS frames per second because you can handle more transmissions per second.
        - RTS/CTS overhead on a 1Mbps WiFi connection: 4%
        - RTS/CTS overhead on a 11Mbps WiFi connection: 25%
    - Sending RTS/CTS frames when you don't need to will only make this worse.
    - **Lesson: Everything is a trade-off!**
        - Researchers and engineers decided that the overhead from this approach was too high to be a feasible solution to the exposed terminal problem.

Things to remember from this lecture

- What is the general idea behind CSMA protocols?
  - Sense the medium before transmitting.

- Why are wireless media more challenging?
  - Collisions can only be detected at the receiver and carrier sensing is not always accurate.
  - The hidden and exposed terminal problems.

- Why do we need a CSMA/CD and CSMA/CA protocol?
  - Collisions can still occur due to transmission delays, CSMA/CD detects these and improves throughput by stopping transmission as soon as it is detected.
  - CSMA/CA tries to avoid collisions entirely by requesting permission from the receiver before transmission.