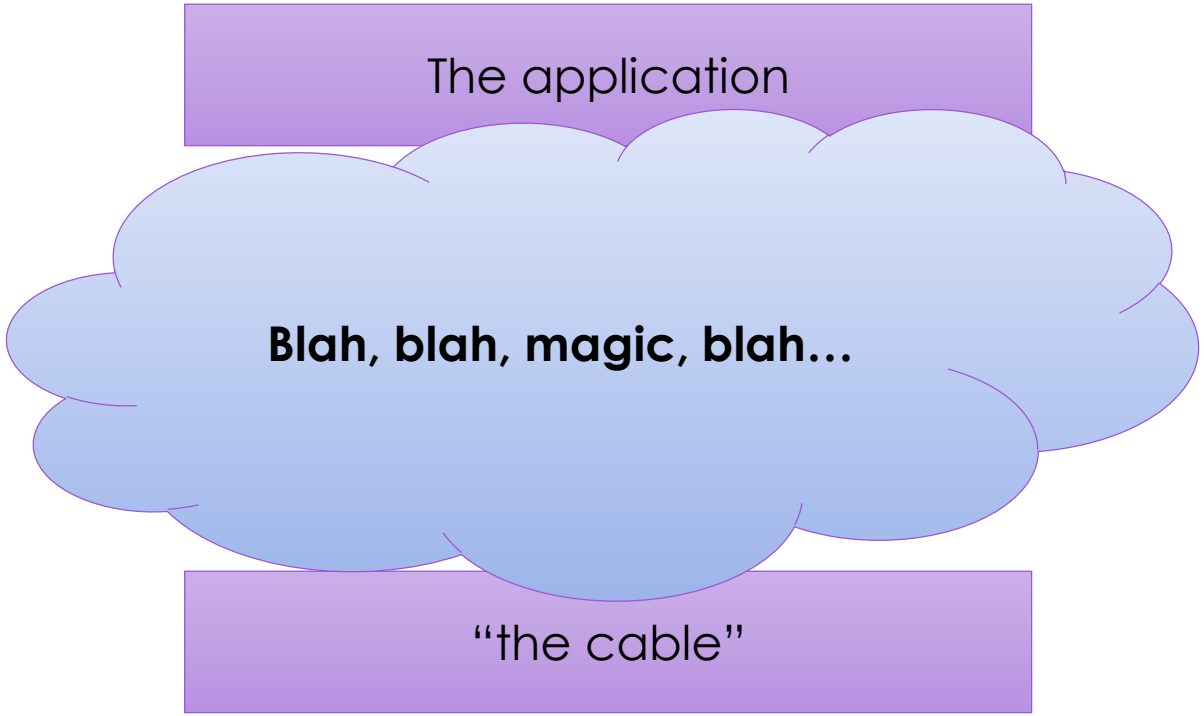
COMP3310/6331 – #6

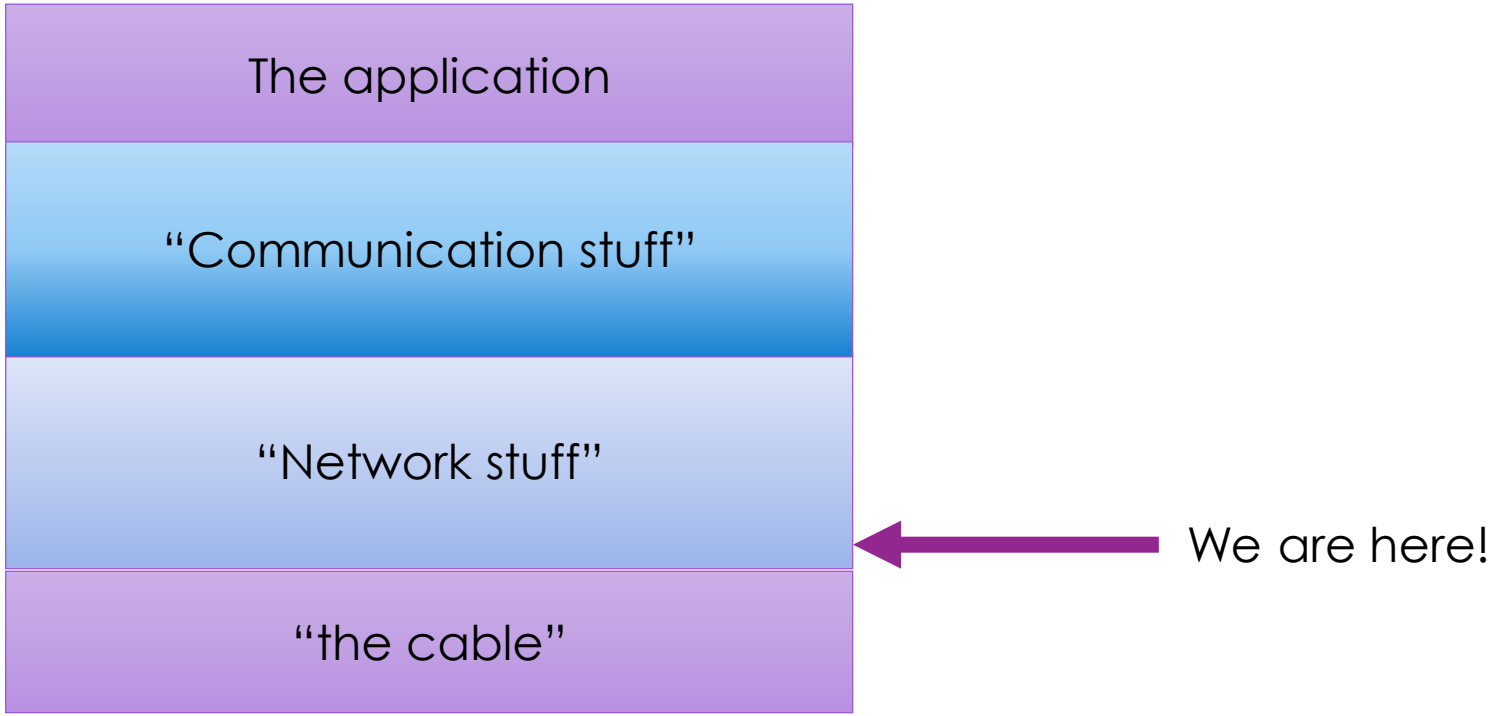
LANs!

Dr Markus Buchhorn: *markus.buchhorn@anu.edu.au*

We're moving up!



Above the cable



Definitions

- LAN = Local Area Network
- MAN = Metropolitan Area Network
- WAN = Wide Area Network
- SAN = Storage Area Network (cf NAS…)
- PAN = Personal Area Network
- …

- Why start with any of these?

LAN

- Ignore the arbitrary 'distance' classification
- LAN is where everything starts
  - You connect devices to a LAN
  - First networks were point-to-point links (2 devices)
  - First useful network had many devices, connected to a common network
  - Started from shared medium/bus topology

- Leads to 2 better concepts:
  - **Administrative** domain – one standard (media, modulation, encoding, …)
  - **Broadcast** domain – my device can talk to anybody on the LAN (without help)

Guarantees

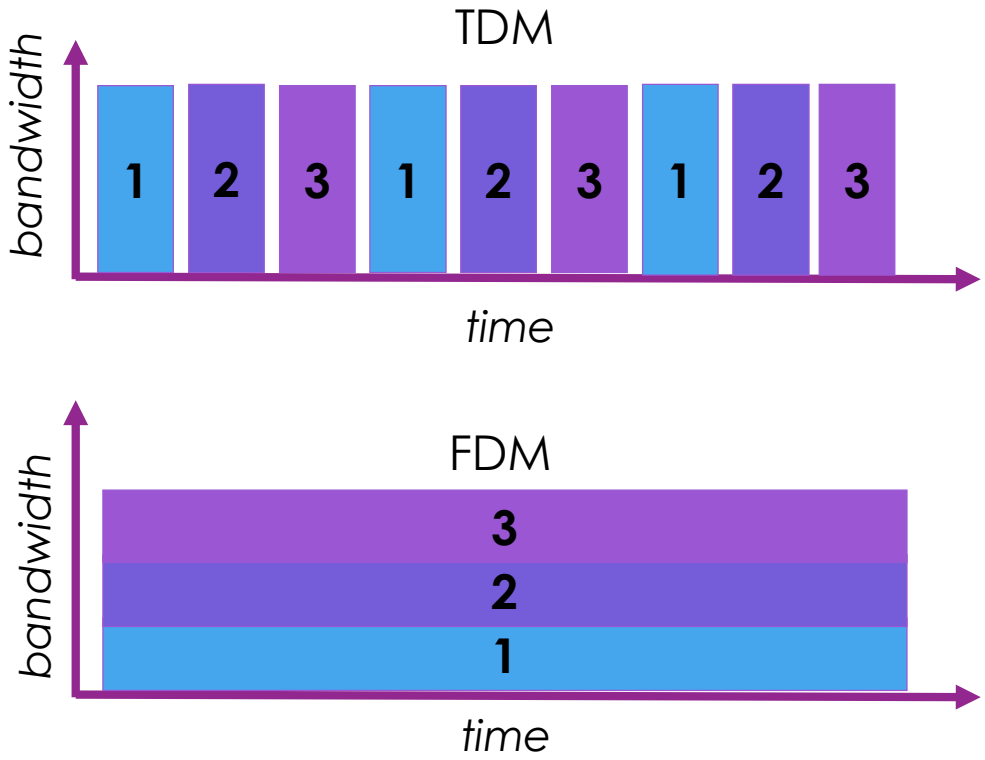
- A good LAN design is really simple
- Get a message from here to there, as fast/efficiently as you can
  - Over a particular ‘cable’

- **No** guaranteed delivery
- **No** built-in error-detection/correction
- **No** specialised features (bulk-transfers, realtime, …)

- Leave all those hard things to the software – in general

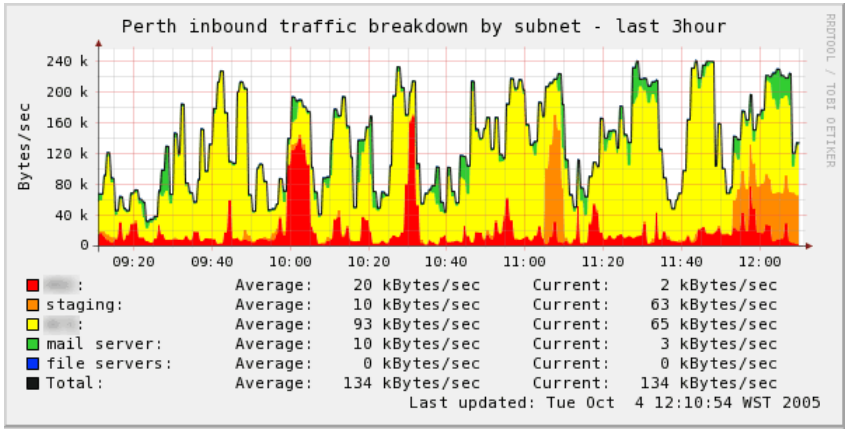
Multiplexing

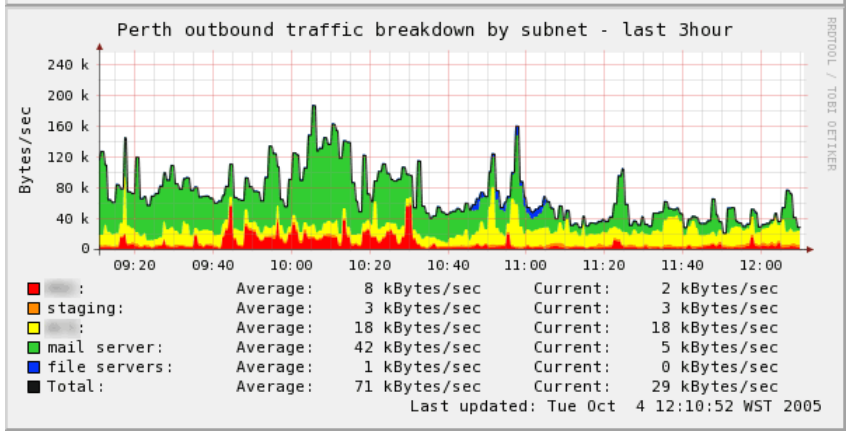
- Multiplexing:
  - By time
  - By space
  - By frequency
- Everyone gets an equal share
- But not everybody needs that much
  - It's effectively a circuit
  - Wasting capacity



Statistical Multiplexing

- Demand for capacity varies with time
  - Random access-priority across all devices
  - Statistically:
    - Don't need all of the bandwidth, all of the time
- Share the bandwidth fairly and easily
  - Whoever needs to, can send, when they want
  - *Probably need to control that…*
  - However much you want to send, you can send it all
  - *Definitely need to control that*





Circuits, Cells and Frames

- **Circuits** – a fixed pipe, just for you, tied to both endpoints
  - Send bits whenever you want, to the other end
- **Cells** – Time Division Multiplexing
  - Send a specific number of bits, when it's your turn, to the other end
- **Frames** – arbitrary length, targeted messages
  - Send bits as and/when you want ?!?

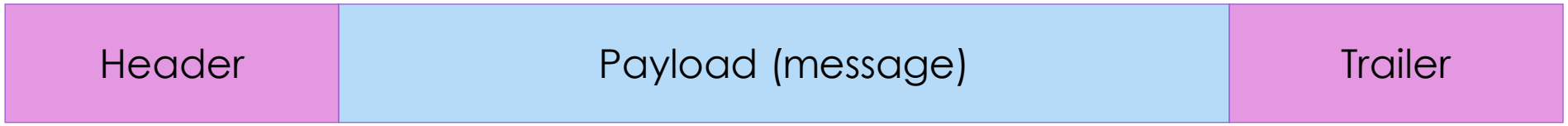
Designing a Frame

- Need to specify where it's going = destination address
- Need to specify where it's come from = source address
- Need to specify where it's assemblage of bits starts and stops

- Need to agree how long a frame could be
  - Infinitely long is not acceptable
  - Don't
    - hog,
    - use up all the memory at the receiver,
    - let errors pile up

- Need to agree how to access the network fairly

Building a Frame



- Header: "**Hey**!!! *I'm* sending this **long** message to over *there*"
- Trailer: "I'm done"

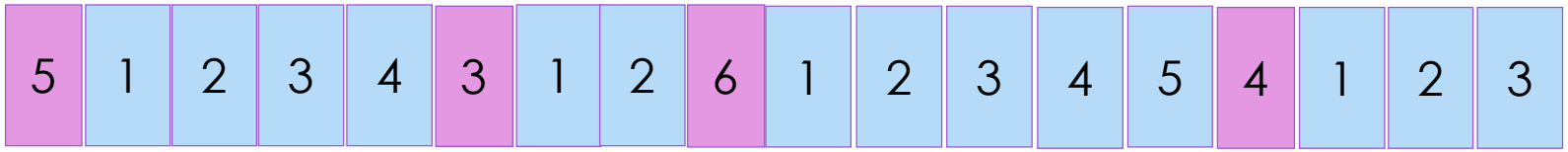
  - "Hey" = Start of Frame
  - "I" = Source address
  - "long" = Frame length indication
  - "there" = Destination address

Simple Frame

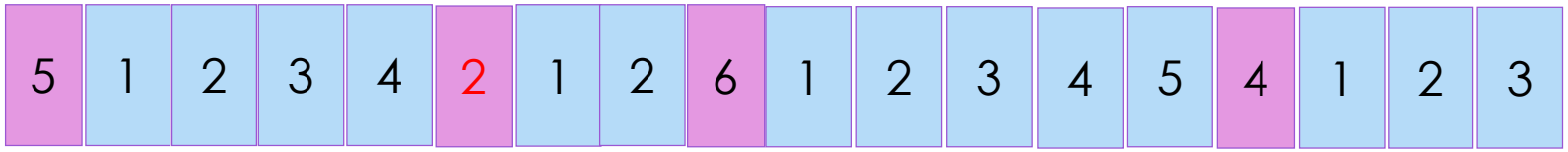
- Could compress this down to just

Given framelength, don't need trailer

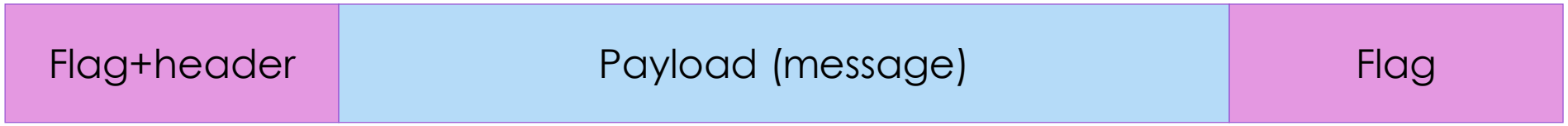
- You just need to be/stay in synch



- Easy to make a mistake



Better Frame



- A frame starts and ends with a '**Flag**'
  - The frame length is what's between (including) some '**flag**' bytes
    - Regain sync by waiting for next frame to start
  - Slight wrinkle: 'flag' byte inside the payload
    - Put an 'escape' byte before any 'flag' bytes inside the payload
  - New wrinkle: 'escape' byte or 'escape/flag' byte pair in the payload
    - Need to 'escape' the 'escape' – and so on.
    - Byte stuffing…

Huh?

- Frame = **FabFcdF** becomes **FabEFcdF**
- Frame = **FabEcdF** becomes **FabEEcdF**
- Frame = **FabEFcdF** becomes **FabEEEFcdF**
- Frame = **FabEEcdF** becomes **FabEEEEcdF**

- *Look familiar?* “”” “”\x”””
- Receiver just drops the escape byte
- Any unescaped flag byte is a real flag!

MAC and sharing

- Media Access Control
- Needs an **address scheme**
  - 'MAC address' – hardwired (sort of) to your network interface
  - Identity of the *interface*, not the computer
  - Listen to (receive) all traffic, respond to stuff sent to you
- Needs an **access scheme** for multiple devices ("multiple access"…)
  - No one is in charge. Think 'party atmosphere'

Or Channel access

- Two common models
  - Randomised access – try your luck
  - Contention-free access – stricter rules

Randomised access

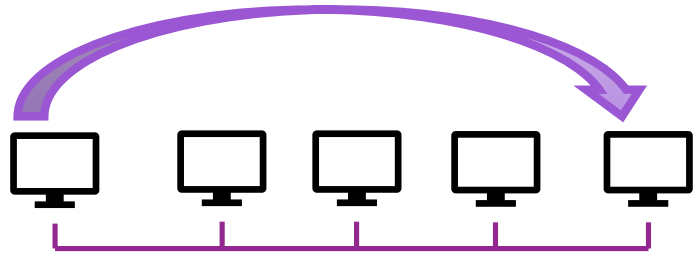
- ALOHA (1960s!)
  1. If you have data to send, send it
  2. If the other end doesn't ACKnowledge it,
     or you hear another device transmitting while sending:

     We've had a COLLISION!
  3. If Collision, wait a random time (back-off), and try again

- Very simple. Very effective. If the load isn't too high
  - Statistical performance up to 18%-36%.
  - Performance depends on the back-off scheme, and better designs exist

CSMA

- Carrier-Sense Multiple Access
  - Good for wired (copper) networks, needs more work with wireless
- Like ALOHA, just check if somebody is sending first *(sense for carrier)*
  - As soon as line is clear, send the whole frame
- Much better – no collisions?
  - Delay on long cables. Two senders can start at the same time without realising
  - As *bandwidth*delay* gets bigger, problem gets bigger
  - Sets upper limits on delay, and minimum frame size
    - Need enough time to detect a collision,
    - Whole frame could be out, you start next one…
    - Minimum frame time is 2*delay



CSMA/C*

- Collision Avoidance (CSMA/CA)
  - Listen for carrier
  - Once it's clear, wait a **random** time
    - *Avoid all the waiting terminals starting at the same time*
  - Then send the whole frame

- Collision Detection (CSMA/CD)
  - Listen for carrier
  - Send, and listen for a collision while sending.
  - If yes, stop sending immediately, and "jam"… ('hey, everyone back off')
  - Then back-off before retrying

Backing off…

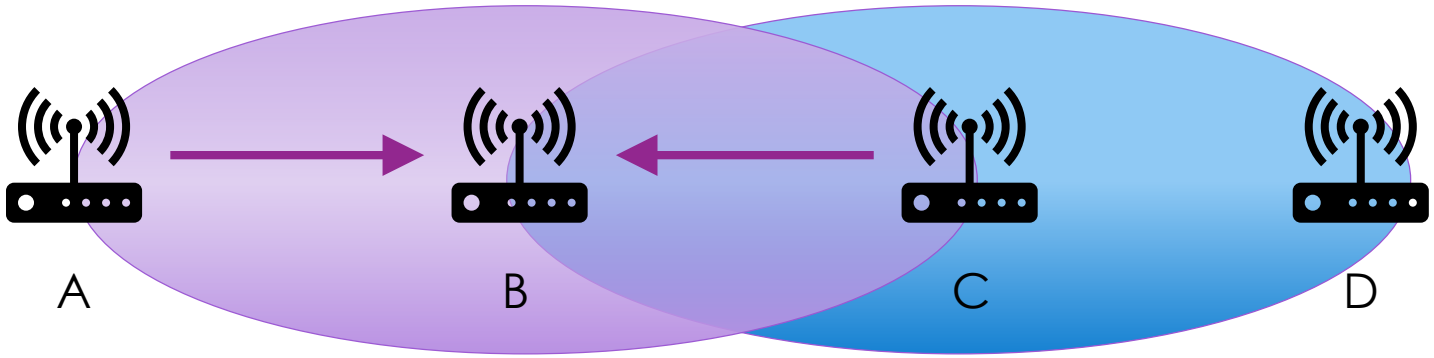
- Need some limits on how long to back-off
  - Can't be too short
  - Can't be too long
- Ideally back-off depending on the number of terminals on the LAN
  - Send with 1/N probability. How do you know N?
- Binary Exponential Back-off (BEB)
  - Counting the number of collisions you experience
  - First time, wait 0-1 frames.
  - Second time wait 0-3 frames
  - Third time wait 0-7 frames.

Wireless is harder

- Each node has ‘coverage’
  - Can be part of a single network but can’t see all the nodes (power limitation)
  - i.e. may be no single carrier to sense

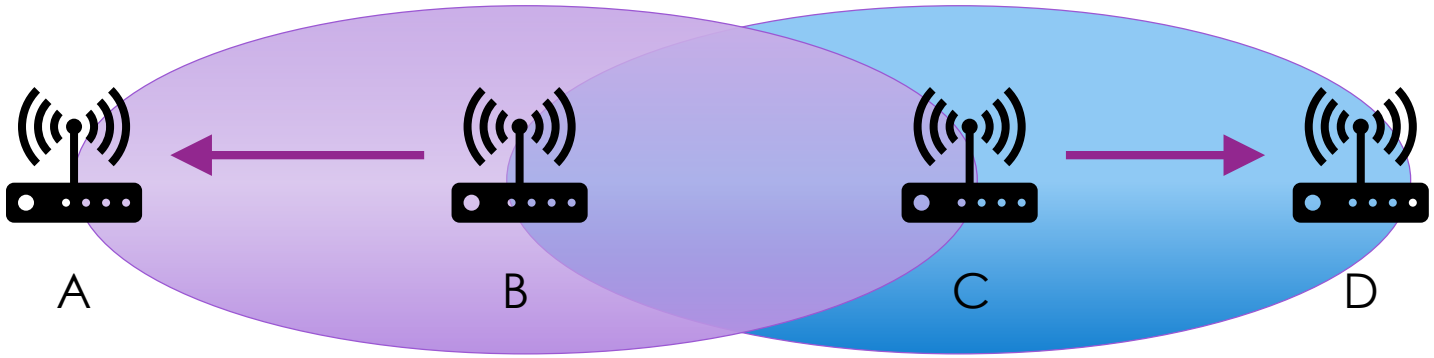
- In wireless, TX can be a million times louder than RX
  - i.e. can’t listen for collisions
  - Can’t respond quickly to collisions, lots of wasted time

Hidden terminals (stations)



A and C are “hidden terminals” from each other:
Both can talk to B – and collide

Exposed terminals

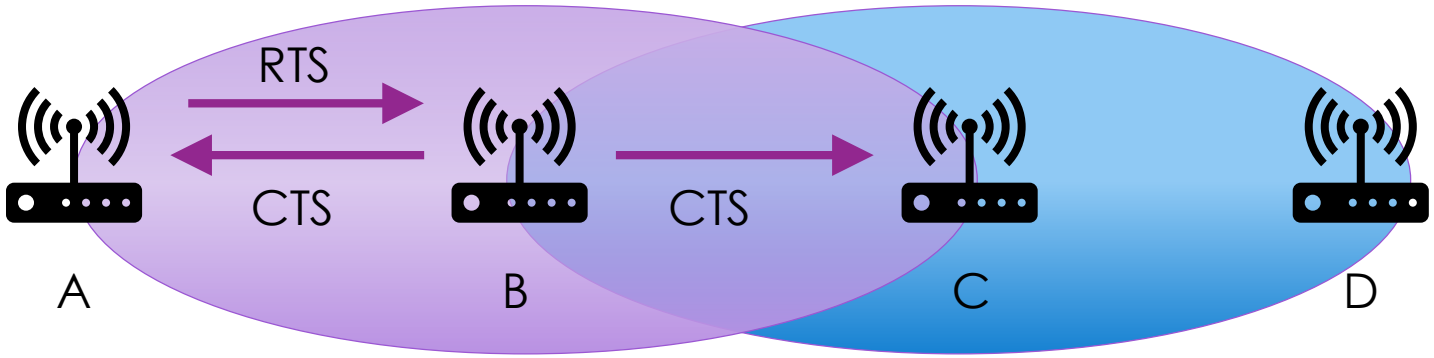


B and C are “exposed terminals” to each other:
Both could talk to an outer neighbour – and NOT interfere
Want to avoid wasting bandwidth

MACA

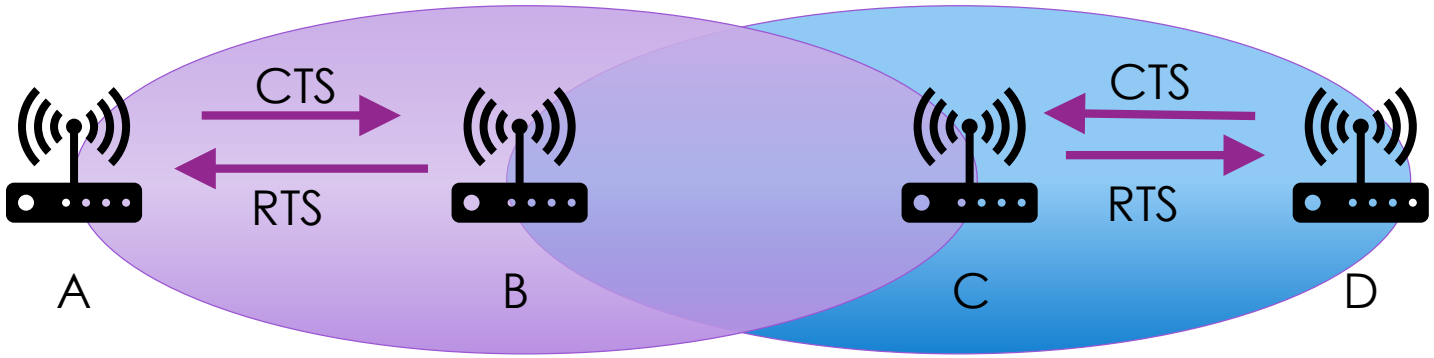
- Multiple Access, Collision Avoidance
- A quick handshake before yelling:
- Sender: Request to Send (RTS) + N bytes
- Receiver: Clear to Send (CTS) + N bytes
- Sender transmits
  - *and any nodes that heard the CTS stay silent for N bytes*
  - *and any nodes that heard the RTS stay silent for the CTS*

Fixing hidden terminal problem



Node C knows something is going on, and waits N bytes

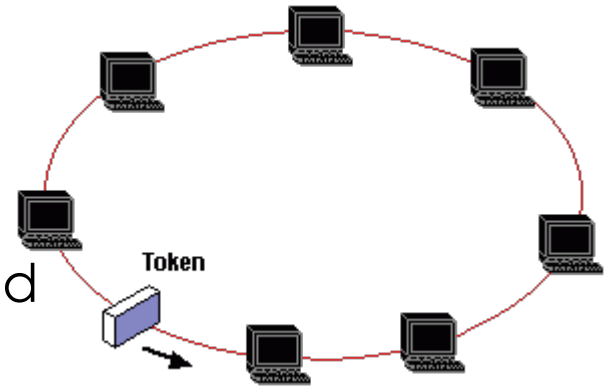
Fixing exposed terminal problem



B can't hear D/C-CTS, C can't hear A/B-CTS – All Clear!

Contention-free access

- Take turns!
- Identify an order of the devices on a network
- E.g. Token Ring: uses a special frame, passed around
  - You can't send if you haven't got the token
  - Which can make it fragile to errors, or engineer around it



Broadcasting

- LAN broadcasts
- I have an announcement
  - That all should know
  - That somebody should know, but I don't know who
- I'm asking a question
- Implemented through a special 'destination' address **(all-1's)**

Topologies

- Most wired LANs have moved away from bus topologies
  - Eventually doesn't scale
  - Make a bus (cable) longer:
    - Needs a repeater or hub (regen)
    - Or a bridge, which links two LANs, and learns addresses on each side

- Today nearly all are 'switched'
  - Crossbar devices that learn source/destination addresses from the traffic
  - Makes every link point-to-point (computer to switch)
  - Greater scalability
  - Greater performance