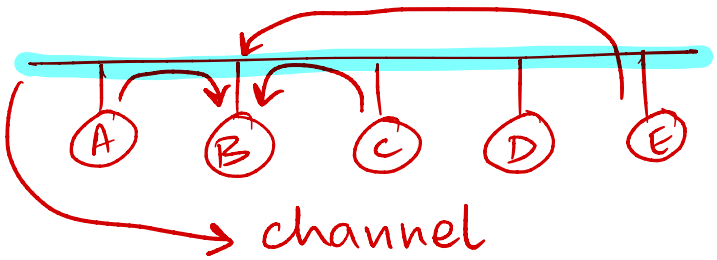
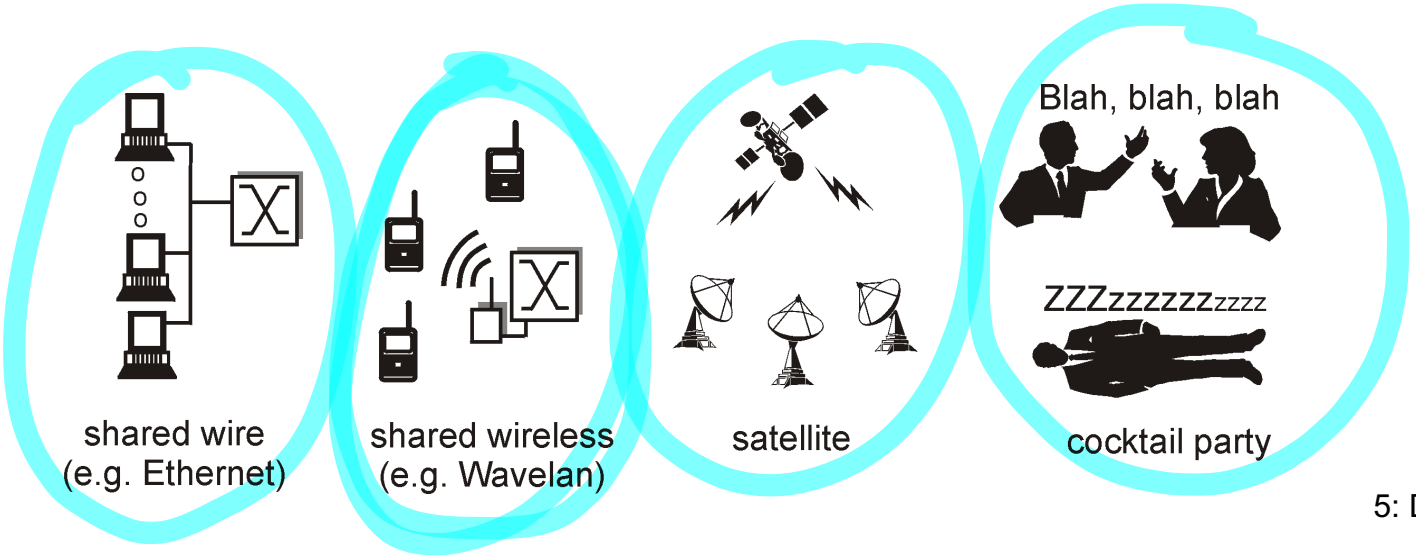
# Multiple Access Links and Protocols

Two types of "links":




- point-to-point
  - PPP for dial-up access
  - point-to-point link between Ethernet switch and host
- broadcast (shared wire or medium)
  - traditional Ethernet
  - Bluetooth
  - 802.11 wireless LAN



# Multiple Access protocols

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
  - collision if node receives two or more signals at the same time

*multiple access protocol*

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
  - no out-of-band channel for coordination

# Ideal Multiple Access Protocol

Broadcast channel of rate R bps

1. When one node wants to transmit, it can send at rate R.
2. When M nodes want to transmit, each can send at average rate R/M
3. Fully decentralized:
   - no special node to coordinate transmissions
   - no synchronization of clocks, slots
4. Simple

# MAC Protocols: a taxonomy

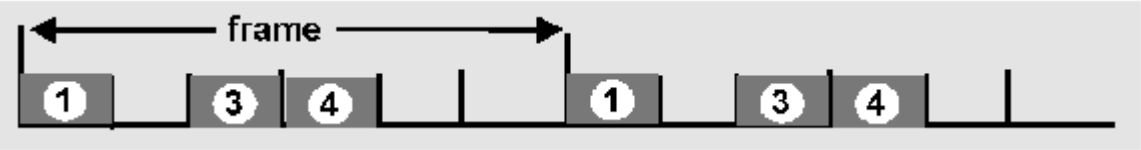
Three broad classes:

- Channel Partitioning
  - divide channel into smaller "pieces" (time slots, frequency, code)
  - allocate piece to node for exclusive use
- Random Access
  - channel not divided, allow collisions
  - "recover" from collisions
- "Taking turns"
  - Nodes take turns, but nodes with more to send can take longer turns

# Channel Partitioning MAC protocols: TDMA

## TDMA: time division multiple access

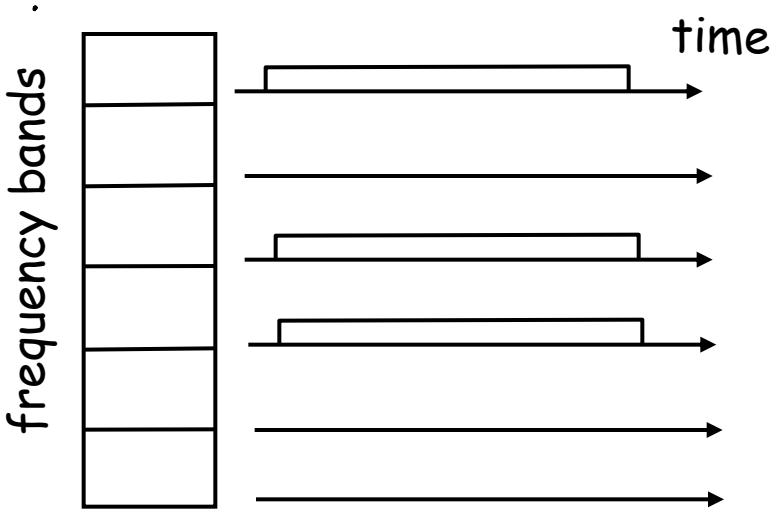
- access to channel in "rounds"
- each station gets fixed length slot (length = pkt trans time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle



# Channel Partitioning MAC protocols: FDMA

## FDMA: frequency division multiple access

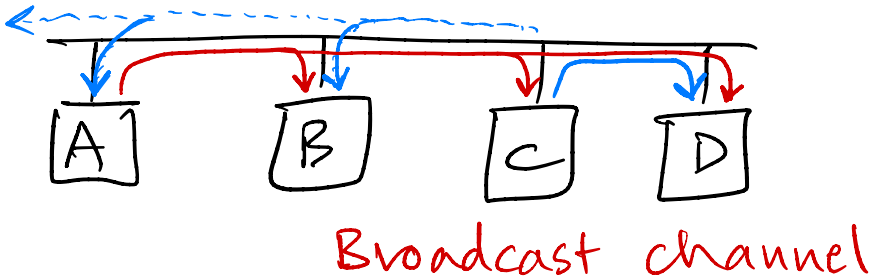
- channel spectrum divided into frequency bands
- each station assigned fixed frequency band
- unused transmission time in frequency bands go idle
- example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



# Random Access Protocols

- When node has packet to send
  - transmit at full channel data rate R.
  - no *a priori* coordination among nodes
- two or more transmitting nodes ➜ "collision",
- random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)
- Examples of random access MAC protocols:
  - slotted ALOHA
  - ALOHA
  - CSMA, CSMA/CD, CSMA/CA

# Slotted ALOHA




## Assumptions

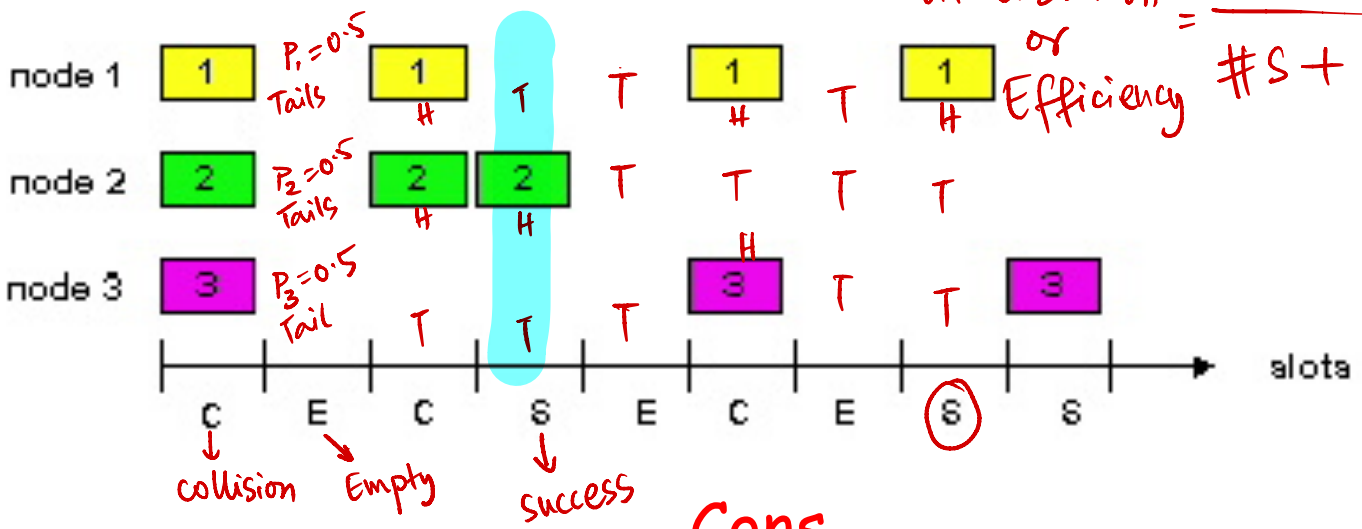
- all frames same size
- time is divided into equal size slots, time to transmit 1 frame
- nodes start to transmit frames only at beginning of slots
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

## Operation

- when node obtains fresh frame, it transmits in next slot
- no collision, node can send new frame in next slot
- if collision, node retransmits frame in each subsequent slot with prob. p until success

# Slotted ALOHA

Utilization or Efficiency $= \dfrac{\#S}{\#S + \#C + \#E}$




## Pros

- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

## Cons

- collisions, wasting slots
- idle slots
- nodes must be able to detect collision in less than time to transmit packet
- clock synchronization

# Slotted Aloha efficiency

**Efficiency** is the long-run fraction of successful slots when there are many nodes, each with many frames to send

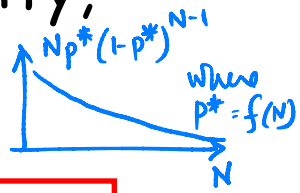
- Suppose N nodes with many frames to send, each transmits in slot with probability *p*
- prob that node 1 has success in a slot $= p(1-p)^{N-1}$
- prob that any node has a success $= Np(1-p)^{N-1}$

$\left(N \text{ choose } 1\right) \cdot p \cdot (1-p)^{N-1}$

- For max efficiency with N nodes, find p* that maximizes $Np(1-p)^{N-1} = f(p)$

$\frac{df(p)}{dp} = 0$ for a given N.

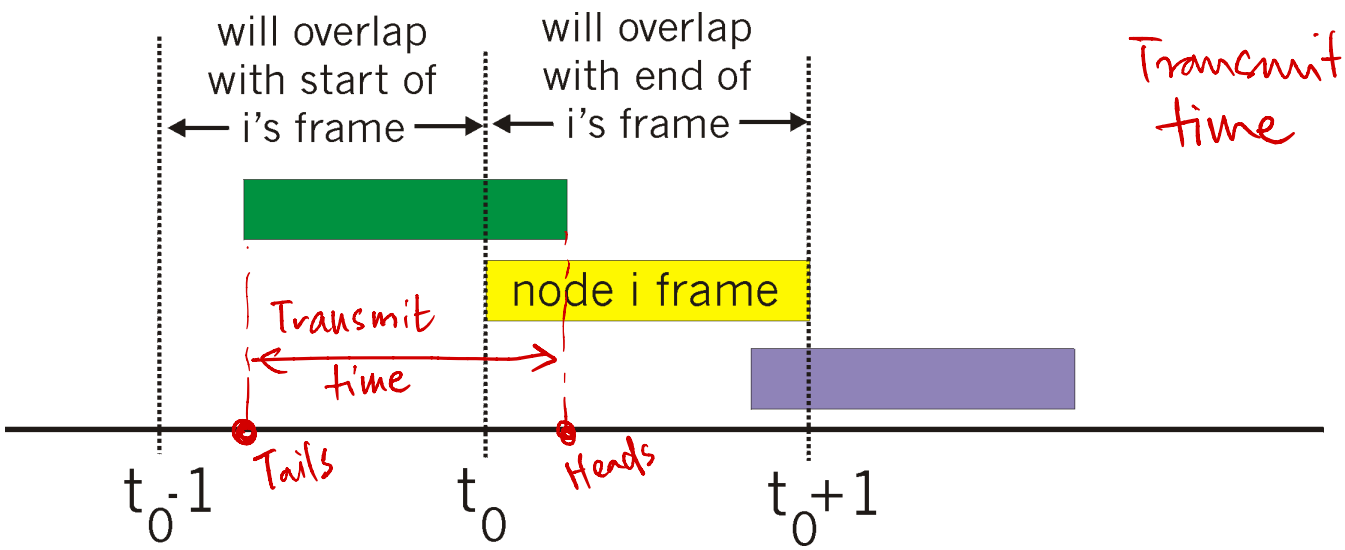
- For many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives 1/e = .37




*At best:* channel used for useful transmissions 37% of time!

# Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- when frame first arrives
  - transmit immediately
- collision probability increases:
  - frame sent at $t_0$ collides with other frames sent in $[t_0-1, t_0+1]$




$$\text{Transmit time} = \frac{\text{Pkt size}}{R}$$

# Pure Aloha efficiency

P(success by given node) = P(node transmits) ·

P(no other node transmits in $[t_0-1,t_0]$ ·

P(no other node transmits in $[t_0,t_0+1]$

$= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1}$

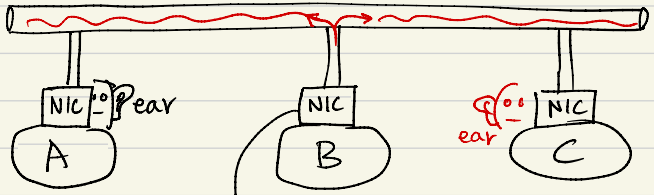
$= p \cdot (1-p)^{2(N-1)}$

… choosing optimum p and then letting n -> infty …

Even worse ! $= 1/(2e) = .18$ 18%

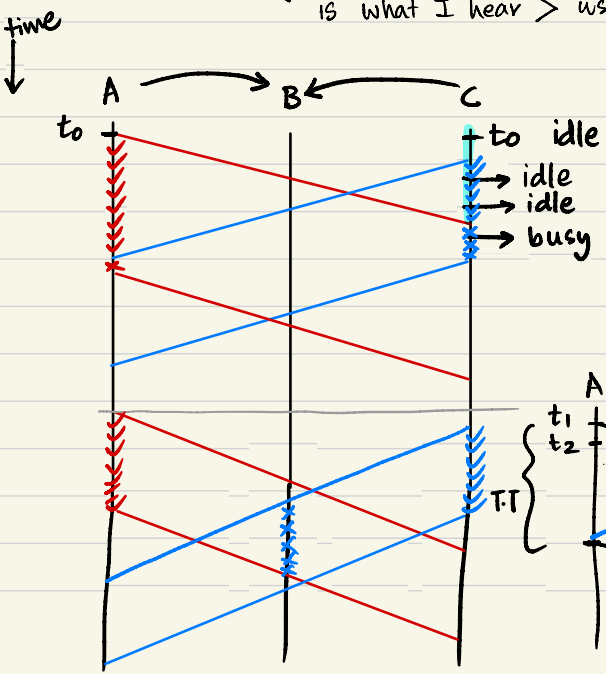
$\lim_{N \to \infty} N p^* (1-p^*)^{2(N-1)}$

# Carrier Sense Multiple Access (CSMA)



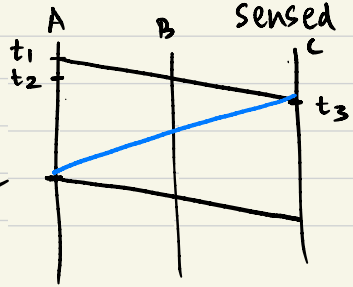

Listen before you talk.

is what I hear > usual noise in my hardware.



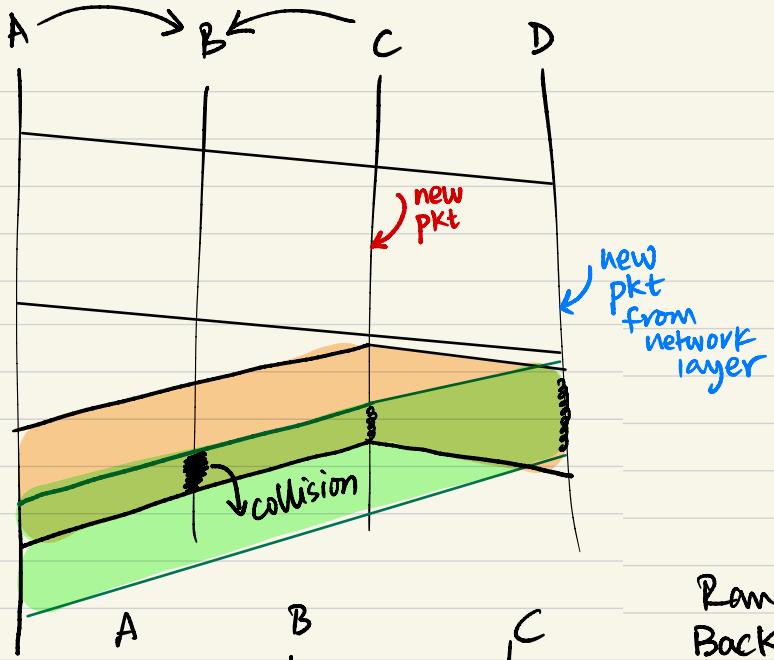

Both A and C need to run the following check constantly during pkt transmission:

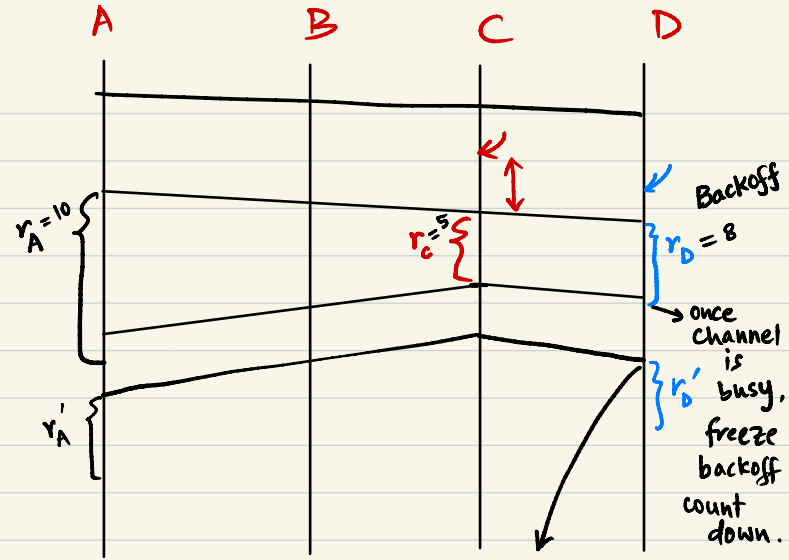
is my transmitted signal == my carrier sensed signal ?

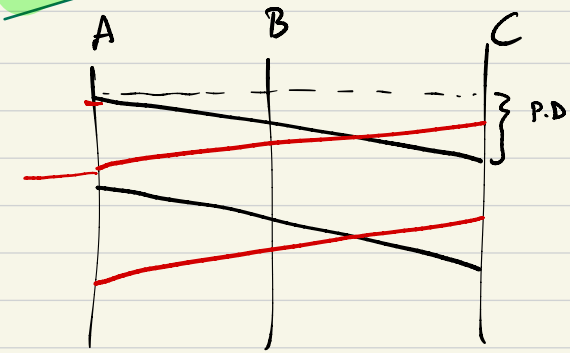



Condition to detect collision:

Transmit time > 2 . max Prop. Delay




Random Backoff is chosen from a range

$[0, CW]$

↳ contention window $\approx 15$

once channel idle again, resume countdown from the frozen value.

if collision $\Rightarrow$ New CW $= 2 \times$ old CW.

if successful $\Rightarrow$ new CW $= 15$ again.

# Ethernet IEEE 802.3

Starting CW $= CW_{min} = 15 \Rightarrow$ Pick B/O $\sim [0, CW_{min}]$

$K^{th}$ Collision $\Rightarrow 2^K CW \Rightarrow$ 1st collision $\sim [0, 31]$
2nd collision $\sim [0, 63]$

↳ Exponential Backoff.

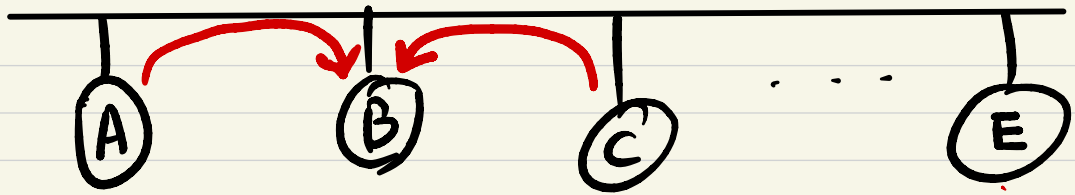
Upon success $\Rightarrow CW = CW_{min} = 15$

If failure after $K=3$ retransmit $\longrightarrow$ Drop packet.

CSMA/CD $\rightarrow$ Collision Detection.

↳ The Tx detects the collision.

## After class




collision detection

SNR is inadequate, hence collision

collision detection

Does not know that collision happened.

If E wants, E can sample the signal, get bits, & then check error detection code.

# CSMA (Carrier Sense Multiple Access)

**CSMA:** listen before transmit:

If channel sensed idle: transmit entire frame

- If channel sensed busy, defer transmission

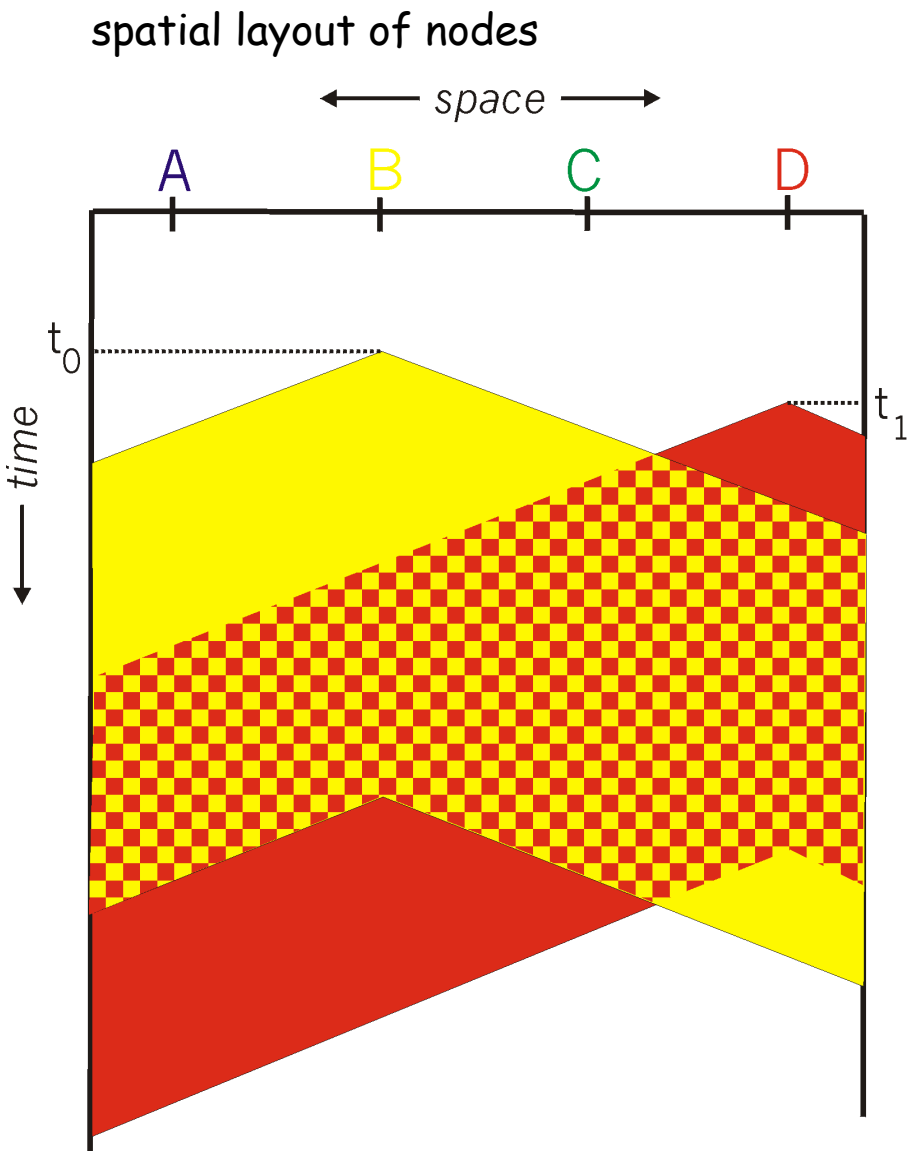
- Human analogy: don't interrupt others!

# CSMA collisions

**collisions *can* still occur:**
propagation delay means two nodes may not hear each other's transmission

**collision:**
entire packet transmission time wasted

**note:**
role of distance & propagation delay in determining collision probability



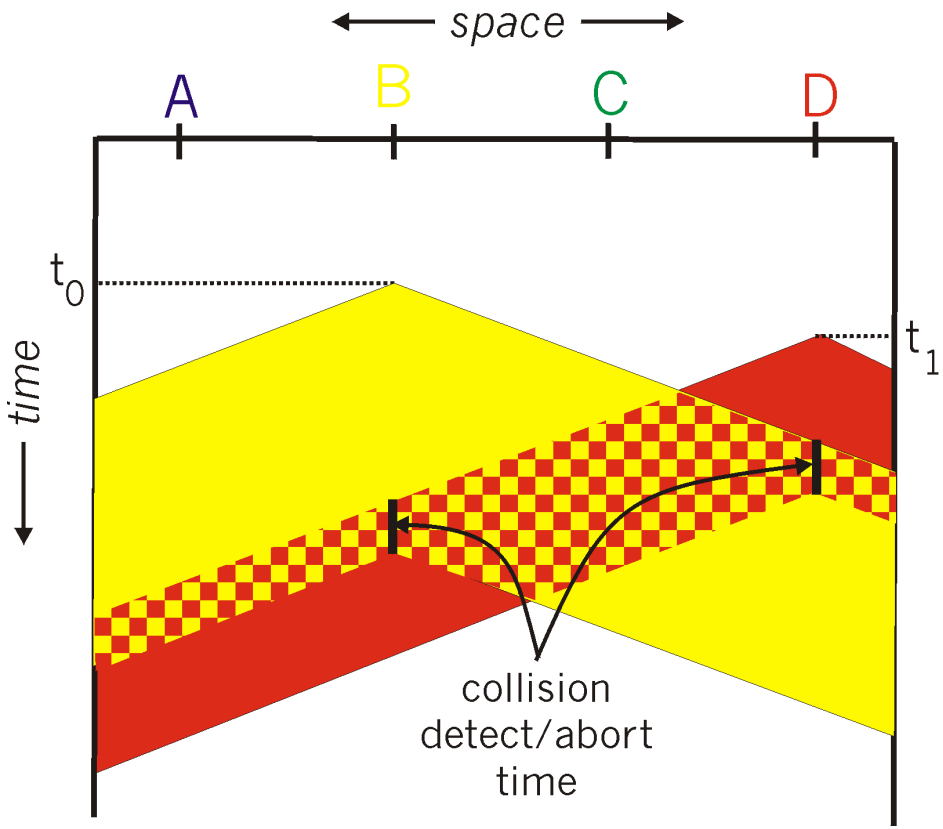
# CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, deferral as in CSMA

- collisions *detected* within short time
- colliding transmissions aborted, reducing channel wastage

- collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: receiver shut off while transmitting
- human analogy: the polite conversationalist

# CSMA/CD collision detection



# "Taking Turns" MAC protocols

channel partitioning MAC protocols:

- share channel efficiently and fairly at high load
- inefficient at low load: delay in channel access, 1/N bandwidth allocated even if only 1 active node!

Random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

"taking turns" protocols

look for best of both worlds!

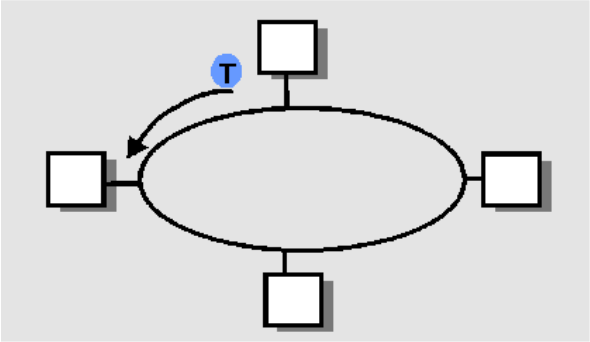
# "Taking Turns" MAC protocols

Polling:

- master node "invites" slave nodes to transmit in turn
- concerns:
  - polling overhead
  - latency
  - single point of failure (master)

Token passing:

- control **token** passed from one node to next sequentially.
- token message
- concerns:
  - token overhead
  - latency
  - single point of failure (token)



# Summary of MAC protocols

- What do you do with a shared media?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Frequency Division
  - Random partitioning (dynamic),
    - ALOHA, S-ALOHA, CSMA, CSMA/CD
    - carrier sensing: easy in some technologies (wire), hard in others (wireless)
    - CSMA/CD used in Ethernet
    - CSMA/CA used in 802.11
  - Taking Turns
    - polling from a central site, token passing

# Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM