Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs
- addressing, ARP
- Ethernet
- switches
- VLANS

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web request

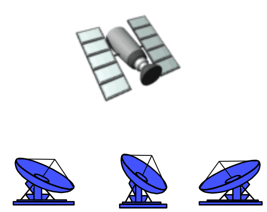
Multiple access links, protocols

two types of "links":

- point-to-point
  - PPP for dial-up access
  - point-to-point link between Ethernet switch, host
- *broadcast (shared wire or medium)*
  - old-fashioned Ethernet
  - upstream HFC
  - 802.11 wireless LAN

shared wire (e.g., cabled Ethernet)

shared RF (e.g., 802.11 WiFi)



shared RF (satellite)

humans at a cocktail party (shared air, acoustical)

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

analogy to human protocols

An ideal multiple access protocol (MAC)

given: broadcast channel of rate R bps

desired features of the protocol:

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

Ideal performance:

1. No idle time (when there is traffic waiting)
2. No wasted time (collisions)
3. No access delay

MAC protocols: taxonomy

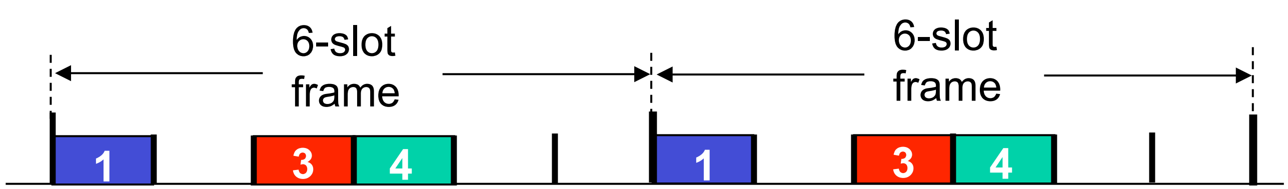
three broad classes:

- *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

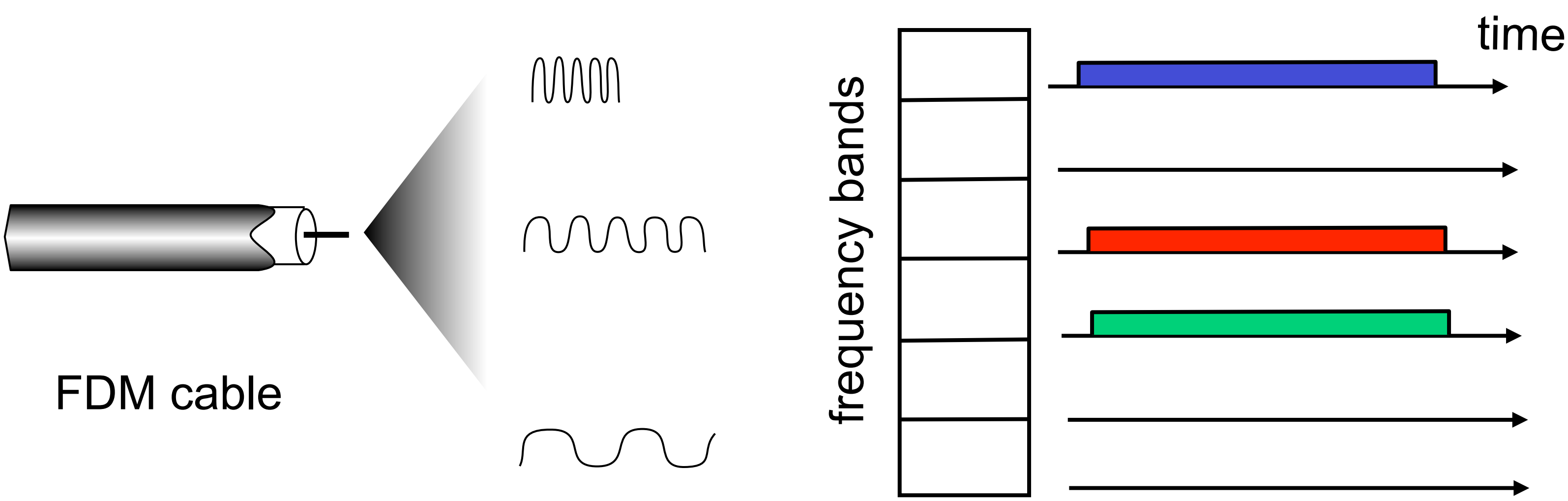
- access to channel in "rounds"
- each station gets fixed length slot (length = pkt trans time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle



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



Channel Partitioning MAC protocols: CDMA

CDMA: code division multiple access

- each station is assigned a different code
- each station uses its code to encode data bits
- stations can transmit simultaneously
- orthogonal codes→ stations can transmit simultaneously

- widely used in wireless

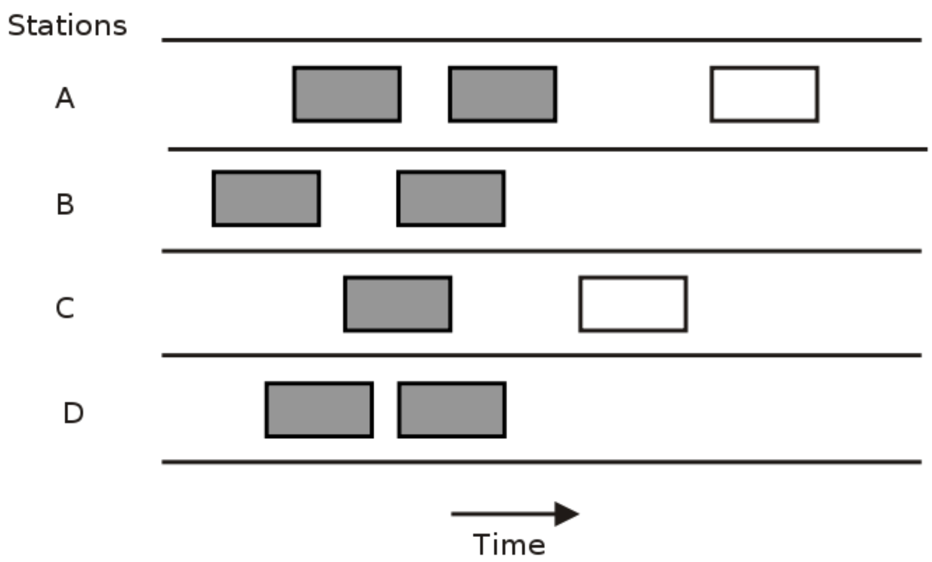
Channel Partitioning

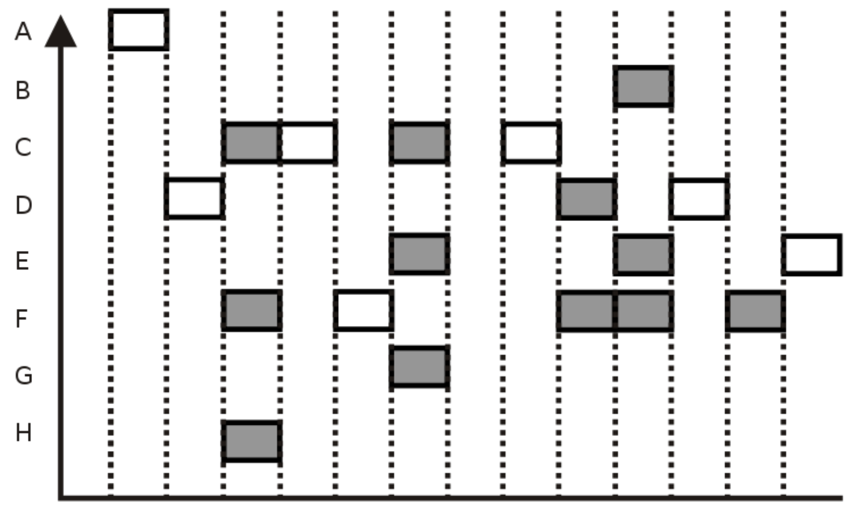
- Pros
  - No collisions
  - Perfectly fair
- Cons
  - Reserved resources (slots, frequencies) can remain idle even if there is traffic waiting
    - Use fraction of the bandwidth
    - Startup delay (for TDMA)
  - Some traffic may get denied access, although there are idle resources
  - A-priori coordination between nodes needed

Random access protocols

- when node has packet to send
  - transmit at full channel data rate R.
  - no coordination among nodes (distributed)
- two or more transmitting nodes ➜ "collision"
- random access MAC protocol specifies:
  - how each node detects collisions
  - how each node recovers from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
  - slotted ALOHA, "pure" ALOHA
  - CSMA, CSMA/CD, CSMA/CA

Aloha





Slotted ALOHA protocol (shaded slots indicate collision)

Slotted ALOHA

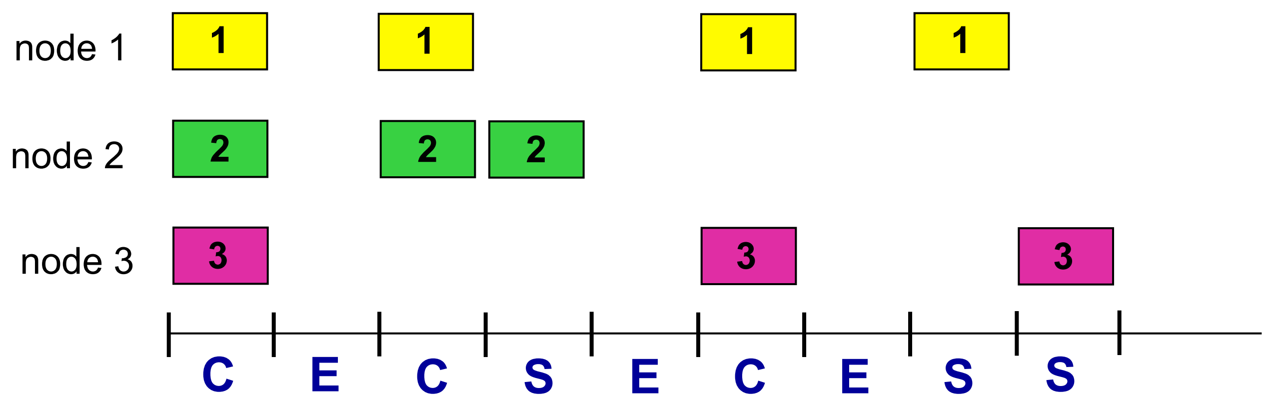
assumptions:

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

operation:

- when node obtains fresh frame, transmits in next slot
  - *if 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



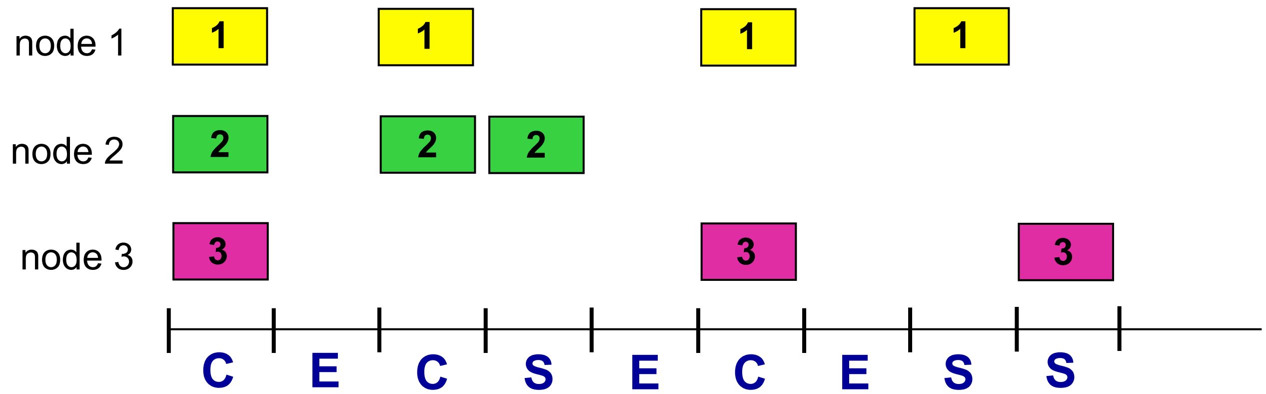
Pros:

- 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 →wasting slots
- clock synchronization
- nodes may be able to detect collision in less than time to transmit packet

Slotted ALOHA: input/output



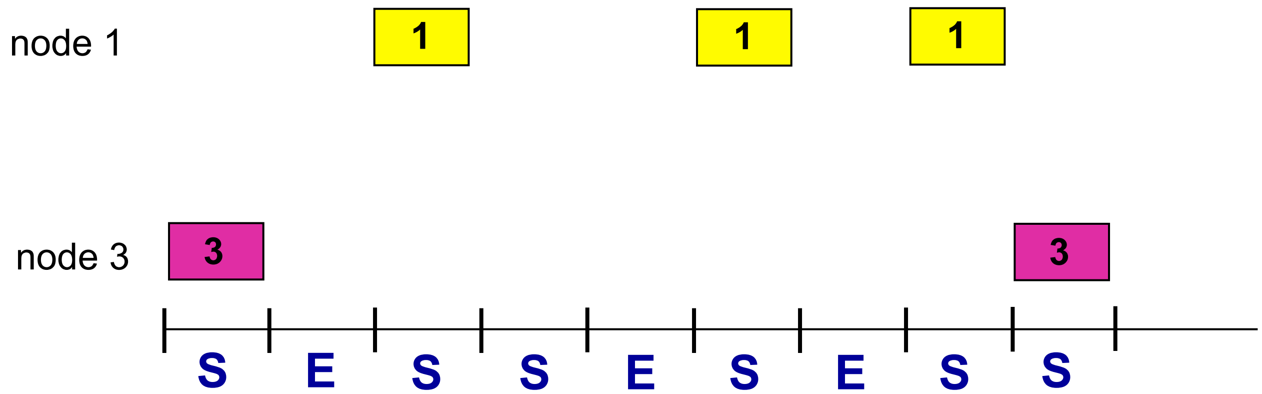
Input:

- *N=1,…:* #nodes with many frames to send
- each node transmits in slot with probability *p in [0,1]*
- *Total offered load: Np*

Output:

- Throughput/efficiency:
  - = long-term fraction of successful slots
  - = prob. for one given slot lead to a successful transmission

Slotted ALOHA: low load regime



Input:

- *Low N:* #nodes with many frames to send

Or

- each node transmits in slot with low prob. p

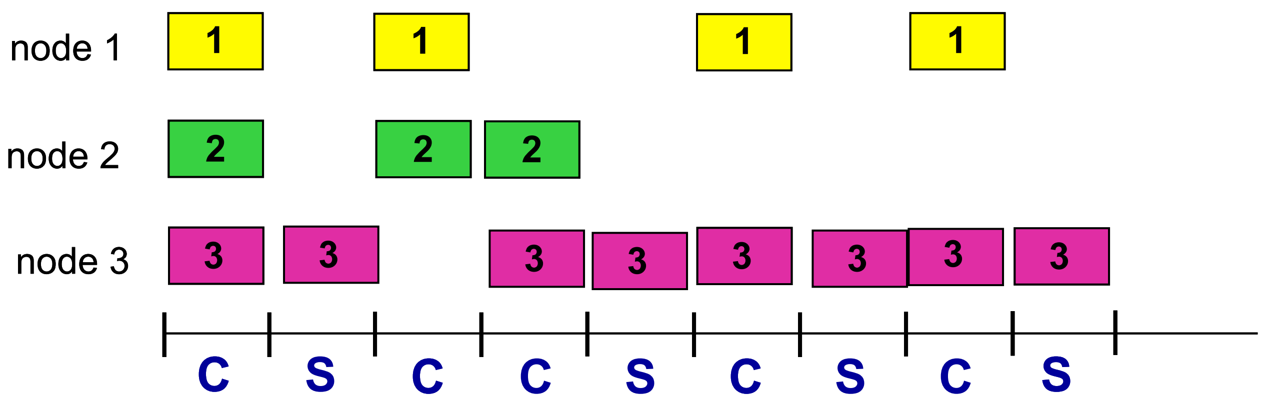
- *Total offered load: Np*
  - is low

Output:

- Throughput = offered load
  - low

  - No collisions
  - Almost all packets get through
  - But may be idle slots

Slotted ALOHA: congestion regime



Input:

- *High N:* # nodes with many frames to send

Or

- each node transmits in slot with high prob. *p*

- *Total offered load: Np*
  - is high

Output:

- low throughput
  - More (re)transmissions than slots
  - Collisions
  - Almost no packets get through

Slotted ALOHA: analysis

efficiency: long-run fraction of successful slots (assume many nodes, all with many frames to send)

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

- max efficiency: $p^*=1/N$ maximizes $Np(1-p)^{N-1}$
  - take derivative dE/dp=0
- for many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives:

  max efficiency = 1/e = .36

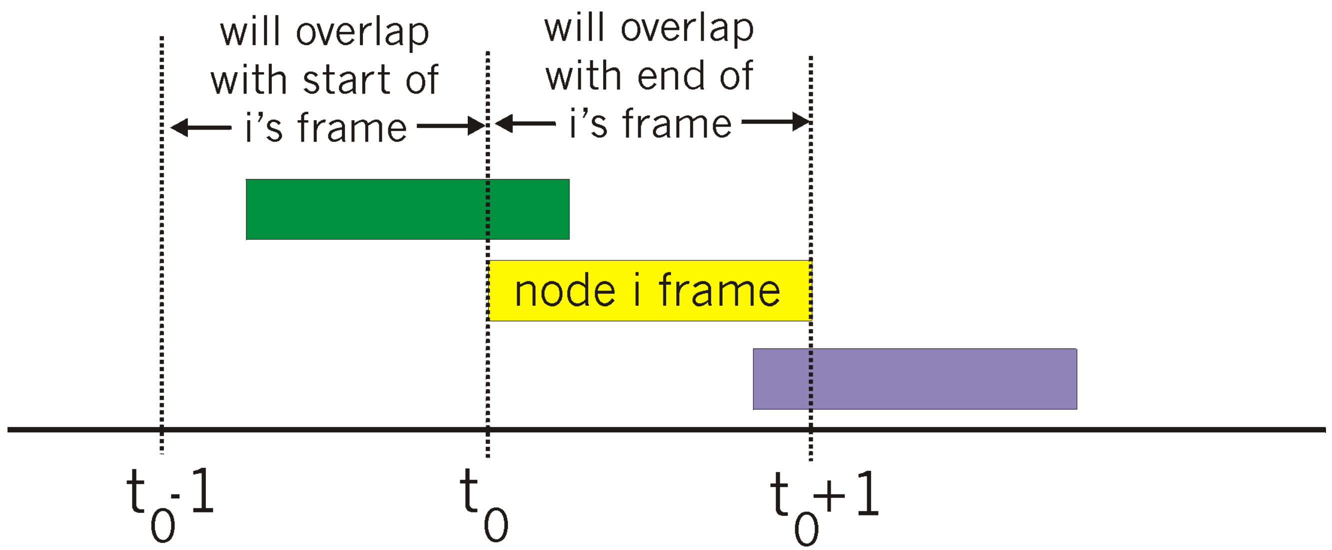
at best: channel used for useful transmissions 36% of time!

!

Another 36% are empty and ~28% go to collisions

Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- when frame first arrives
  - transmit immediately
- If the message collides, try sending "later"
  - Wait a random time ~= transmit in the next time frame w.p. p
- Q: is collision probability higher than in slotted Aloha?
  - A: frame sent at t_0 collides with other frames sent in $[t_0-1, t_0+1]$



Pure ALOHA efficiency

Assume that the prob. of a node transmitting at any given frame time is p.
P(success by given node) = P(node transmits) $\cdot$

P(no other node transmits in $[t_0\text{-}1,t_0]$ $\cdot$

P(no other node transmits in $[t_0\text{-}1,t_0]$

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

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

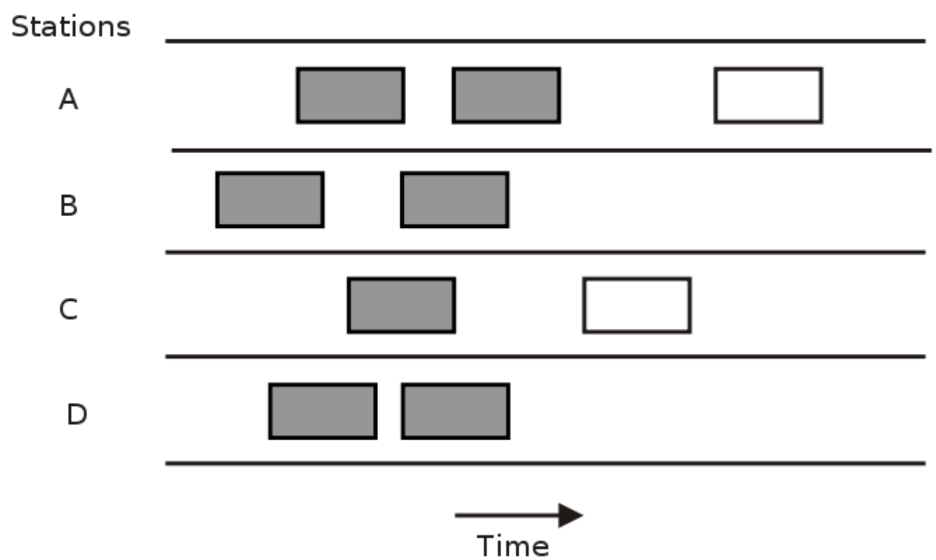
… choosing optimum $p^*=1/(2N\text{-}1)$ and then letting $N \rightarrow \infty$

$$= 1/(2e) = 0.18$$

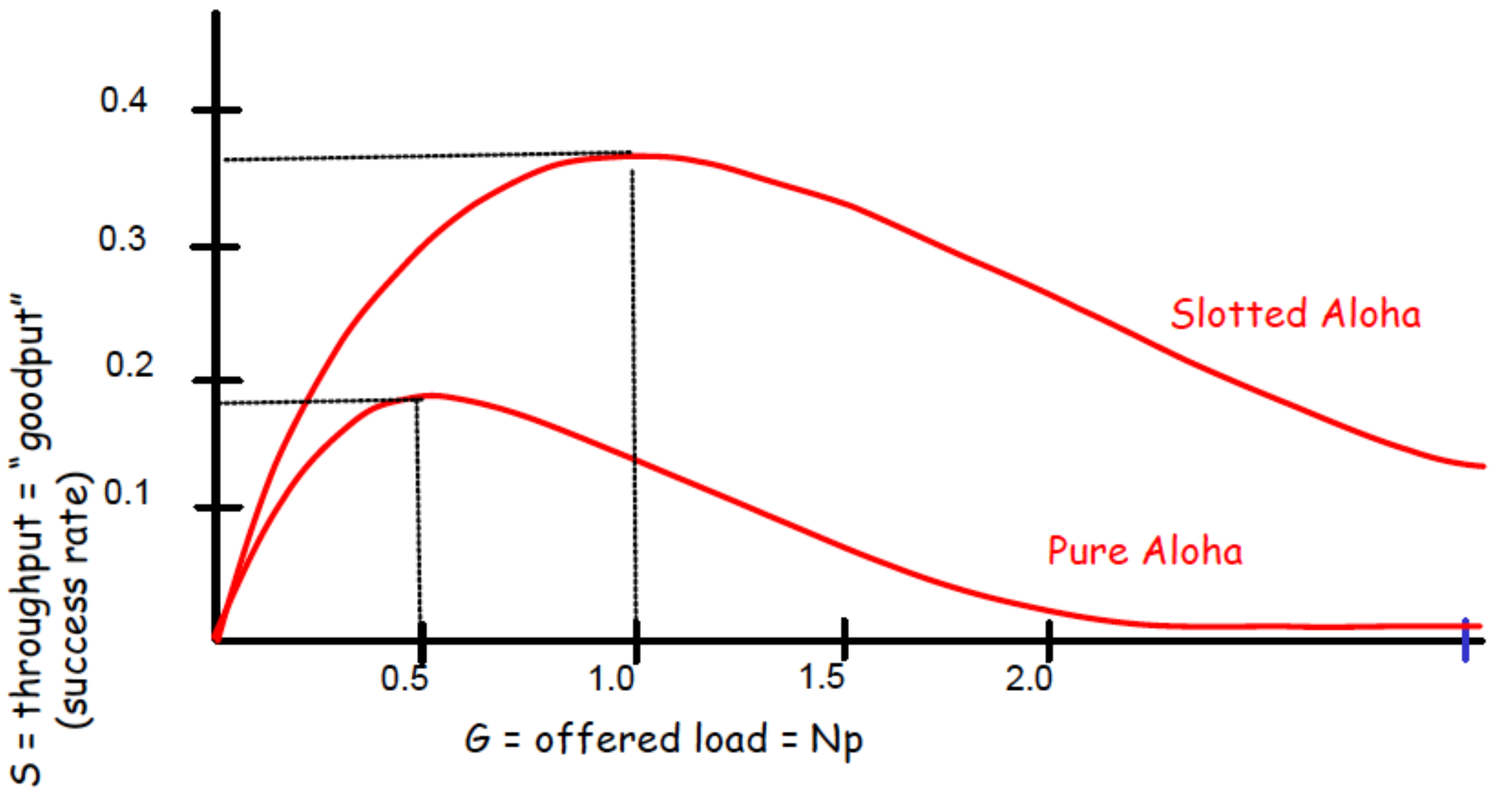
Even *worse* than slotted Aloha! This is the price for being distributed (no clock sync).

A different type of analysis for Aloha

- [Finite and infinite population analysis]
- [http://en.wikipedia.org/wiki/ALOHAnet]



Aloha and Slotted Aloha efficiency



Last time: Random access protocols

- when node has packet to send
  - transmit at full channel data rate R.
  - no coordination among nodes (distributed)
- two or more transmitting nodes ➜ "collision"
- random access MAC protocol specifies:
  - how each node detects collisions
  - how each node recovers from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
  - ALOHA family: slotted ALOHA, "pure" ALOHA
  - CSMA family: CSMA, CSMA/CD, CSMA/CA

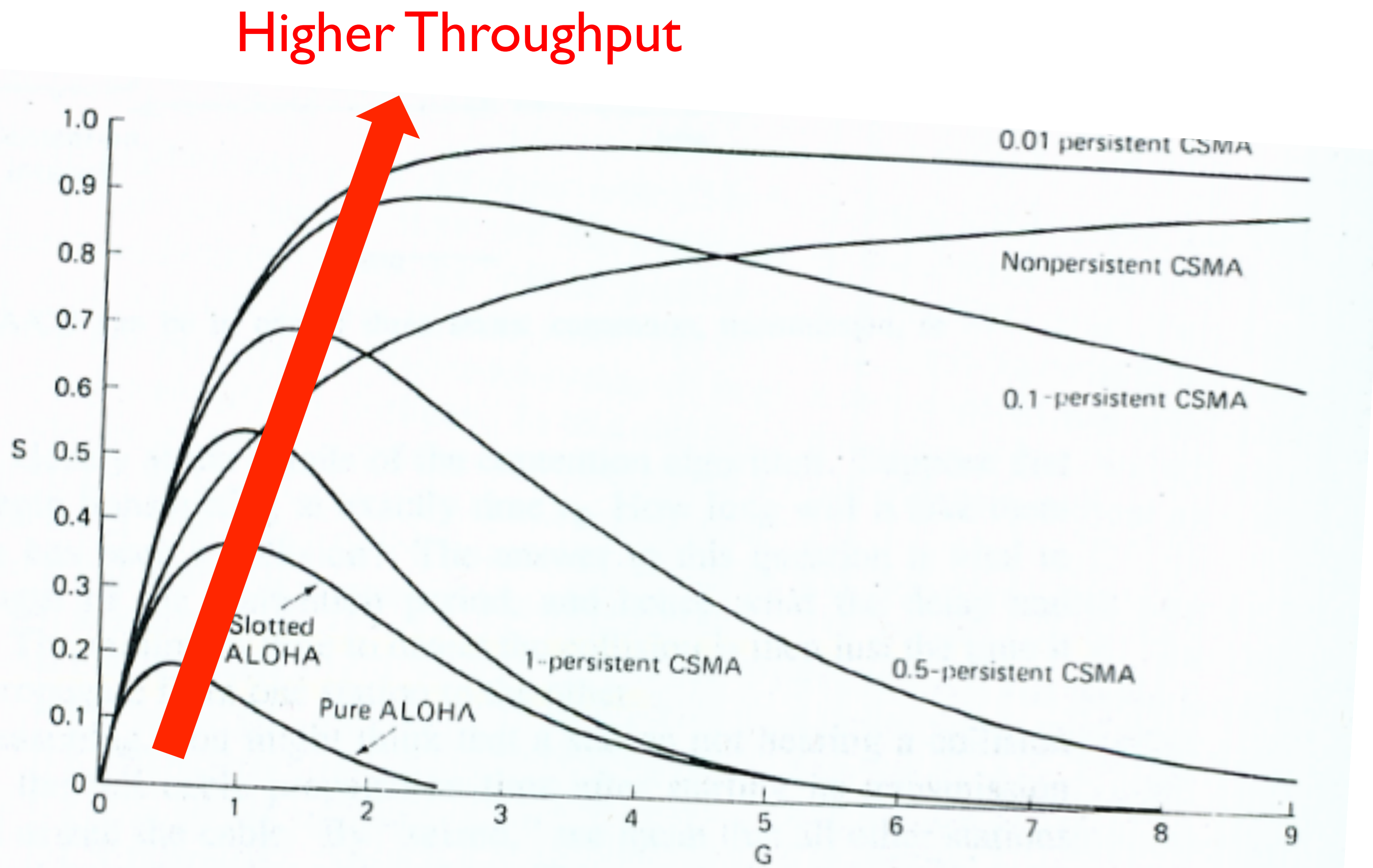


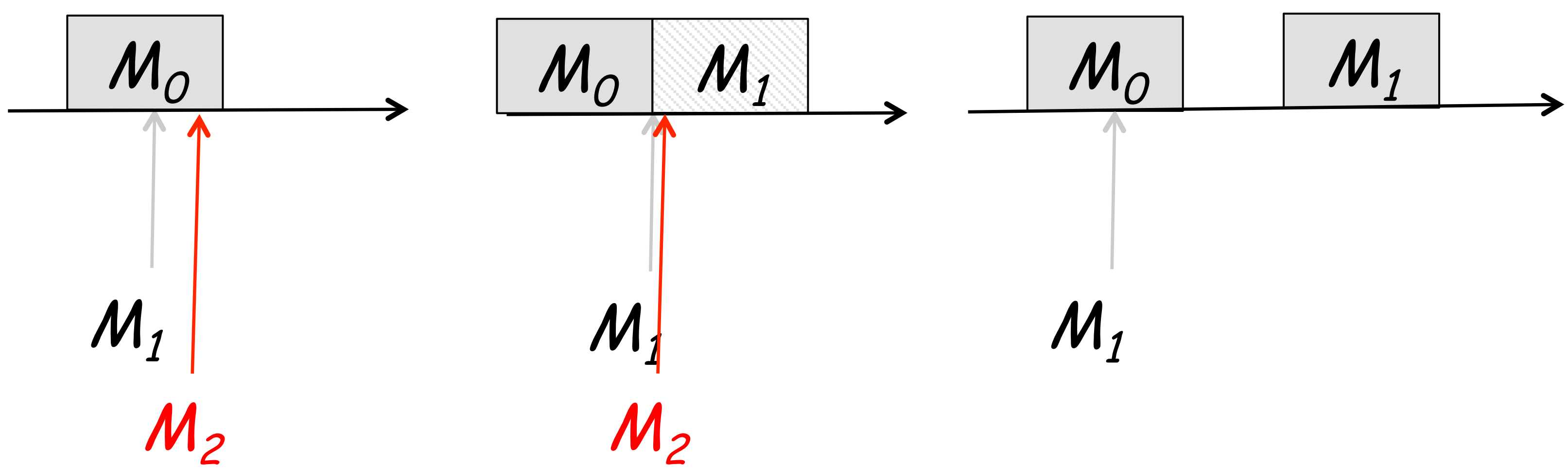
Fig. 3-5. Comparison of the channel utilization versus load for various random access protocols.

CSMA (carrier sense multiple access)

CSMA: listen before transmit:

if channel sensed idle: transmit entire frame

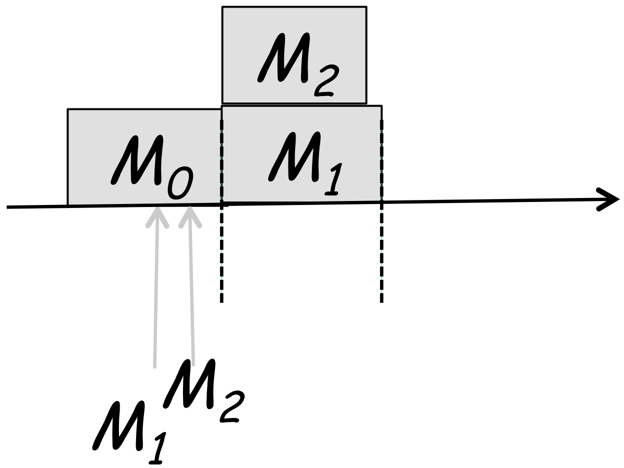
- if channel sensed busy, defer transmission
- human analogy: dont interrupt others!



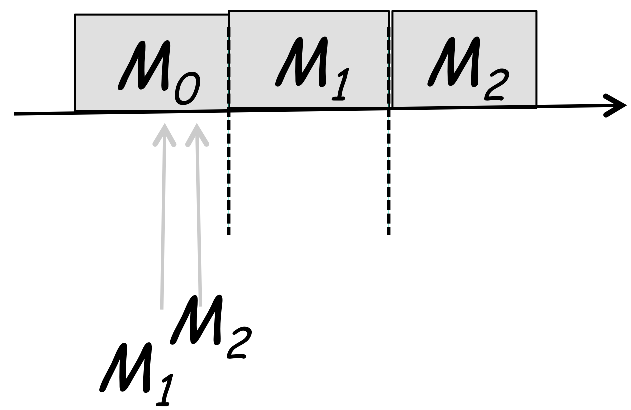
- One would hope that CS eliminate collisions. It doesn't!
  - Because of protocol (other users synchronization)
  - Because of physics (sensing is not instantaneous)

CSMA: Carrier Sensing and Persistence

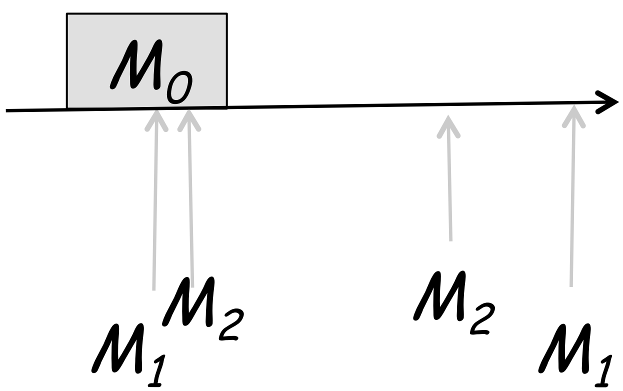
(1-) Persistent: defer transmission until channel becomes idle



P-Persistent: once the channel becomes idle, transmit with prob. p; or defer until next slot



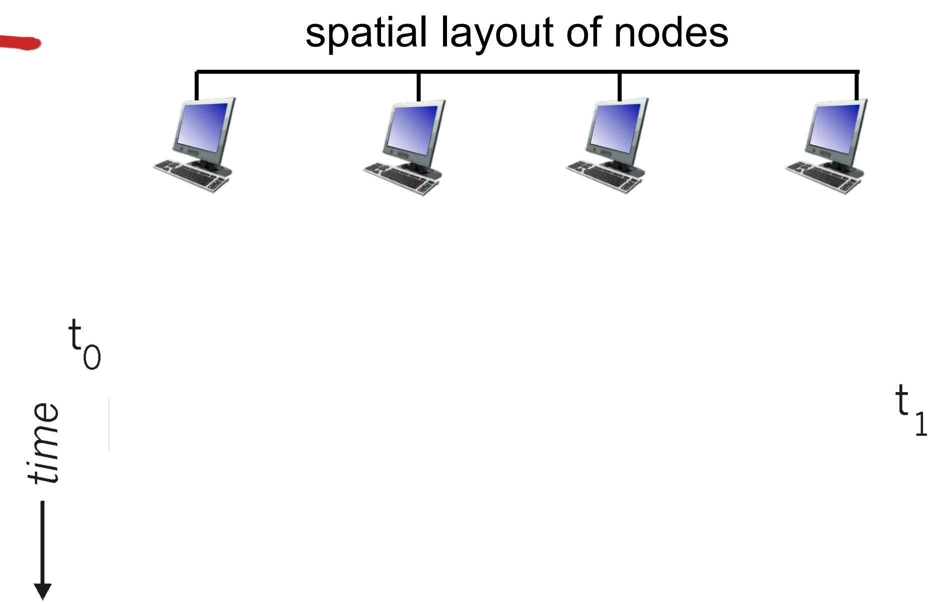
(0-) Non-Persistent: defer transmission, sense again after a random time



Choice of p:
throughput vs delay

CSMA collisions

- Sensing is not instantaneous: propagation delay means two nodes may not hear each other's transmission
- collision: entire packet transmission time wasted
  - distance & propagation delay play role in in determining collision probability



CSMA/CD (with collision detection)

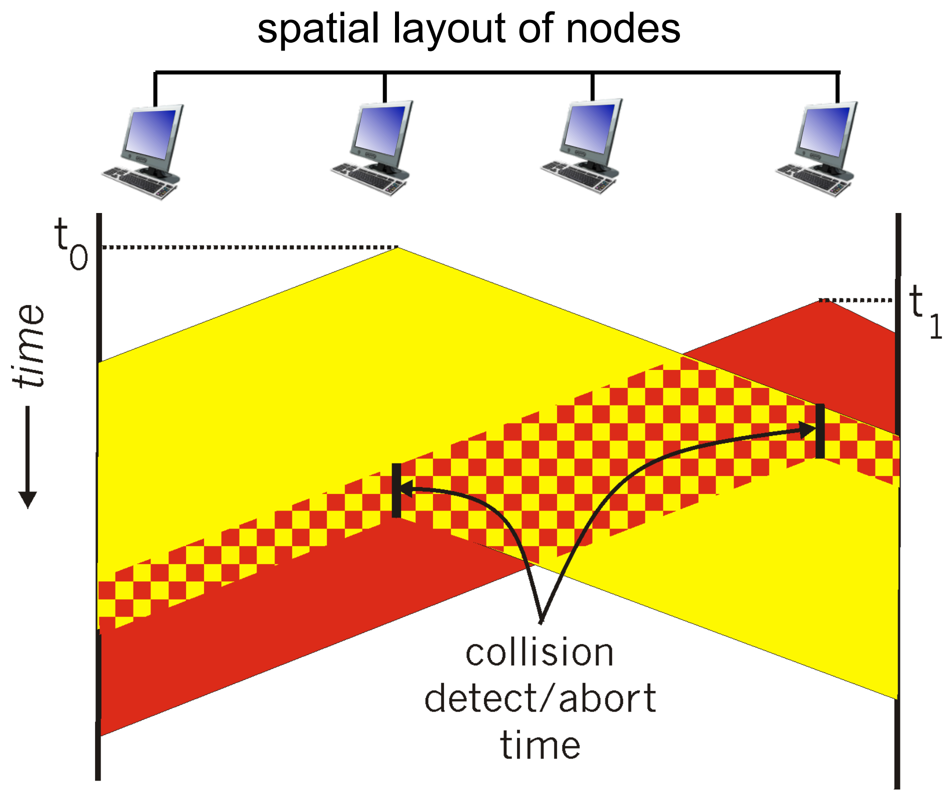
CSMA/CD: carrier sensing, deferral as in CSMA

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

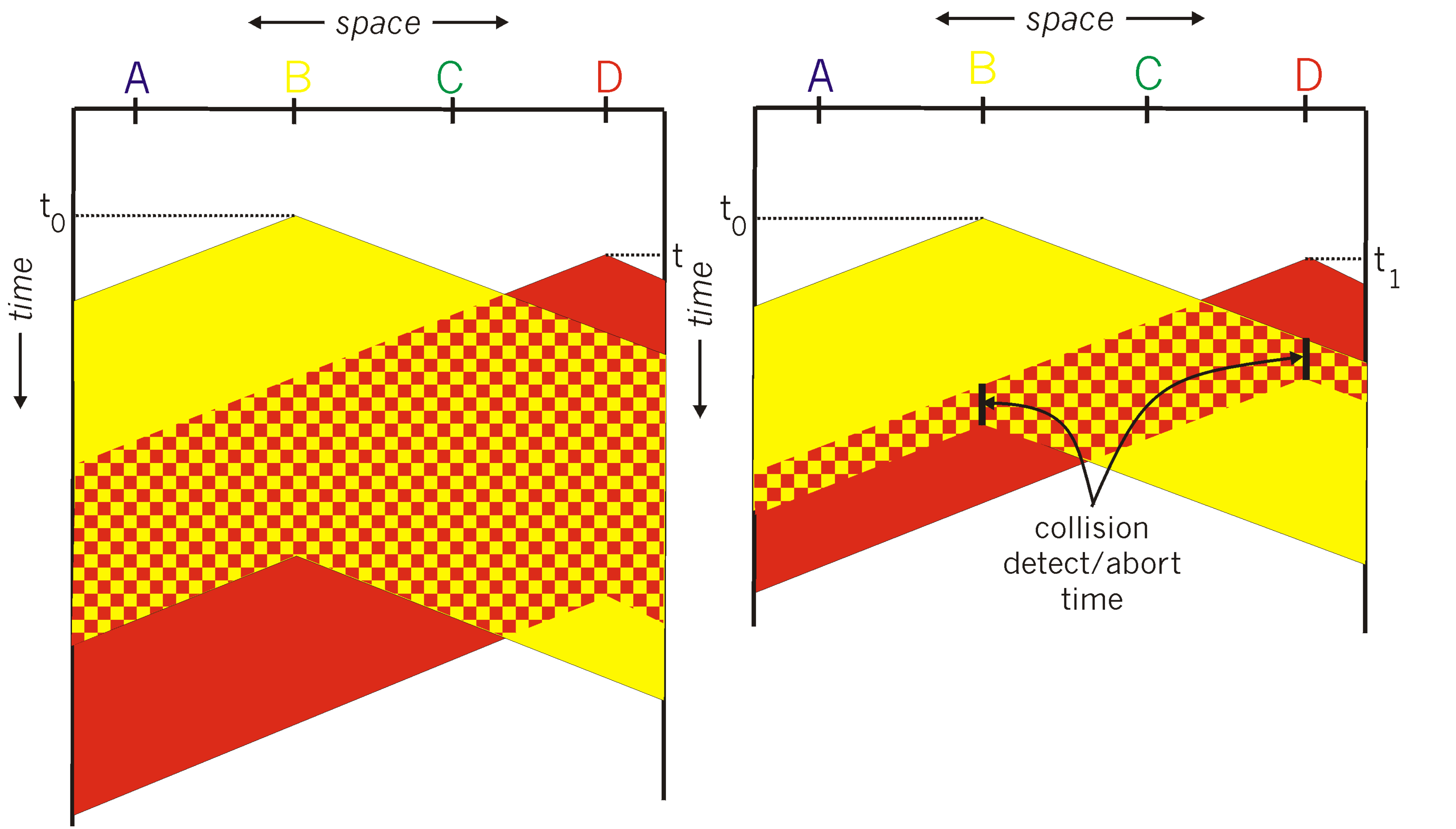
- collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength

- human analogy: the polite conversationalist

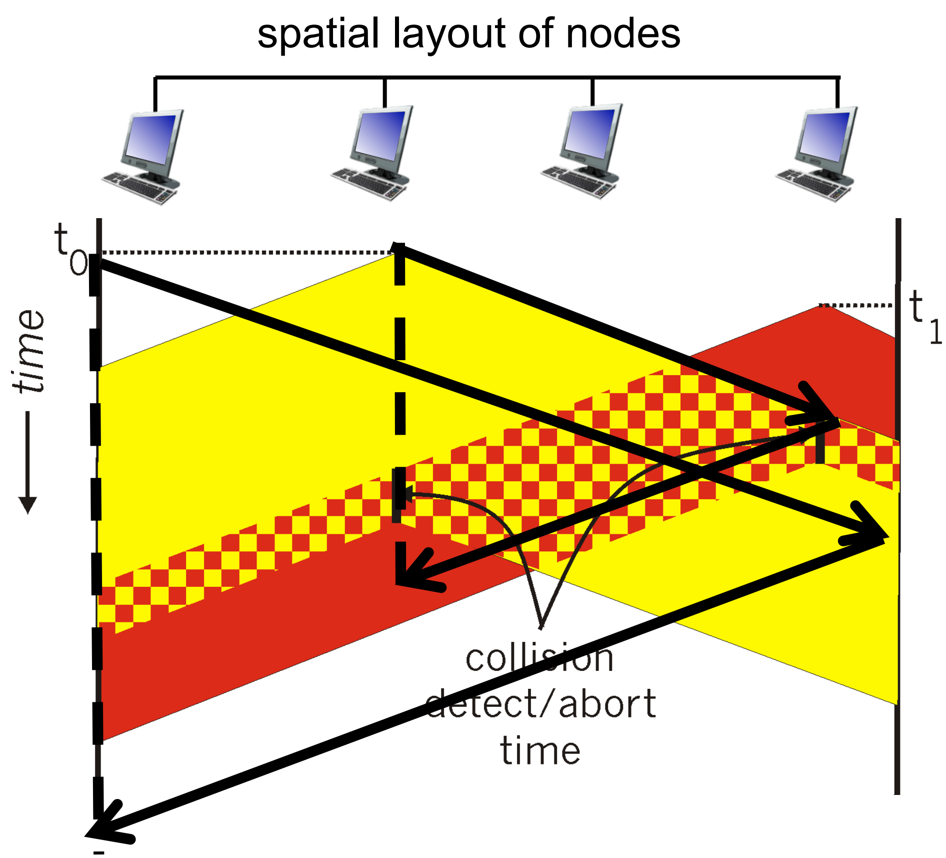
CSMA/CD (collision detection)



CSMA collisions with/without CD



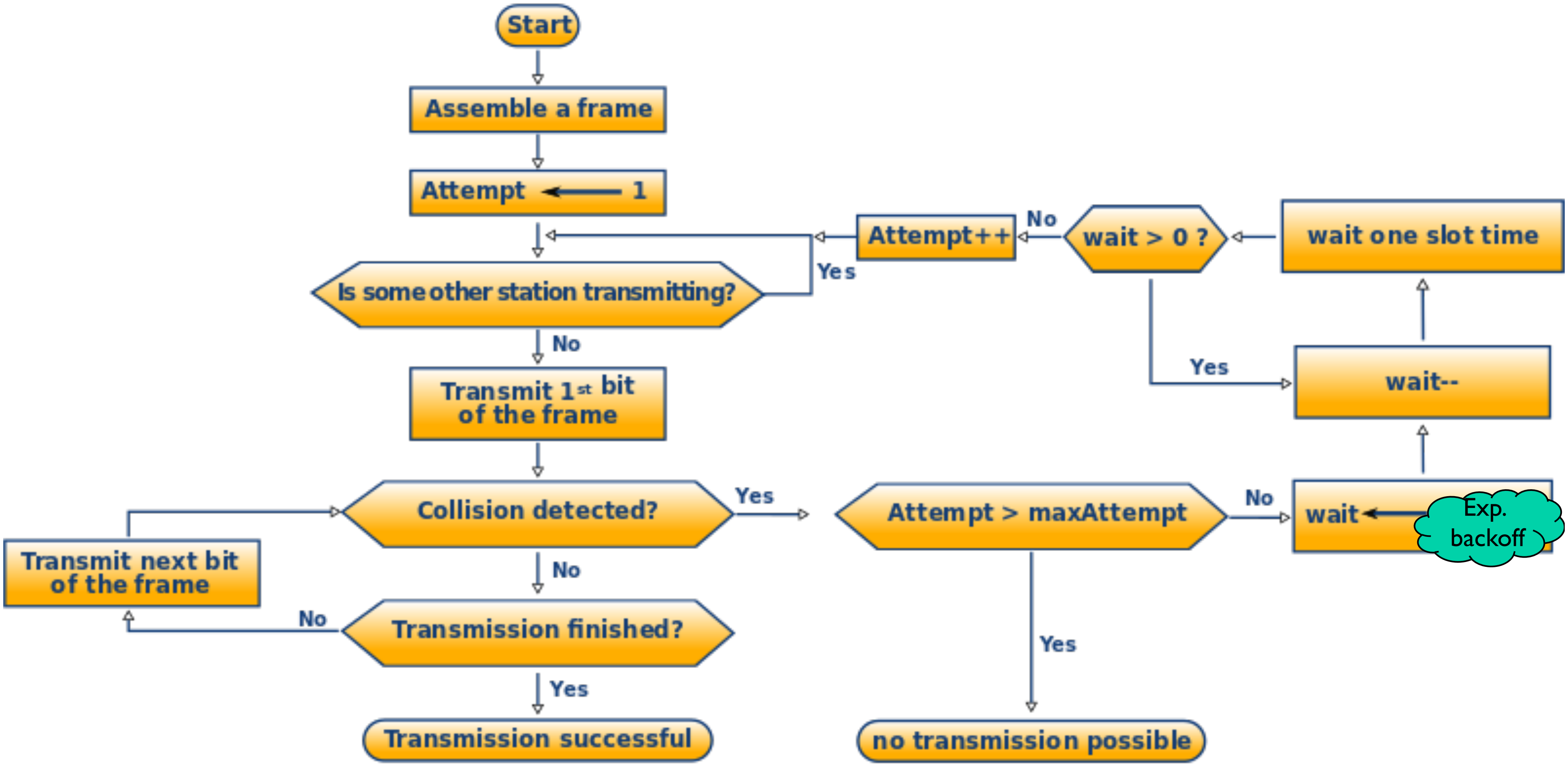
You are sure you "seized" the channel after $2d_{prop}$



Ethernet CSMA/CD algorithm

1. NIC receives datagram from network layer, creates frame
2. If NIC senses channel idle, starts frame transmission. If NIC senses channel busy, waits until channel idle, then transmits.
3. If NIC transmits entire frame without detecting another transmission, NIC is done with frame !
4. If NIC detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, NIC enters *binary (exponential) backoff:*
   - after *m*th collision, NIC chooses *K* at random from $\{0,1,2, \ldots, 2^m-1\}$. NIC waits K·512 bit times, returns to Step 2
   - longer backoff interval with more collisions

CSMA/CD Algorithm



http://en.wikipedia.org/wiki/Carrier_sense_multiple_access_with_collision_detection

CSMA/CD principles

- CSMA: 1-/p-/non- persistent
- CD
- Retransmission after exponential backoff:
  - Why random: to avoid synchronization
  - Why exponential: ~ TCP's multiplicative decrease
  - Why adjust after every collision: p essentially adapts to N
- Minimum frame=contention slot=512bits
  - =worst case RTT (for 10Mbps, length 2500m)

[Analysis of CSMA/CD]

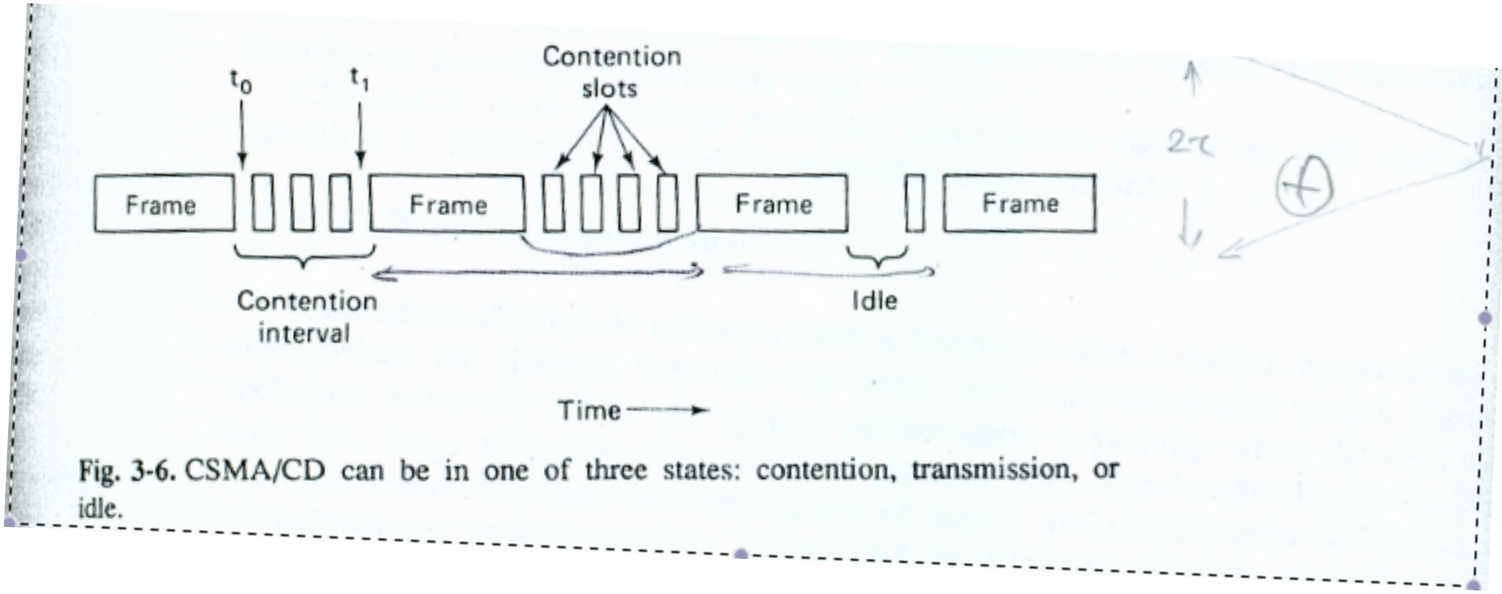


Fig. 3-6. CSMA/CD can be in one of three states: contention, transmission, or idle.

- Cycles of: successful transmission, idle and contention
- Contention slot: *S>=2*(prop.delay)*
- Channel efficiency= $\dfrac{\textit{Successful Tx time (in a cycle)}}{\textit{Duration of a cycle}}$
- Because analysis of exp. backoff is difficult
  - Ch.4, Problem 20's Simplification: k stations, each transmitting with prob. p in each slot [this looks like slotted Aloha]

CSMA/CD efficiency

- T_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame
- Result of analysis:

$$efficiency = \frac{1}{1 + 5t_{prop}/t_{trans}}$$

- efficiency goes to 1
  - as t_{prop} goes to 0
  - as t_{trans} goes to infinity
- better performance than ALOHA
- and simple, cheap, decentralized!

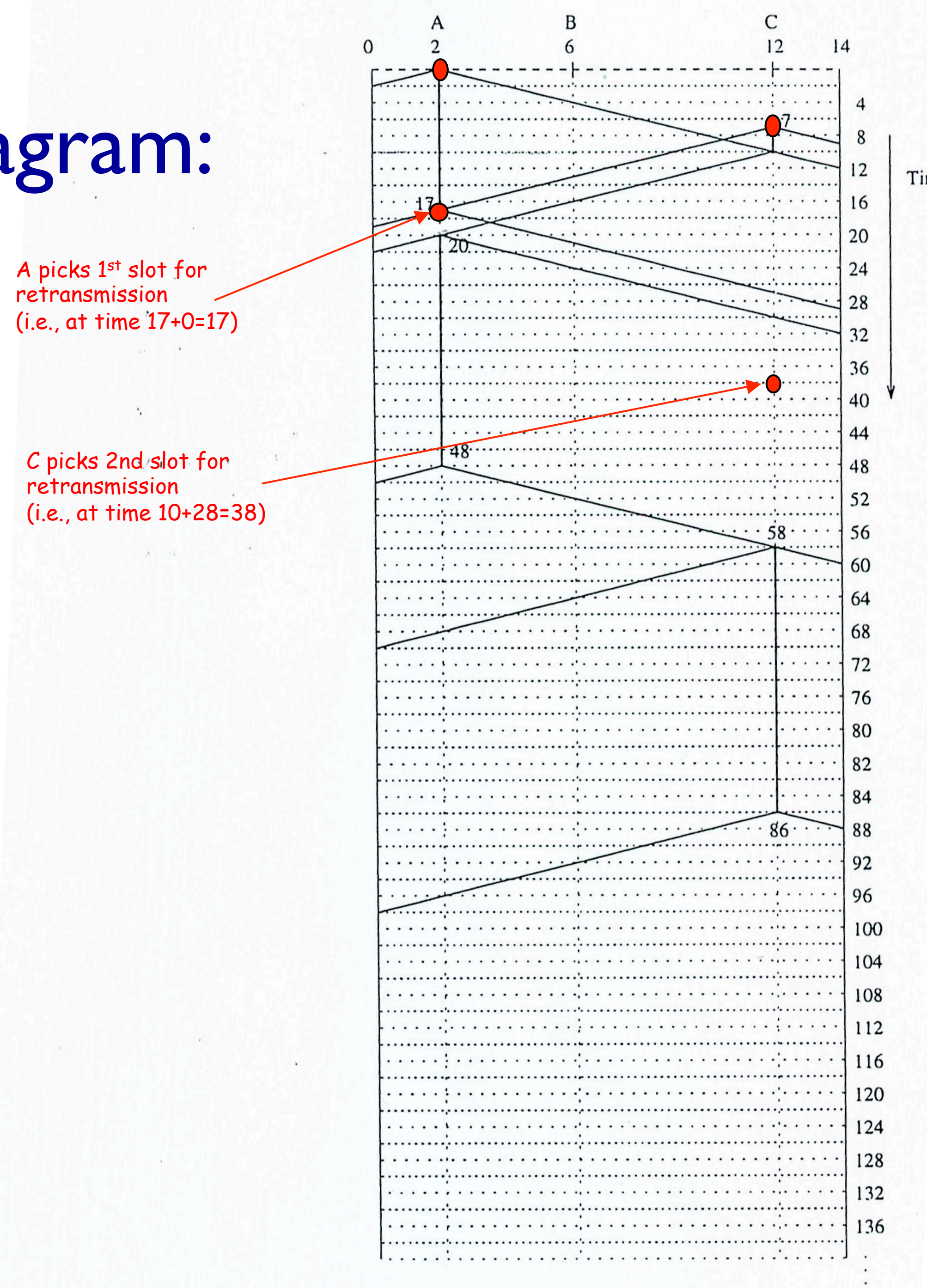
WiFi(802.11) vs. Ethernet (802.3)

- Both CSMA-based
- Collision detection: possible in Ethernet not on WiFi
- Collision avoidance in WiFi:
  - "transmit after you *seize* the channel"
  - Seize the channel through sensing in Ethernet
  - Seize the channel through RTS/CTS in WiFi
  - http://en.wikipedia.org/wiki/Carrier_sense_multiple_access_with_collision_avoidance

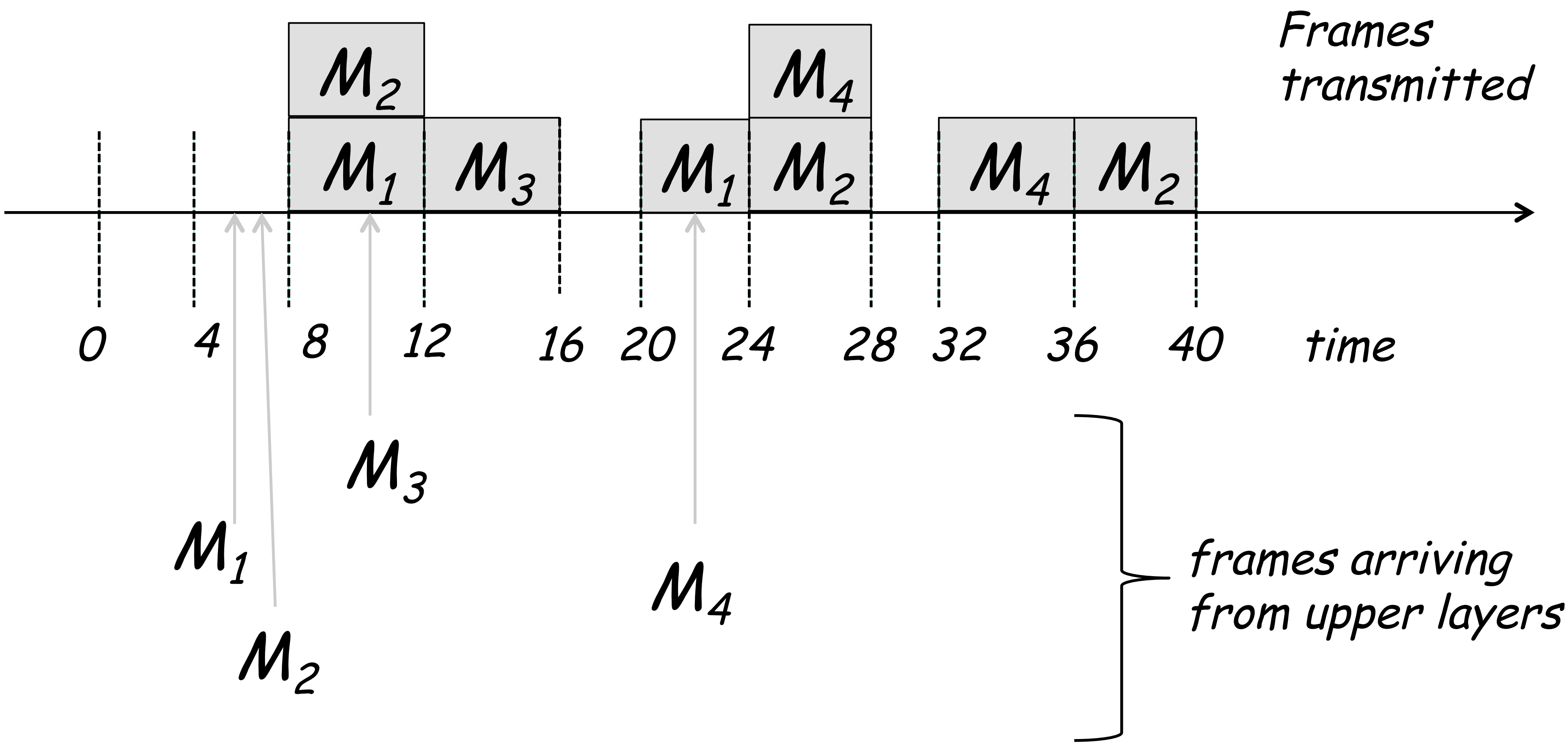
Complete the Diagram:

- The signal travels 1 unit distance in 1 unit time
- Frame=2*14=28 units
- Minislit=frame=28 units
- A wants to send a frame at time =0
- C wants to send a frame at time =7
- Assume instantaneous CD

- Protocol: CSMA/CD, 1-persistent, with exponential backoff
- Random numbers in [0,1] chosen by each station are given:
  - A: 0.45, 0.11, 0.71
  - C: 0.83, 0.41, 0.25



[Example (final S11→HW5)]



True or False: can these transmissions be generated by:

- Aloha? Slotted Aloha?
- CSMA? CSMA/CD? With exponential backoff?

[Example (final S12→HW5)]

3. *(20 Points)* **Random Access.** Consider a slotted Aloha system, where the time slot equals the fixed duration of each packet. Assume that there are 4 stations A,B,C,D sharing the medium.

   (a) Stations A,B,C,D receive one packet each from higher layers at times 1.3, 1.5, 2.6, 5.7 respectively. Show which transmissions take place when, according to the Slotted Aloha Protocol; describe all transmissions until all four packets have been successful. If needed, each station has access to the following sequence of random numbers, provided by a random number generator and drawn uniformly between 0 and 1:

   - Station A draws numbers: 0.31, 0.27, 0.78, 0.9, 0.9, 0.11, 0.22....
   - Station B draws numbers: 0.45, 0.28, 0.11, 0.83, 0.37, 0.22, 0.91....
   - Station C draws numbers: 0.1, 0.2, 0.3, 0.4, 0.5, ...
   - Station D draws numbers: 0.36, 0.77, 0.9, 0.1, 0.1, 0.1, 0.1, 0.83.....

   (b) In slotted aloha, a station transmits in each time slot with a given probability. What probabilities would you assign to each of the four stations so as to:

   i. maximize the efficiency of the protocol?
   ii. maximize fairness among the four stations?

   (c) Will the efficiency increase or decrease if we modify slotted aloha as follows:

   i. Get rid of slots and allow stations to transmit immediately?
   ii. Implement carrier sensing?
   iii. Implement collision detection?
   iv. Implement collision avoidance?

Summary of MAC protocols

- *channel partitioning,* by time, frequency or code
  - Time Division, Frequency Division
- *random access* (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*
  - Passing a token: FDDI, token ring
  - polling from central site: e.g. bluetooth
- Practical protocols mix and match these ideas
  - [E.g. protocols for cable internet access in 5.3.4]