Lecture 4: Datalink layer

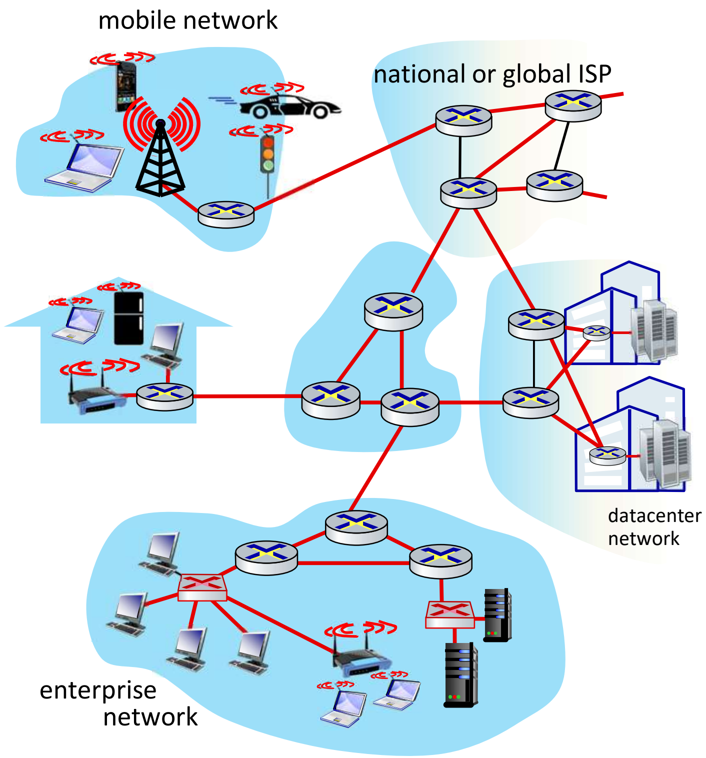
- Functionalities:
  - Encapsulation, addressing
  - Error detection and correction
  - Flow control
  - Media access control

Overview of Data link layer

Link layer: introduction

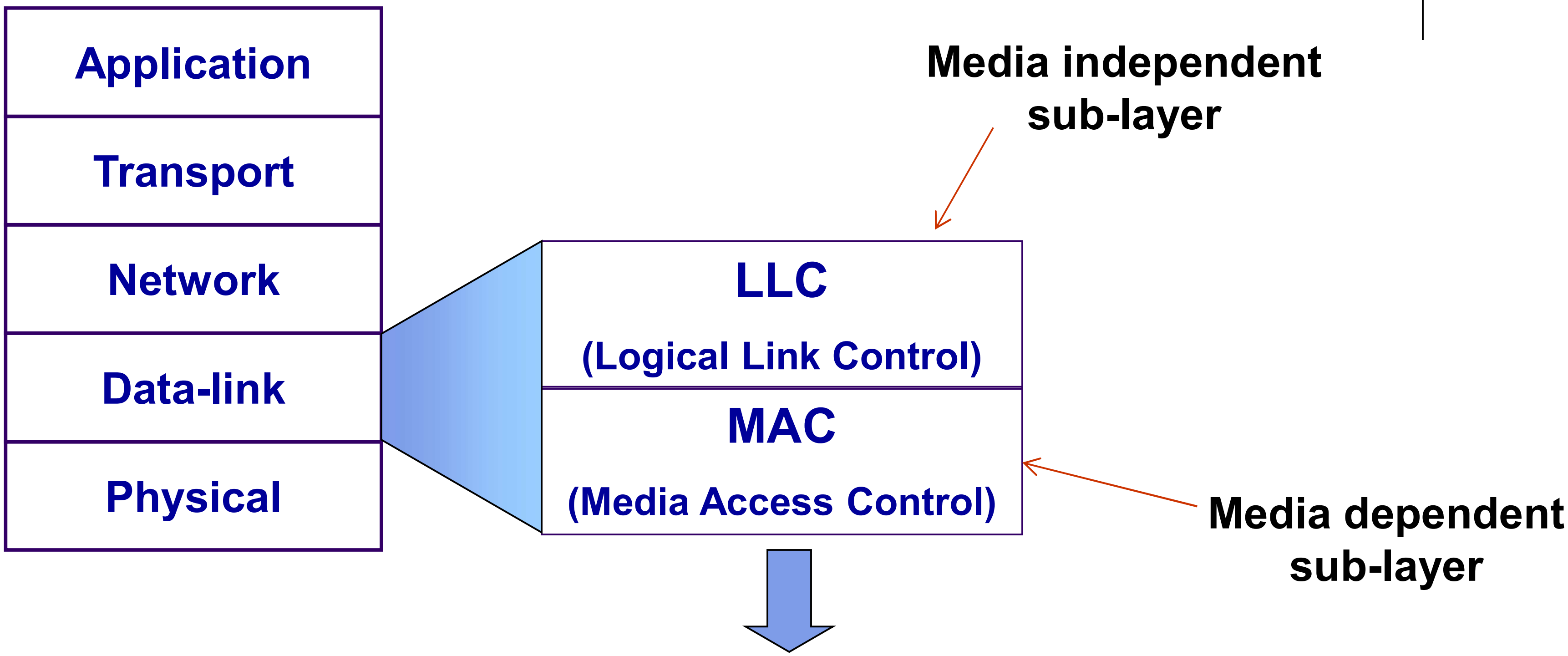
terminology:

- hosts and routers: nodes
- communication channels that connect adjacent nodes along communication path: links
  - wired
  - wireless
  - LANs
- layer-2 packet: *frame*, encapsulates datagram



> *link layer has responsibility of transferring datagram from one node to physically adjacent node over a link*

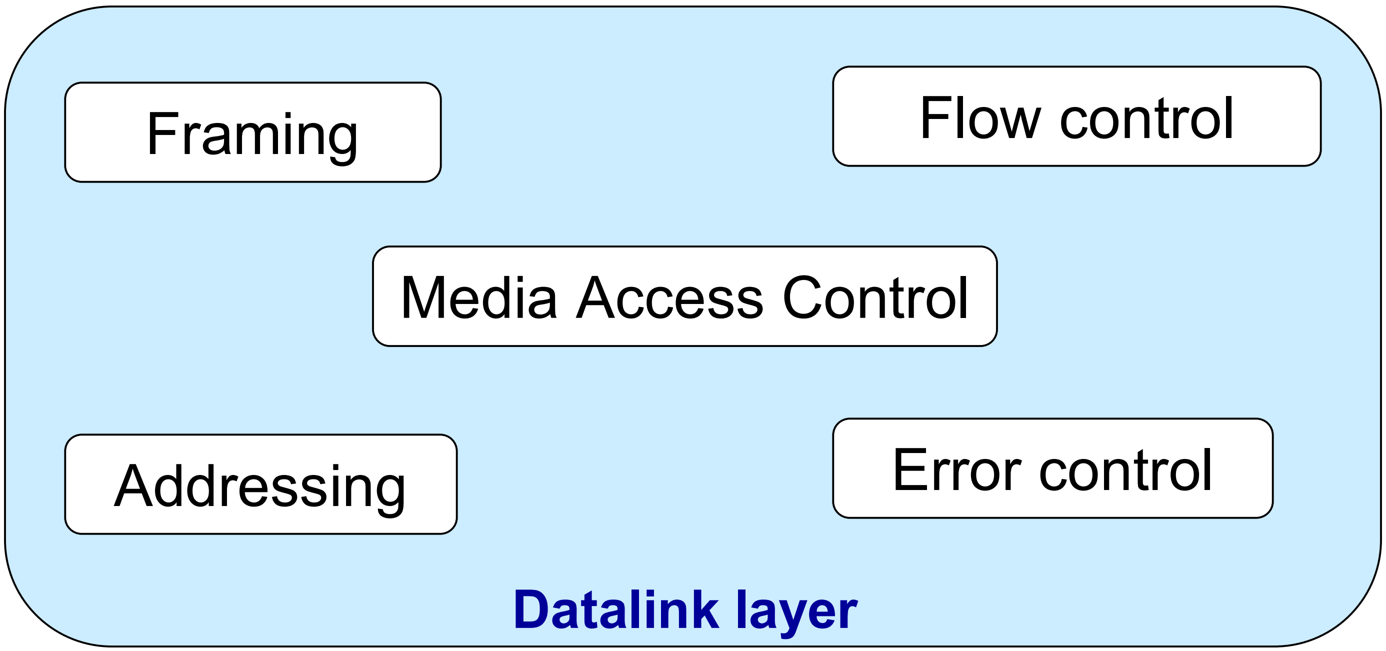
Datalink layer in Layer architecture



802.2 LLC					
802.3 Ethernet	802.4 Token Bus	802.5 Token Ring	802.11 Wi-Fi	…..	802.16 Wi-Max

IEEE 802.x series

Overview of Datalink services (functionalities)



Link layer: context

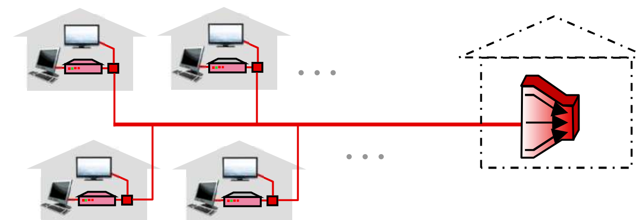
- datagram transferred by different link protocols over different links:
  - e.g., WiFi on first link, Ethernet on next link
- each link protocol provides different services
  - e.g., may or may not provide reliable data transfer over link

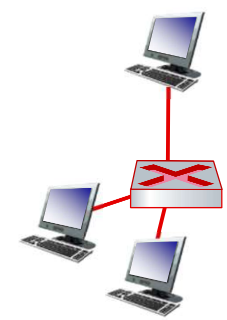
transportation analogy:

- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link-layer protocol
- travel agent = routing algorithm

Link layer: services

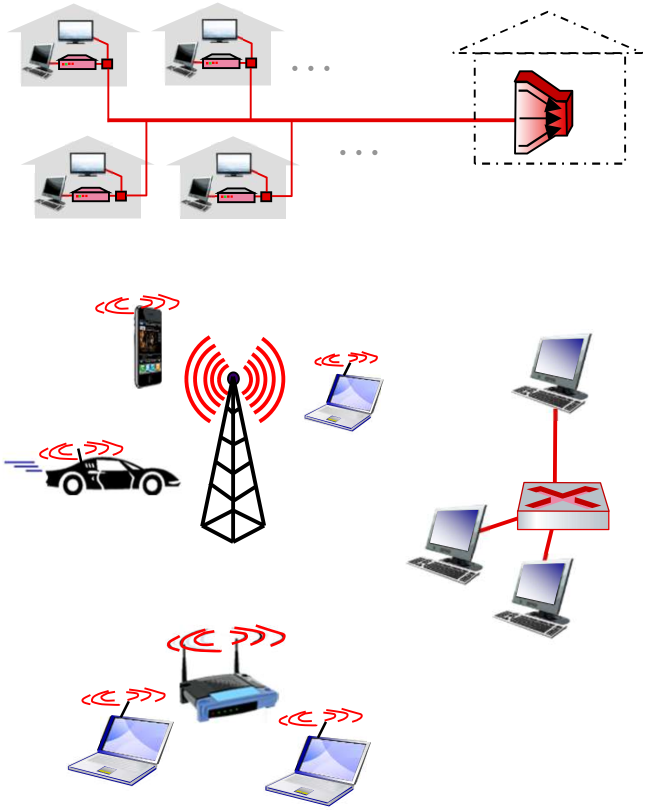
- framing, link access:
  - encapsulate datagram into frame, adding header, trailer
  - channel access if shared medium
  - “MAC” addresses in frame headers identify source, destination (different from IP address!)
- Media access control:
  - If the nodes in the network share common media, a Media access control protocol is required





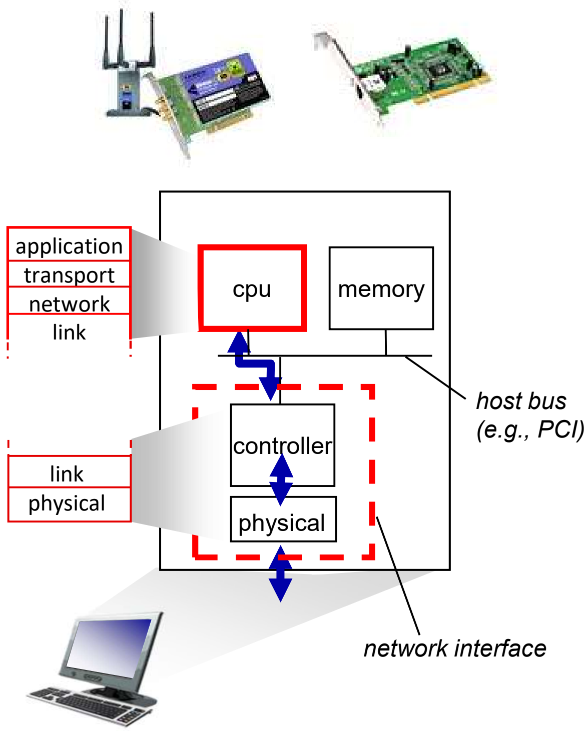
Link layer: services (more)

- flow control:
  - pacing between adjacent sending and receiving nodes
- error detection:
  - errors caused by signal attenuation, noise.
  - receiver detects errors, signals retransmission, or drops frame
- error correction:
  - receiver identifies *and corrects* bit error(s) without retransmission
- half-duplex and full-duplex:
  - with half duplex, nodes at both ends of link can transmit, but not at same time

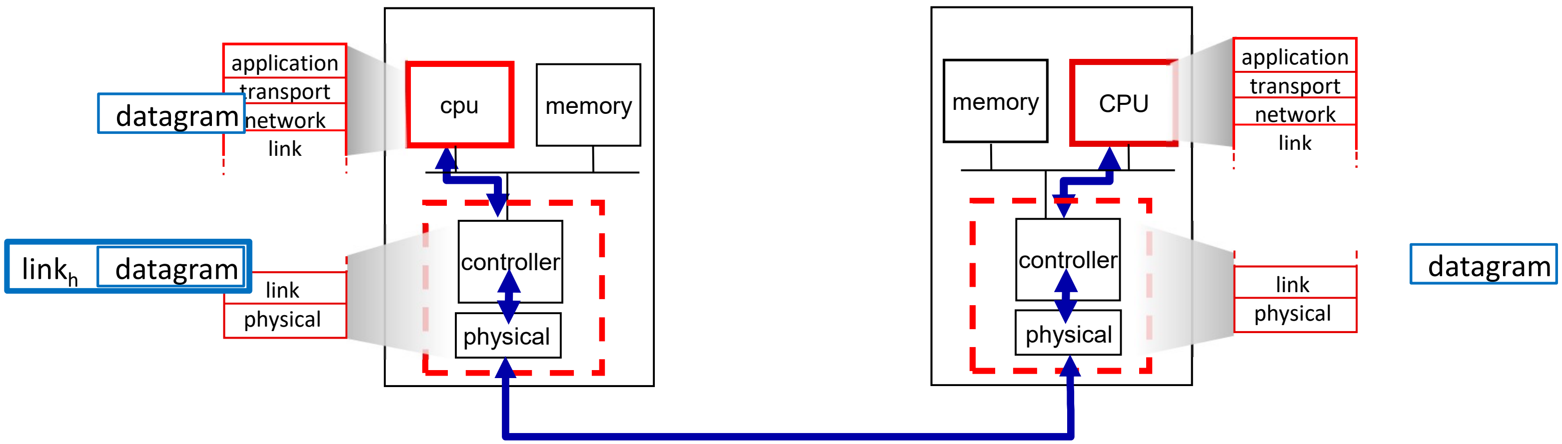


Where is the link layer implemented?

- in each-and-every host
- link layer implemented in *network interface card* (NIC) or on a chip
  - Ethernet, WiFi card or chip
  - implements link, physical layer
- attaches into host's system buses
- combination of hardware, software, firmware



Interfaces communicating



sending side:

- encapsulates datagram in frame
- adds error checking bits, reliable data transfer, flow control, etc.

receiving side:

- looks for errors, reliable data transfer, flow control, etc.
- extracts datagram, passes to upper layer at receiving side

Identifier: MAC address

- MAC address: 48 bit, organized by IEEE
- Each port is assigned one MAC
  - Cannot be changed
  - Physical address
- No hierarchical system, flexible
  - MAC Address is unchanged when changing networks
- Broadcast address in LAN:

  FF-FF-FF-FF-FF-FF

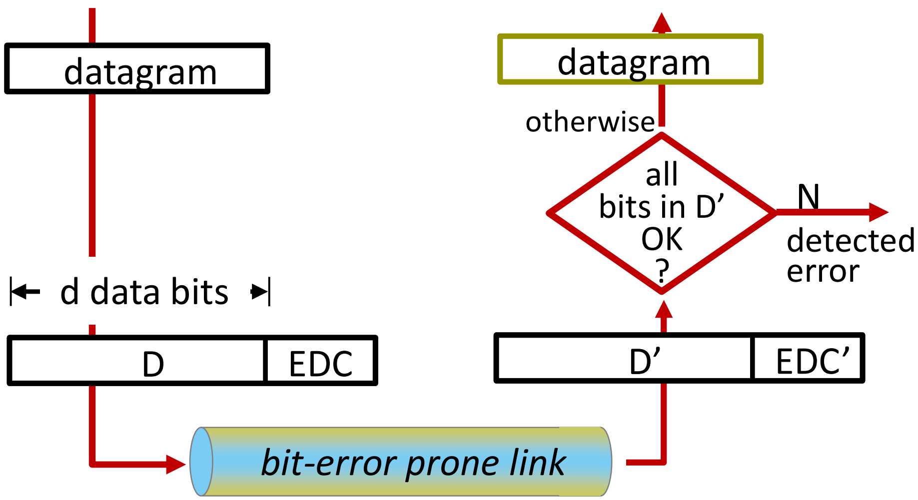
Error control

Error detection

Error correction

Principle of error detection

EDC: error detection and correction bits (e.g., redundancy)
D: data protected by error checking, may include header fields



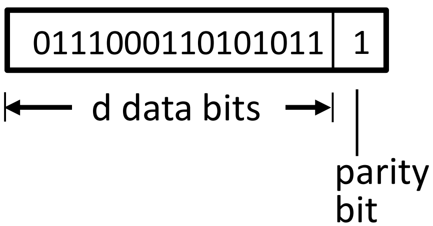
Error detection not 100% reliable!

- protocol may miss some errors, but rarely
- larger EDC field yields better detection and correction

Parity checking

single bit parity:

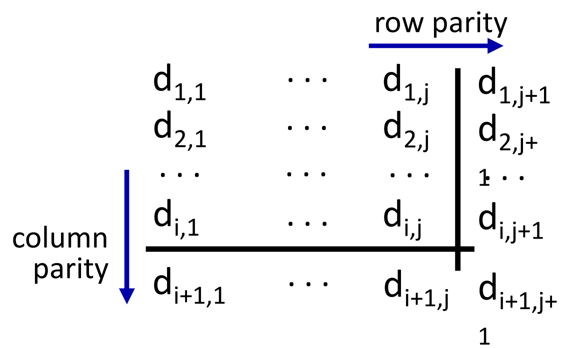
- detect single bit



Even parity: set parity bit so there is an even number of 1's

two-dimensional bit parity:

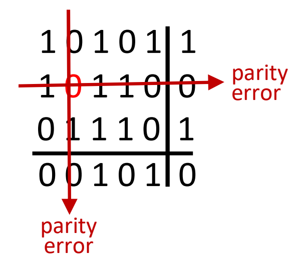
- detect *and correct* single bit errors



no errors:

1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0

detected and correctable single-bit error:



* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Checksum

Goal: detect errors (*i.e.,* flipped bits) in transmitted segment

sender:

- Divide data to n-bit segments
- Calculate the sums of segments. If having overflow bits, add them to the results
- checksum: addition (one's complement sum) of segment content

receiver:

- Divide data to n-bit segments
- Calculate the sums of segments. If having overflow bits, add them to the results
- Add the received checksum with the results
- Check the final outcome
  - Contains 0 - error detected
  - Only 1 - no error detected. *But maybe errors nonetheless?*

checksum: Example

```
Data: 0011 0110 1000
checksum 4 bit:
     0011
   + 0110
     1000
    -----
    10001
Overflow
  bit      1
    -----
     0010
Reverse bit →  checksum: 1101
Transfering bits: 0011 0110 1000 1101
```

checksum: receiving

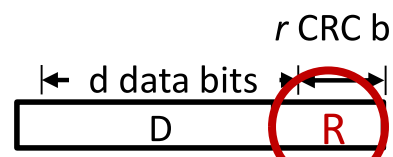
Receiving bit: 0011 0110 1000 1101

Checking:

```
     0011
     0110
+    1000
     1101
    -----
    11110
Overflow bit
        1
     1111 → No errors
```

Cyclic Redundancy Check (CRC)

- more powerful error-detection coding
- D: data bits (given, think of these as a binary number)
- G: bit pattern (generator), of *r+1* bits (given)



bit pattern

<D,R> = D$*$2^r XOR R

formula for bit pattern

goal: choose *r* CRC bits, R, such that <D,R> exactly divisible by G (mod 2)

- receiver knows G, divides <D,R> by G. If non-zero remainder: error detected!
- can detect all burst errors less than r+1 bits
- widely used in practice (Ethernet, 802.11 WiFi)

Cyclic Redundancy Check (CRC): example

We want:

$D \cdot 2^r \ \ XOR \ \ R = nG$

or equivalently:

$D \cdot 2^r = nG \ \ XOR \ \ R$

or equivalently:

if we divide $D \cdot 2^r$ by G, want remainder R to satisfy:

$$R = remainder\left[\frac{D \cdot 2^r}{G}\right]$$

```
   G                1 0 1 0 1 1
1 0 0 1 ) 1 0 1 1 1 0 0 0 0      D*2^r
          1 0 0 1
              1 0 1
              0 0 0
              1 0 1 0
              1 0 0 1
                  1 1 0
                  0 0 0
                  1 1 0 0
                  1 0 0 1
                    1 0 1 0
                    1 0 0 1
                      0 1 1
                        R
```

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

CRC under polynomial form

- 1011 <-> $x^3 + x + 1$
- Example of some CRC using in the pratice:
  - CRC-8 = $x^8 + x^2 + x + 1$
  - CRC-12 = $x^{12}+x^{11}+x^3+x^2+x$
  - CRC-16-CCITT = $x^{16} + x^{12} + x^5 + 1$
  - CRC-32 = $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- The longer G is, the more possible that CRC detects errors.
- CRC is widely used in the practice
  - Wi-fi, ATM, Ethernet…
  - Operation XOR is implemented in hardware
  - Capable to detect less than r+1 bits errors

CRC – Example

Frame : 1101011011

Generator : $G(x) = x^4 + x + 1 \rightarrow P = 10011$

Dividend : Fk = 11010110110000

R = Fk mod P = 1110

Send : 11010110111110

CRC – Example

1101011011**0000** | 10011

10011

~~0~~10011

10011

~~00000~~10110

10011

~~00~~10100

10011

~~00~~1110 → Remainder:CRC

CRC – Check 11010110111110

```
11010110111110 | 10011
               |--------
10011
------
010011
 10011
 -----
 0000010111
      10011
      -----
      0010011
        10011
        -----
        000000 → No errors
```

CRC – Check 11010010111110

11010010111110 | 10011

10011

~~0~~10010

10011

~~0000~~11011

10011

~~0~~10001

10011

~~000~~10110

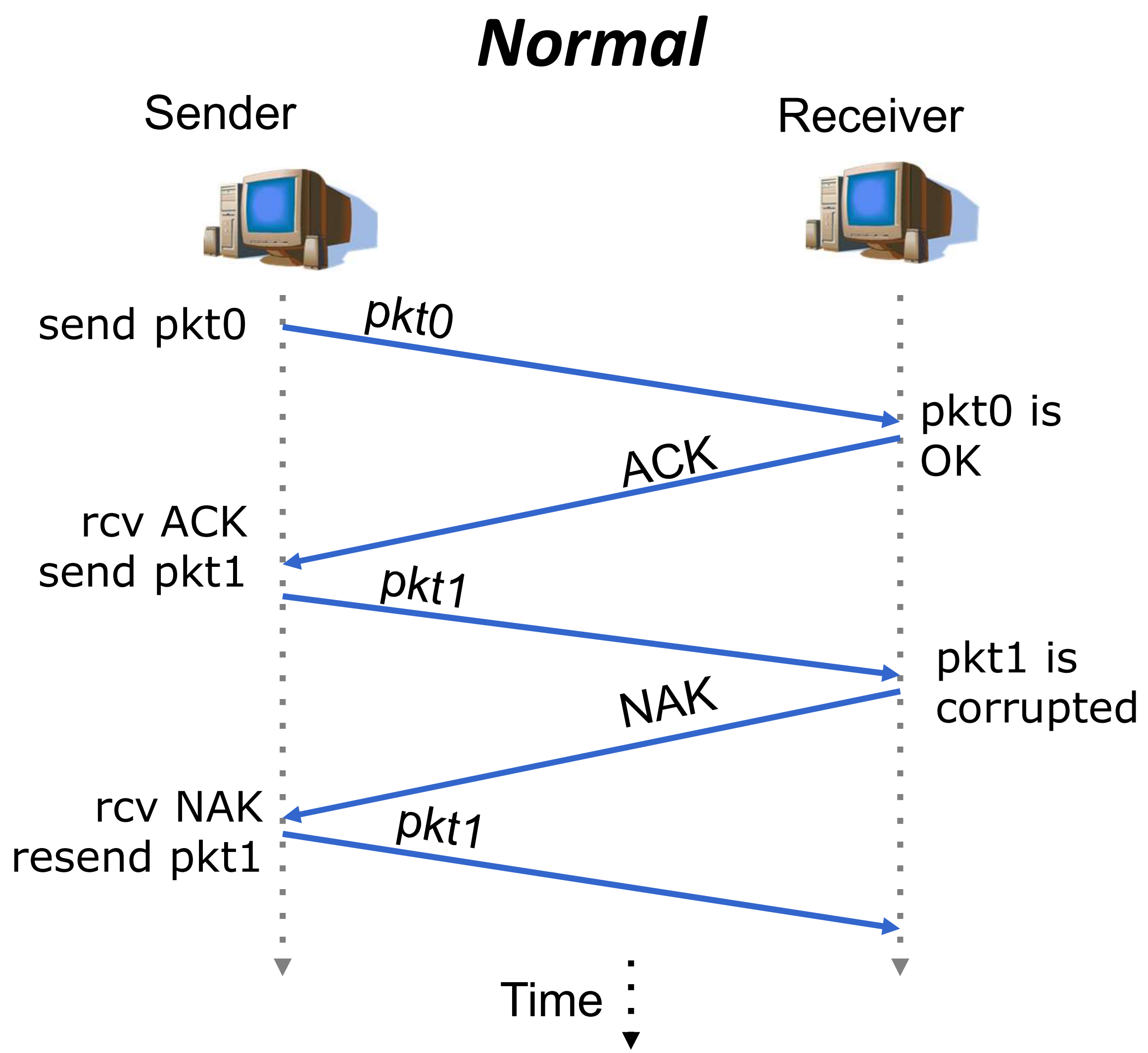
10011

00101 → not 0 → errors

Reaction when errors detected

- Objective: assure that data are transmitted correctly even though the chanel is not realiable.
- Condition
  - Data fram must be transmitted correctly
  - Negligible transmission delay.
- Possible errors
  - Whole frame loss
  - Error frame
  - Loss of error warning message
- Popular techniques:
  - Error detection (as we seen)
  - Acknowledgement/confirmation
  - Retransmis after timeout
  - Retransmis after a clear confirmation that frame is not arrived
- ARQ technique: (automatic repeat request). There are 3 versions:
  - Stop and Wait ARQ
  - Go Back N ARQ
  - Selective Reject ARQ
- Similar to techniques used in flow control.

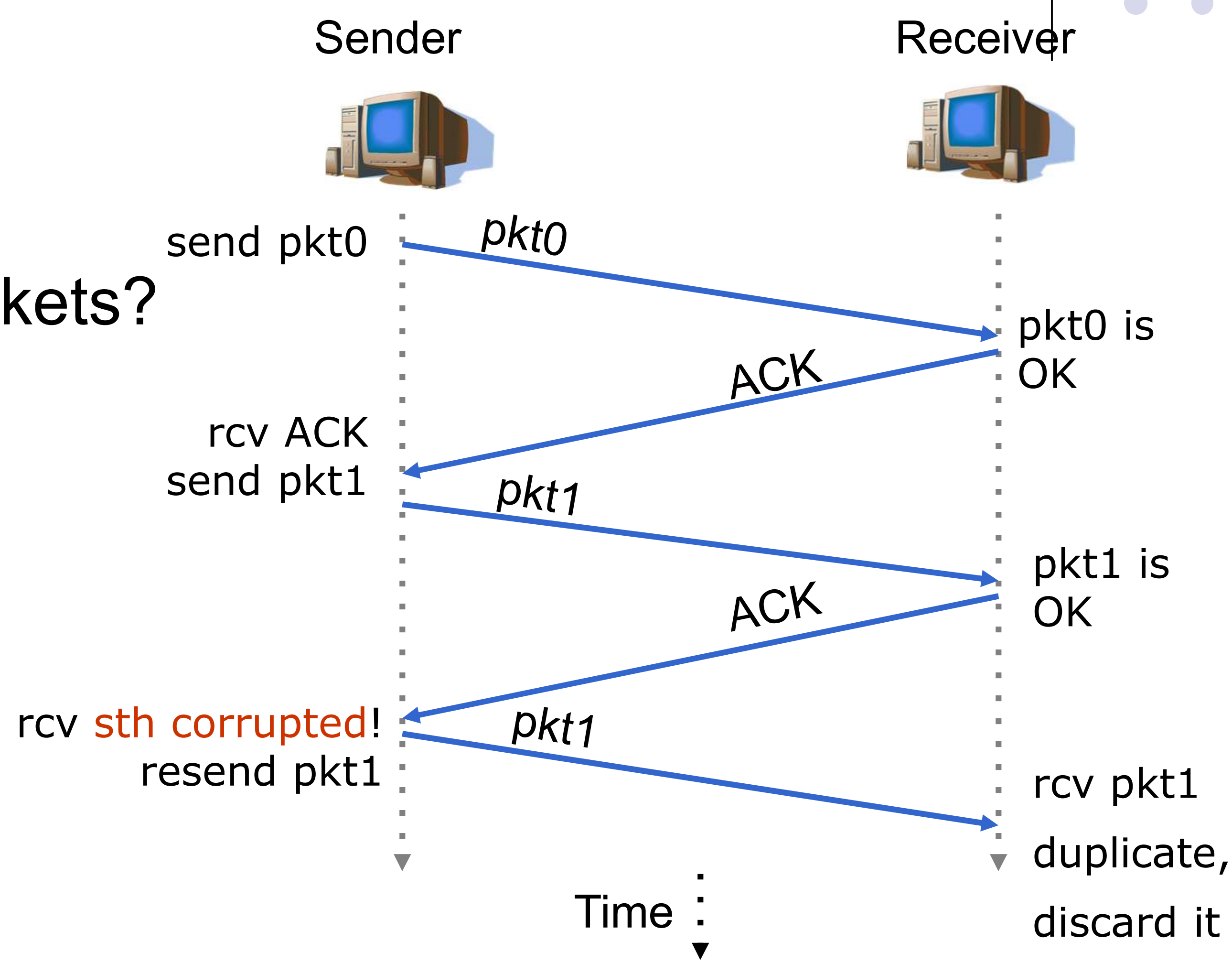
Stop-and-wait ARQ



Stop-and-wait ARQ

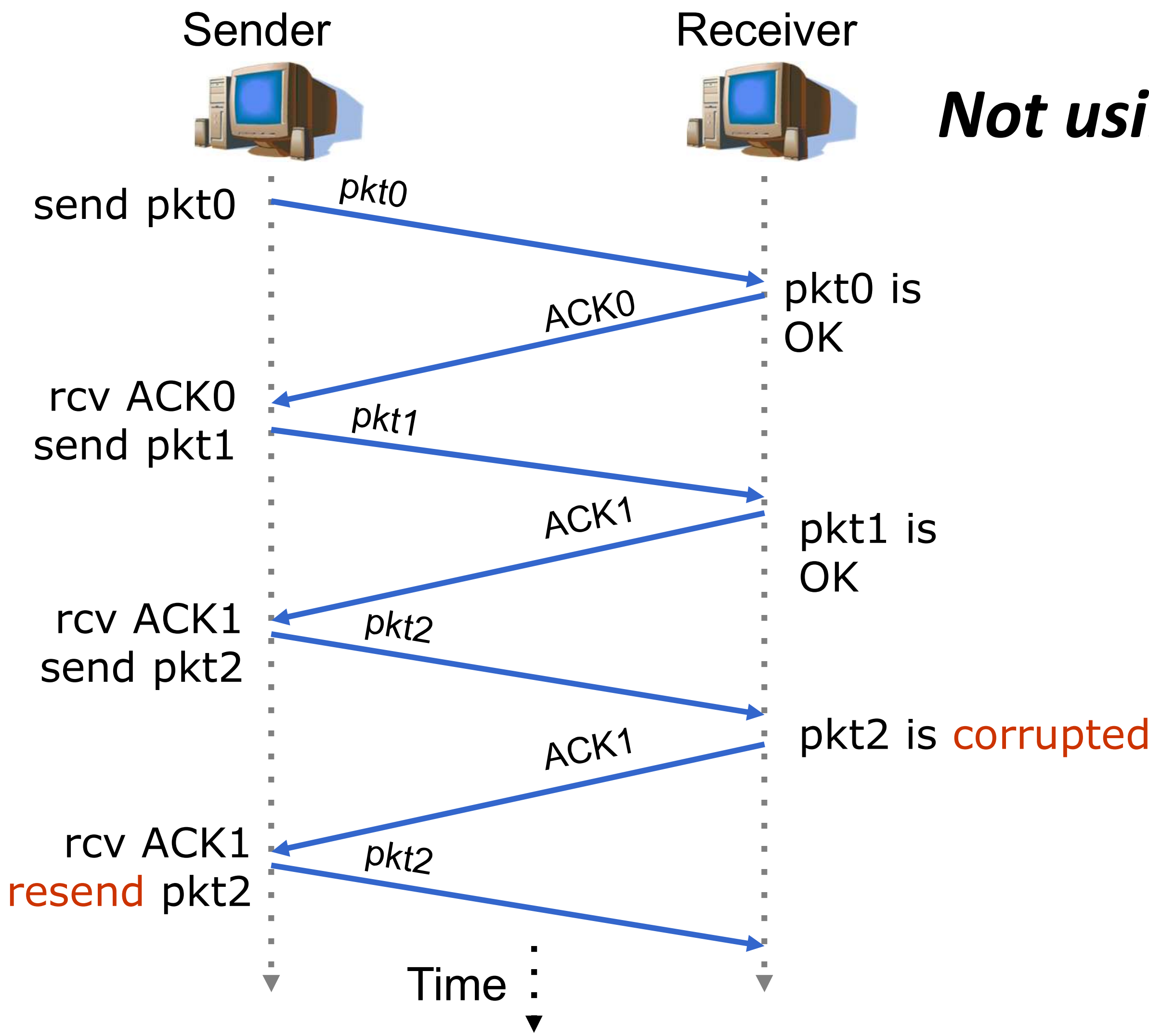
ACK/NAK errors

- Resend
- Repetitive packets?
- Add Seq. No.



Stop-and-wait ARQ

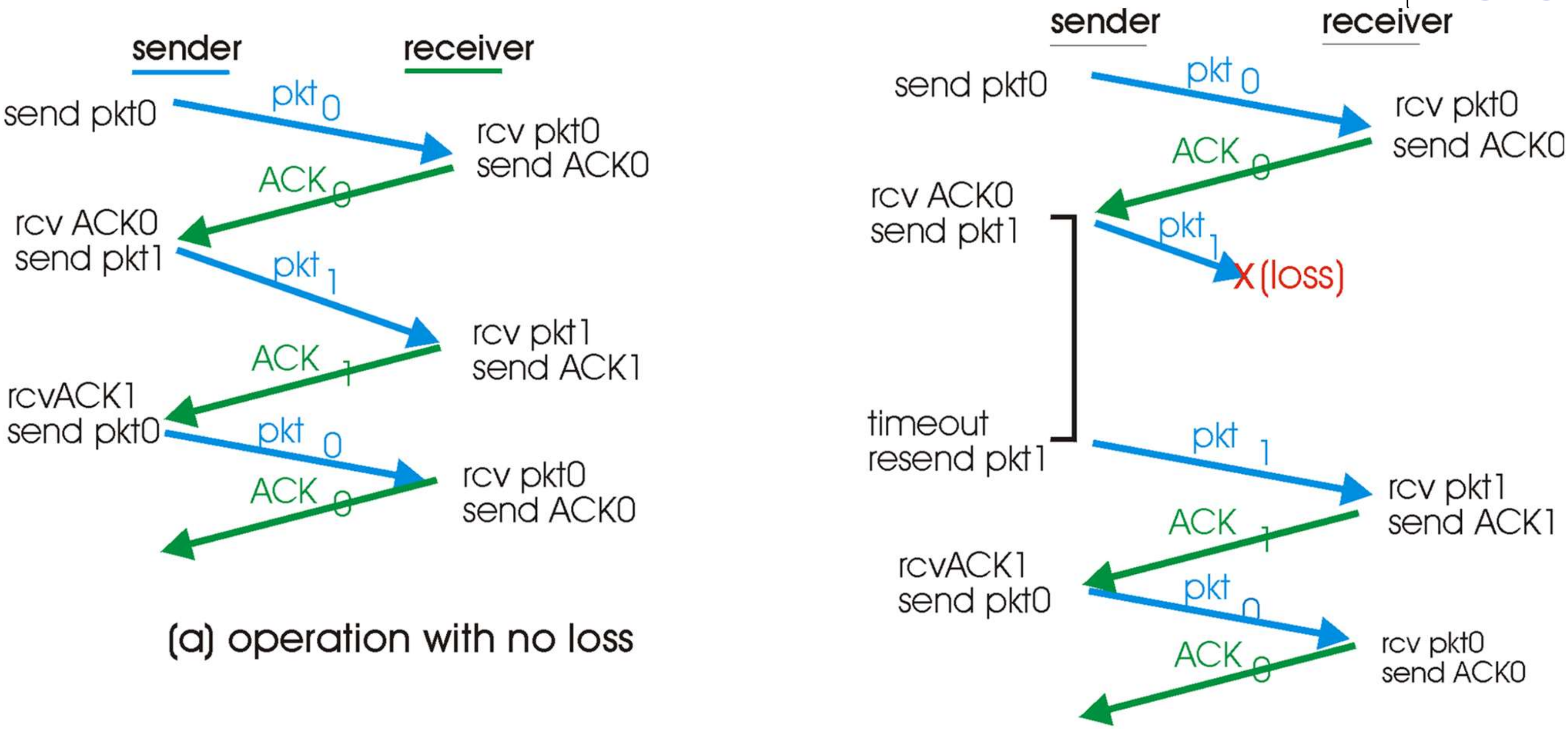
Not using NAK



Stop-and-wait ARQ: Lost ACK?

- Data and ACK can be lost
  - Can't receive ACK?
  - How to resend data?
- Mechanism
  - Packets will be numbered and sender automatically resends packets until receiving ACK of packets
  - Resending after time-out
- How long of timeout?
  - At least 1 RTT (Round Trip Time)
  - Each sent packets should have 1 timer
- Received packets but lost ACK?
  - Duplicate packets
  - Receiver will remove duplicate ones

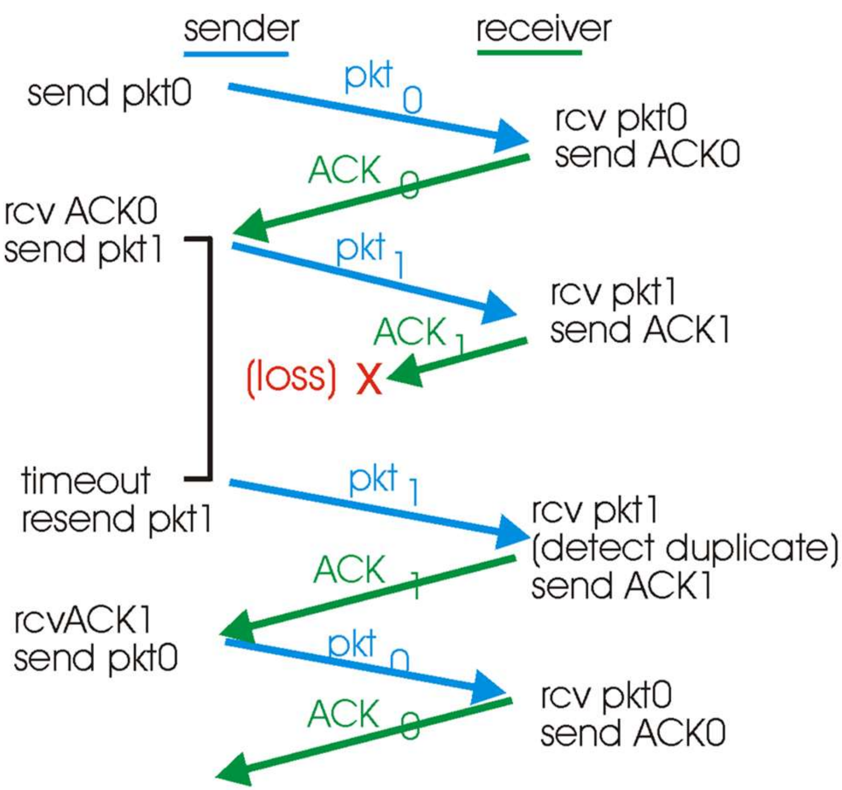
ARQ illustration



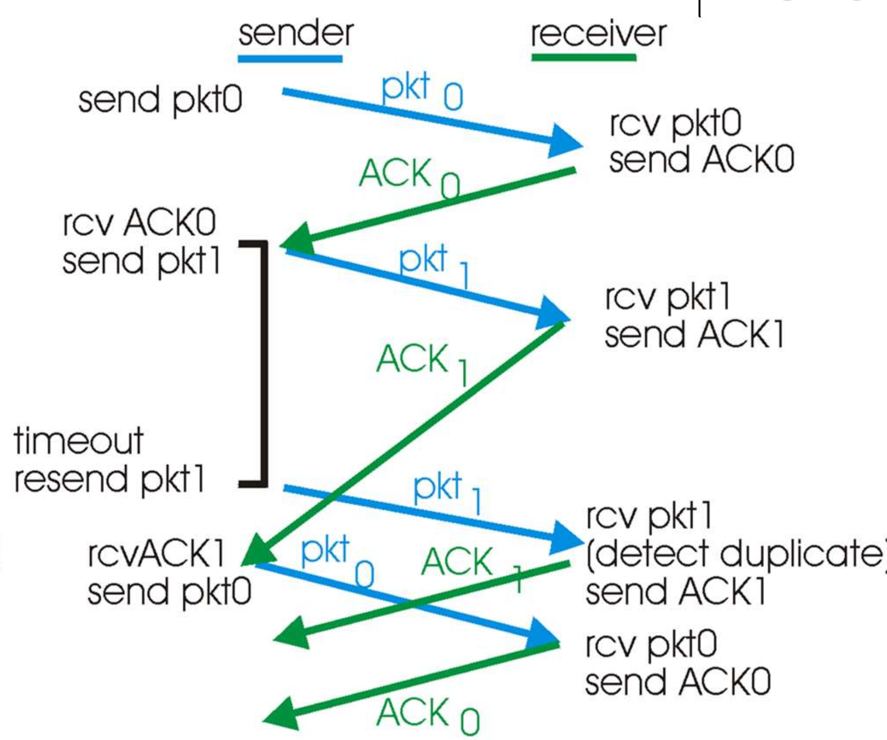
(a) operation with no loss

(b) lost packet

ARQ illustration (cont.)



(c) lost ACK



(d) premature timeout

Flow control

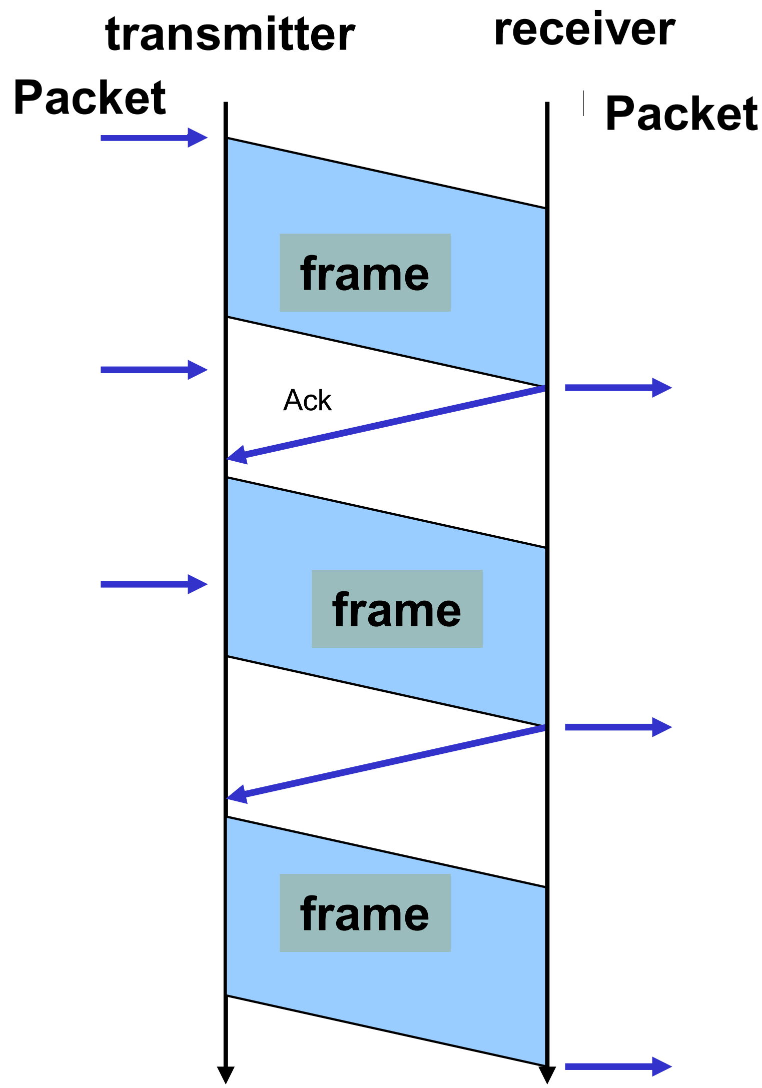
What is flow control

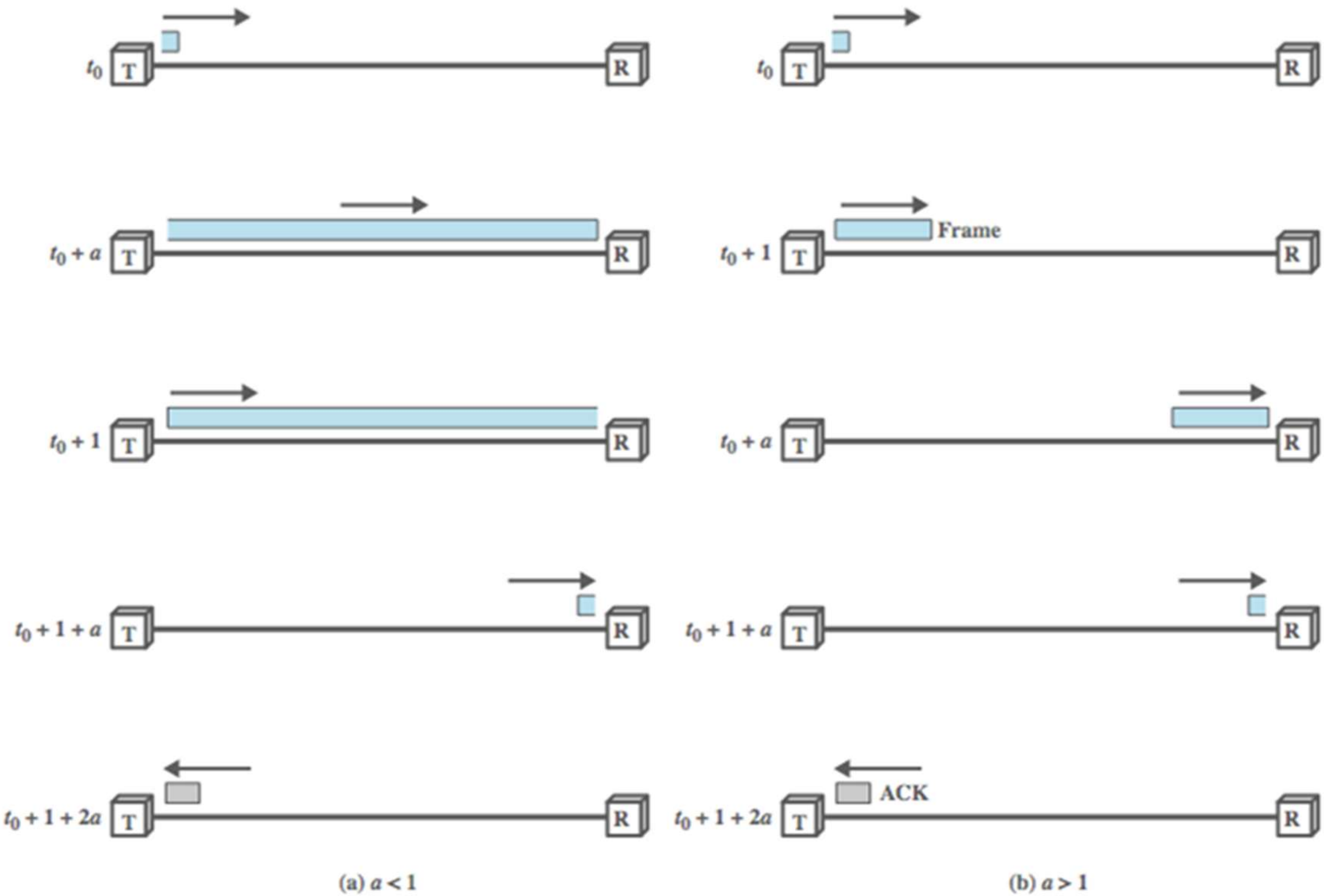
- Goal: Make sure that the sender does not overload the receiver
- Why overloading?
  - The receiver stores data frame in buffer.
  - Receiver performs some processing before deliver data to the upper level.
  - Buffer could be full, leaving no space for receiving more frame → some data fram must be dropped.
- Problem of errors in transmission is excluded
  - All frames are transmitted to correct receiver without error
  - Propagation time is small and could be ignored
- Solution
  - Stop-and-wait mechanism
  - Sliding window mechanism

Stop-and-wait

- Principles
  - Transmitter sends a single frame
  - Receiver receives the fram, process and then informs the transmitter that it is ready to receives next frames by a clear acknowledgement (ACK).
  - Transmitter waits until reception of the ACK before sending next frames.

Stop-and-wait





t_0
T
R
$t_0 + a$
$t_0 + 1$
$t_0 + 1 + a$
$t_0 + 1 + 2a$
(a) $a < 1$
Frame
ACK
(b) $a > 1$

Stop-and-wait

- Advantage
  - Simple, suitable for transmission of big size frames
- Weakness
  - When frames are small, the transmission chanel are not used efficiently.
  - Cannot use often for big size frame due to
    - Limitation in buffer size
    - Big size frame prones to bigger error probability
    - In shared medium, it is not convenient to leave one station using medium for long time

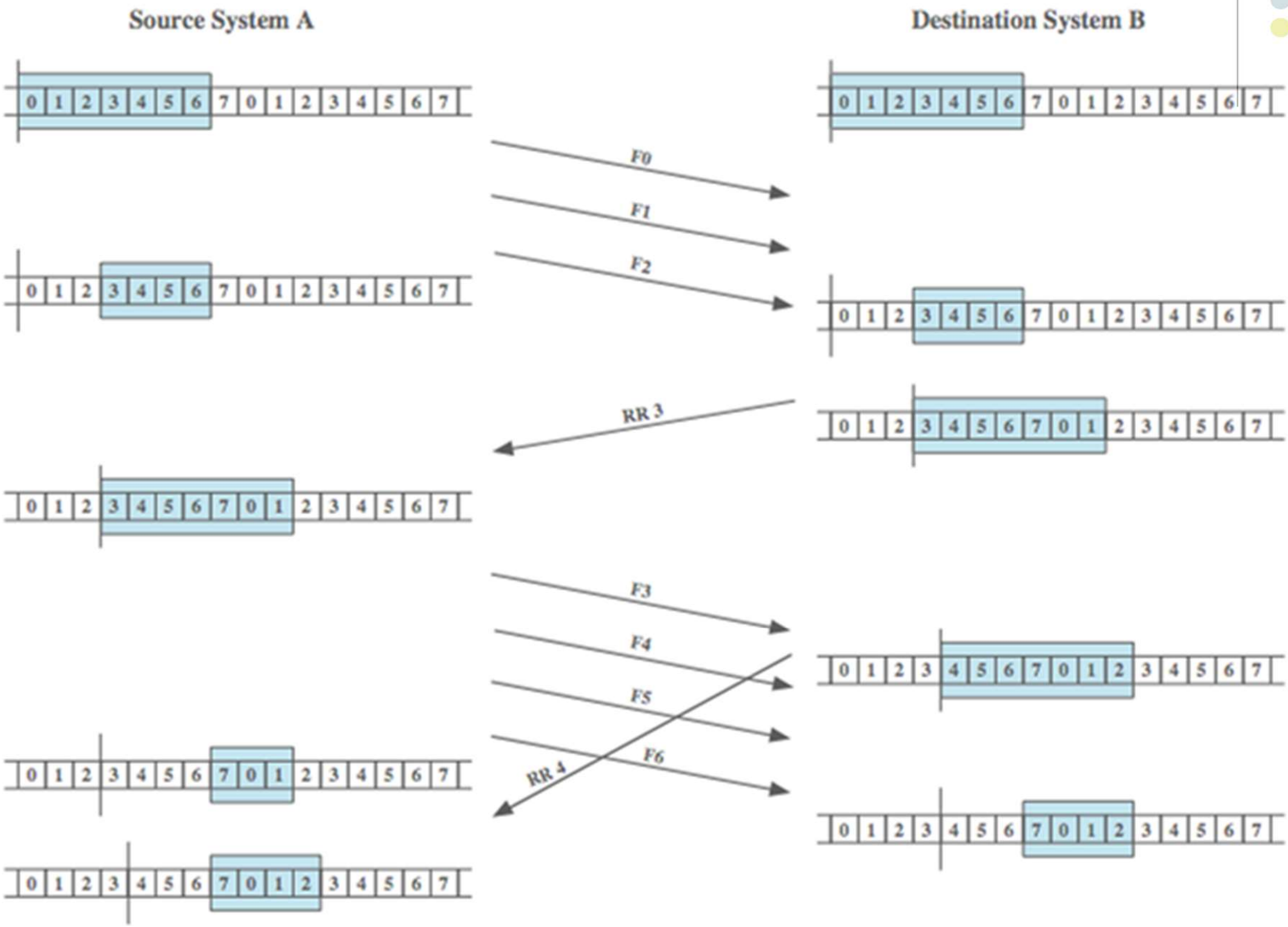
Sliding window: principle

- Transmitter sends more than one frame without waiting in order to reduce waiting time
- Transmitted frame without ACK will still be stored in buffer.
- Number of frame to be transmitted without ACK depends on the size of buffer at transmitter
- When transmitter receives ACK, it realises the succesfully transmitted frame from buffers
- Transmitter continues sending a number of frame equivalent to the number of succesfully trasmitted frames.

Sliding window: principle

- Assume that A and B are two stations connected by a full duplex media
  - B has a buffer size of n frame.
  - B can receives n frame without sending ACK
- Acknowledgement
  - In order to keep track of ACKed frames. It is neccessary to number frames.
  - B acknowledge a frame by telling A which fram B is waiting for (by number of frame), implicitely saying that B receives well all other frame before that.
  - One ACK frame serves for acknowledes several frames.

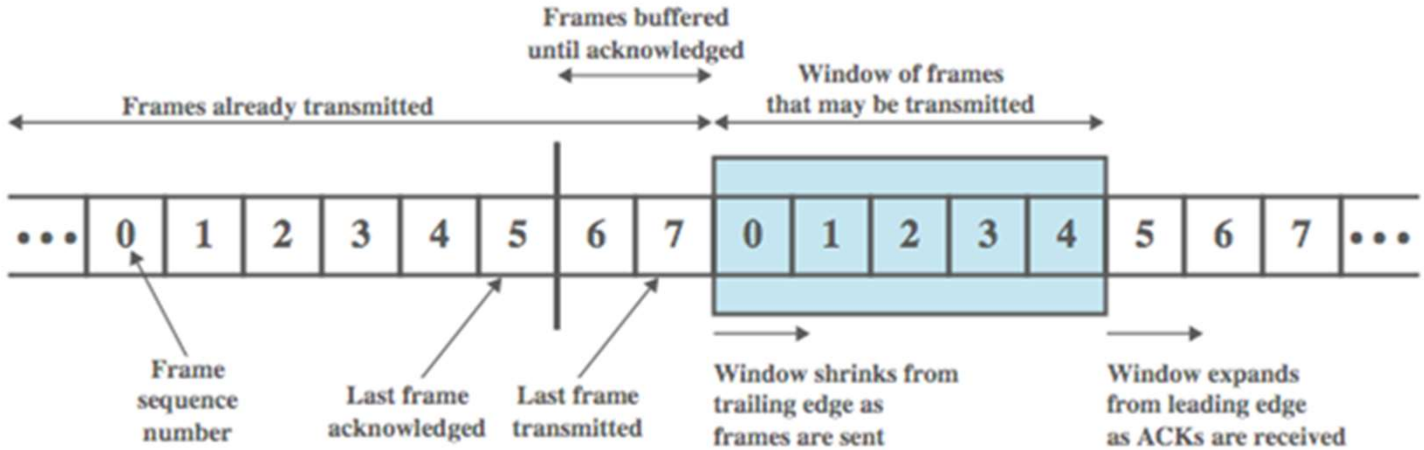
Sliding windows: principle



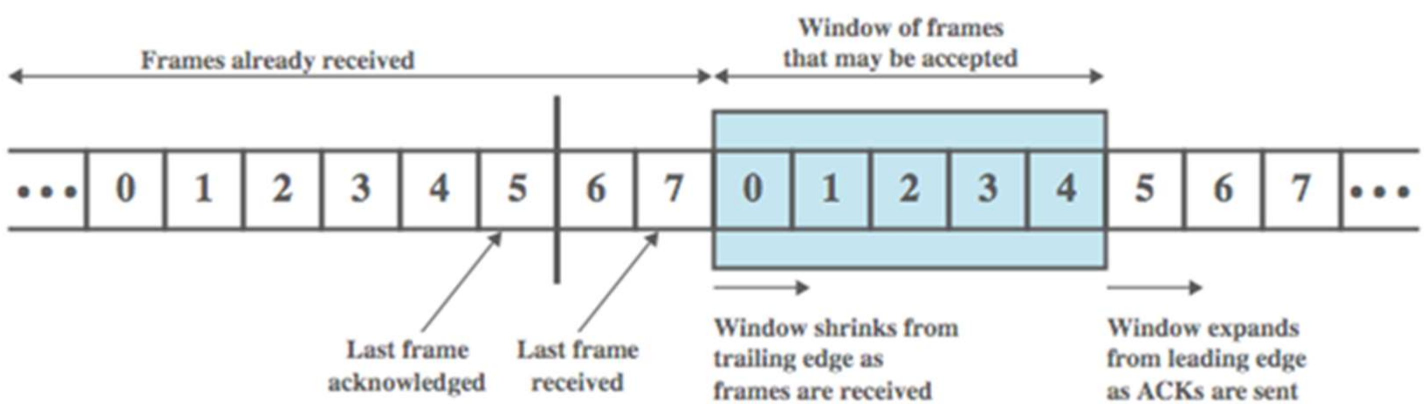
Window list the frames to transmit

Window list the frames in waiting to receive

Sliding windows



(a) Sender's perspective



(b) Receiver's perspective

Sliding windows

- Frame are numbered. The maximum number must not be smaller than the size of the window.
- Frame are ACKed by another message with number
- Accumulated ACK: If frame 1,2,3,4 are well receive, just send ACK 4
- ACK with number k means all frame k-1, k-2 …already well received.

Sliding windows

- Transmitter needs to manage some information:
  - List of frames transmitted sucessfully
  - List of frames transmitted without ACK
  - List of frames to be sent immediatly
  - List of frames NOT to be sent immediately
- Receiver keep tracks of
  - List of frames well received
  - List of frames expected to receive

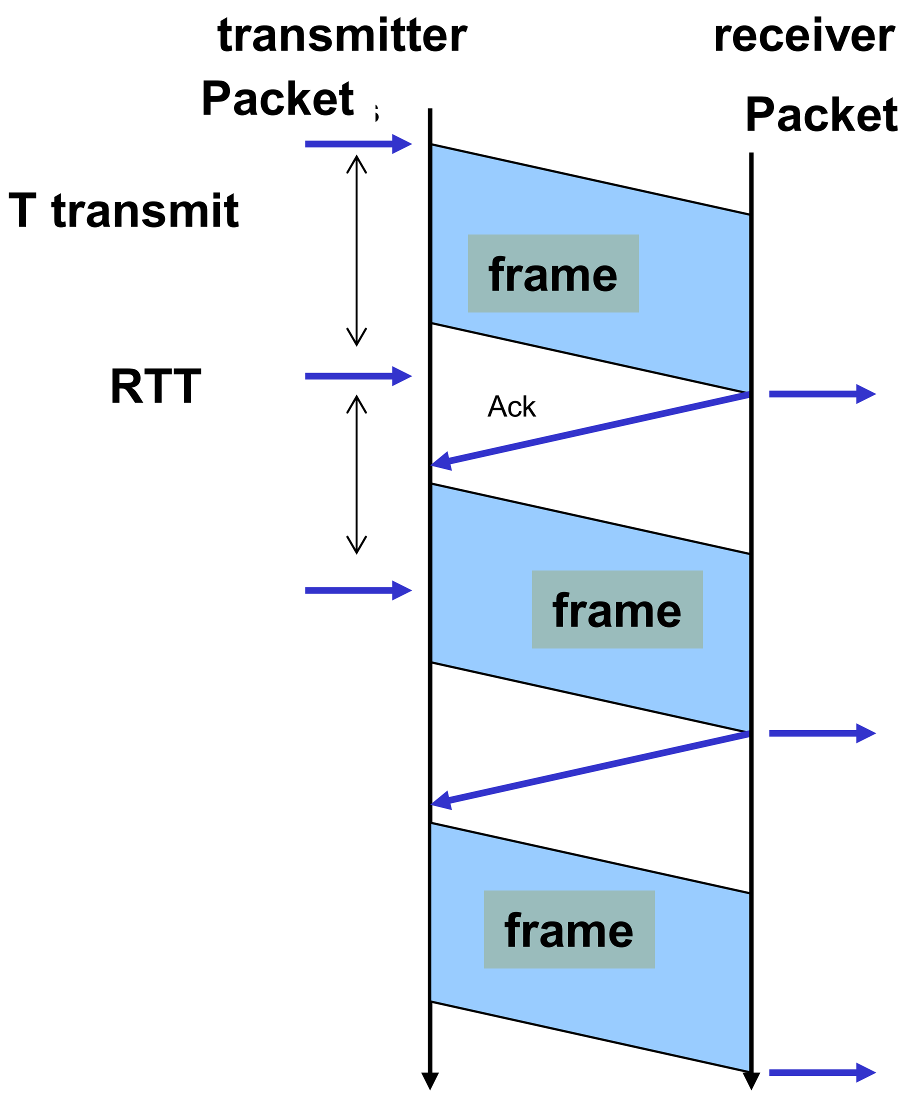
Piggy backing

- A and B transmitte data in both sides
  - When B needs to send an ACK while still needs to send data, B attaches the ACK in the Data frame: Piggybacking
  - Otherwise, B can send an ACK frame separatly
  - After ACK, if B sends some other data, it still put the ACK information in data frame.
- Sliding window is much more efficient than Stop-and-Wait
- More complicated in management.

Exercices

- Given a link with rate R=100Mbps
- We need to send a file over data link layer with file size L=100KB
- Assume that the size of a frame is: 1KB, header size is ignored
- Round trip time (RTT) between 2 ends of the link is 3ms
- An ACK message is sent back from receiver whenever a frame is arrived. Size of ACK message is negligible
- What is the transmission time required if using Stop-and-wait mechanism?
- Transmission time with sliding window if the window size is =7?
- Which size of window allow to obtain the fastest transmission?

Transmission time with Stop-and-wait

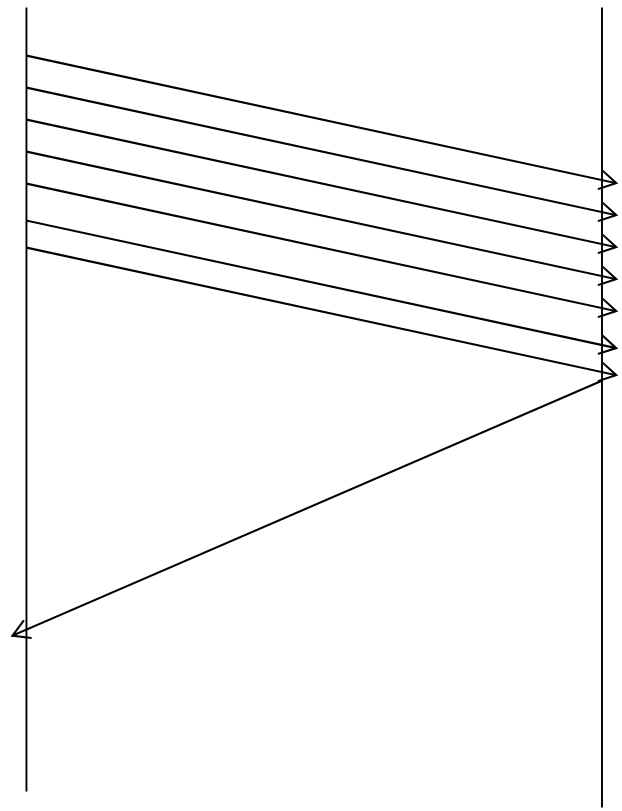


Transmission time with Stop-and-wait

- T total= Nb.frame * (T_transmit + RTT)
- T_transmit (F) = L(Frame)/ R
- Nb. frame = L/L(frame)

- With the given parametters
- Nb. frame =100 KB/1KB =100
- T_transmit (F) = 1KB/100 Mbps =10^3*8/10^8 =8. 10^-5 (s)=0.08 (ms)

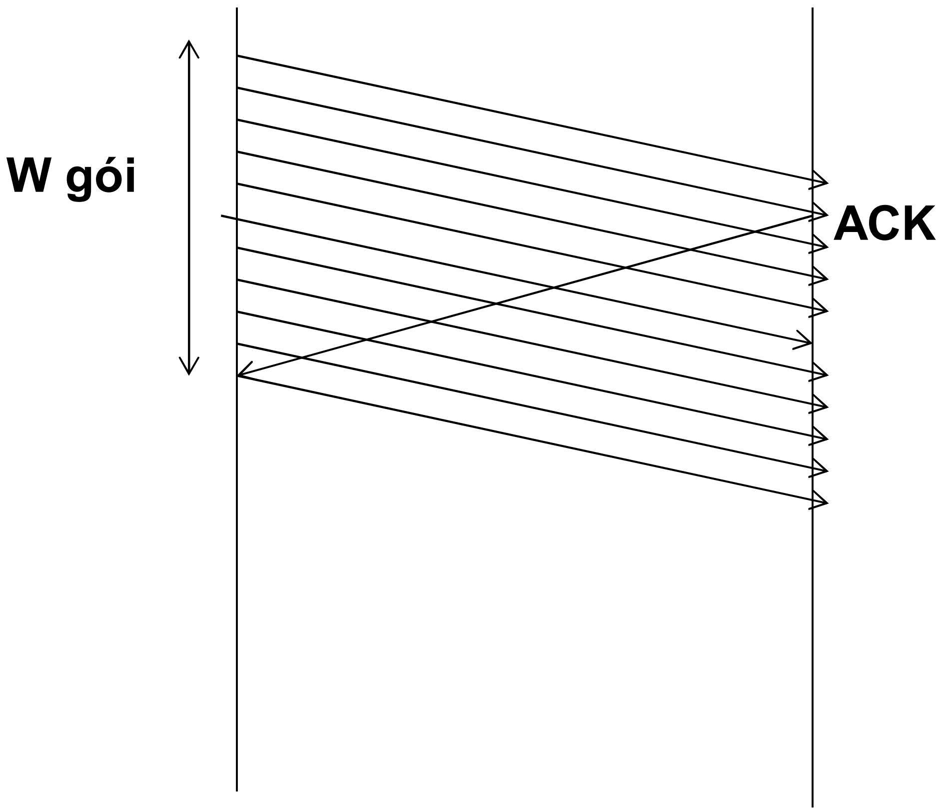
Sliding windows



Trasmission time with window size 7

- T fastest= (T transmit 7 frames+ wait) * Nb. Waiting time.
- 1 waiting= (T transmit 1 frame+ RTT) – T transmit 7 frames
- Nb. Waiting time= Nb frame /7

Fastest transmission time with sliding window



- Fastest transmission time obtained if transmitter receives ACK of the first frame when it finishes transmitting the last frame of the sliding window.
- Window size:W
- T transmit(W fram) >= T transmit first frame + RTT

Fastest transmission time with sliding window

- T transmit (W frame) = W * 1KB/R
- => (W-1)*1KB/R >= RTT
- => W >= RTT*R/1KB +1
-     W>= 3ms * 100 Mbps/ 1KB + 1
-     W>=38.5
- Smallest value of W = 39
- Time to transmit all data L = L/R + RTT =8 ms +3ms =11 ms

Media access control

Connection types

- Point-to-point
  - ADSL
  - Telephone modem
  - Leased Line….
- Broadcast
  - LAN using bus topology
  - Wireless LAN
  - HFC:
  - …
- Broadcast networks need media access control protocol in order to avoid collision when nodes try to send data.

Multiple access links, protocols

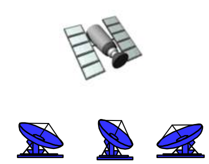
two types of "links":

- point-to-point
  - point-to-point link between Ethernet switch, host
  - PPP for dial-up access
- broadcast (shared wire or medium)
  - old-fashioned Ethernet
  - upstream HFC in cable-based access network
  - 802.11 wireless LAN, 4G/4G. satellite

shared wire (e.g., cabled Ethernet)

shared radio: 4G/5G

shared radio: WiFi



shared radio: 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

An ideal multiple access protocol

given: multiple access channel (MAC) of rate *R* bps

desiderata:

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: taxonomy

three broad classes:

- channel partitioning
  - divide channel into smaller “pieces” (time slots, frequency, code)
  - allocate piece to node for exclusive use
  - e.g. time - TDMA, frequency- FDMA, Code- CDMA
- *random access*
  - channel not divided, allow collisions
  - “recover” from collisions
  - e.g. Pure Aloha, Slotted Aloha, CSMA/CD, CSMA/CA…
- “taking turns”
  - nodes take turns, but nodes with more to send can take longer turns
  - Token Ring, Token Bus

Channel division

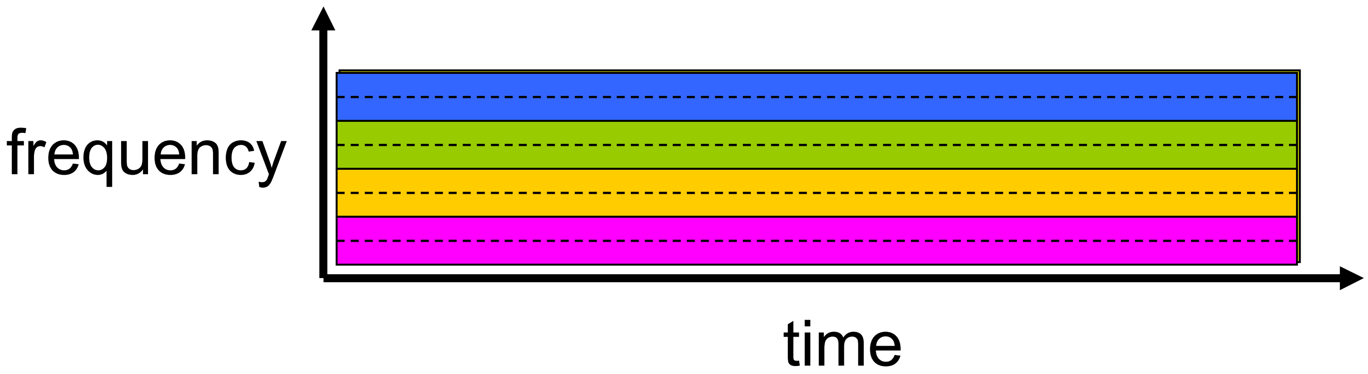
- FDMA: frequency division multiple access
- TDMA: time division multiple access
- CDMA: code division multiple access

TDMA và FDMA

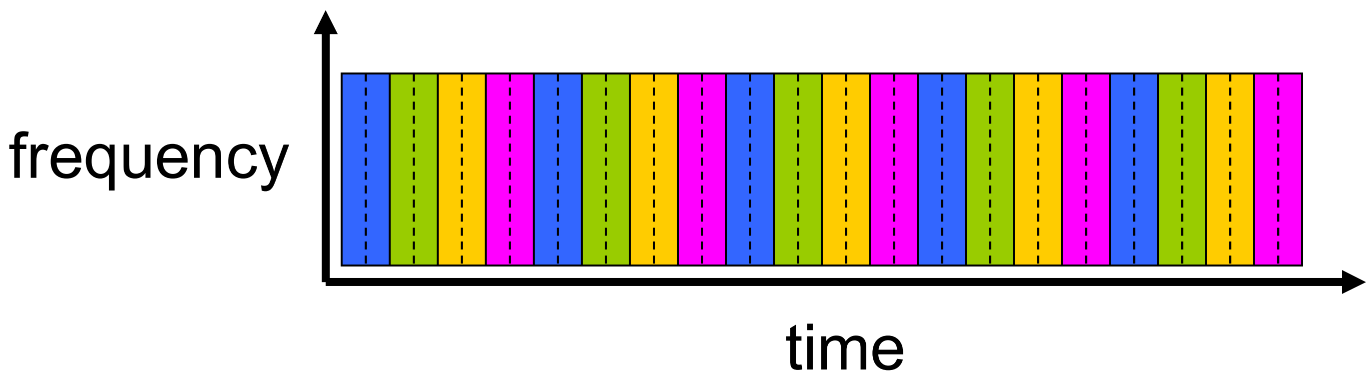
ex

4 stations

FDMA



TDMA:



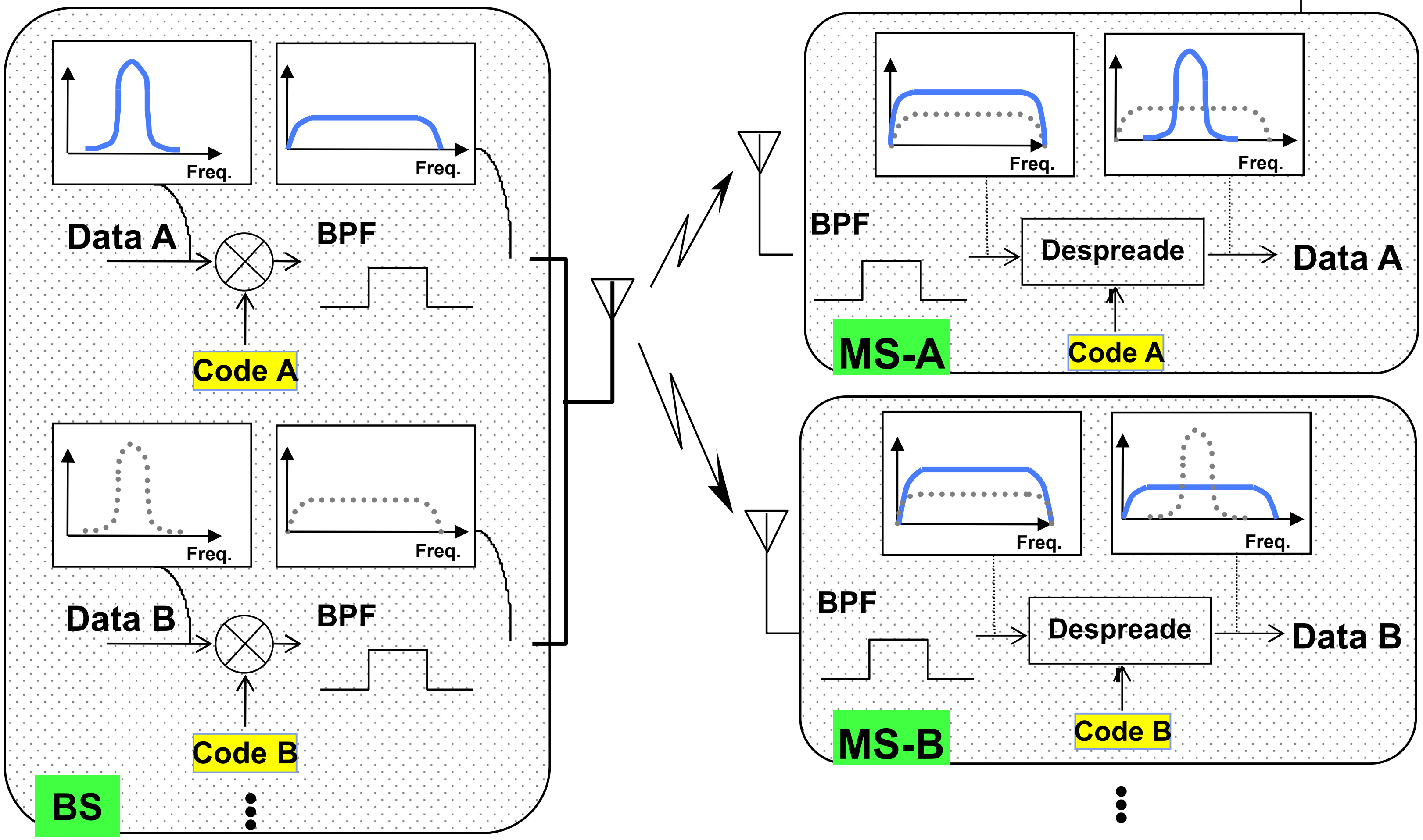
CDMA

- Several senders can share the same frequency on a single physical channel.
- Signals come from different senders are encoded by a different random code
- Encrypted signals are mixed and then transmit on a common frequency.
- The signals are recovered at the receiver by using the same codes as at sender side.
- CDMA use the spread spectrum theory, **CDMA** shows a lot of advantages that other technology cannot achieve.

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

DS-CDMA System Overview (Forward link)

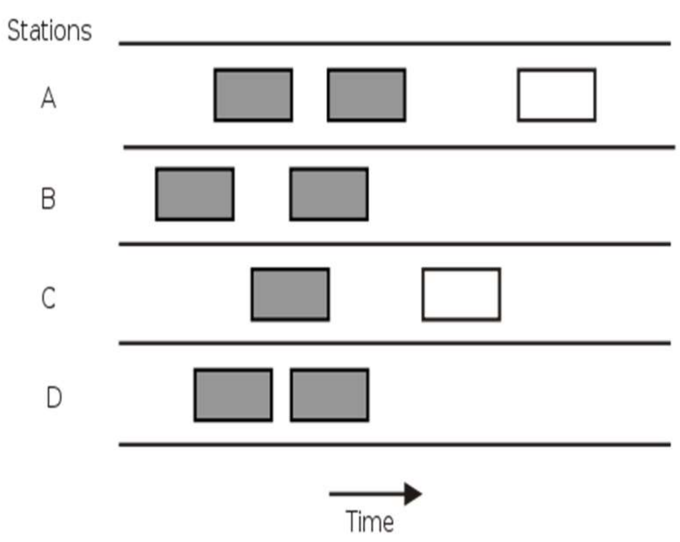
CDMA is a multiple spread spectrum.



Difference between each communication path is only the spreading code

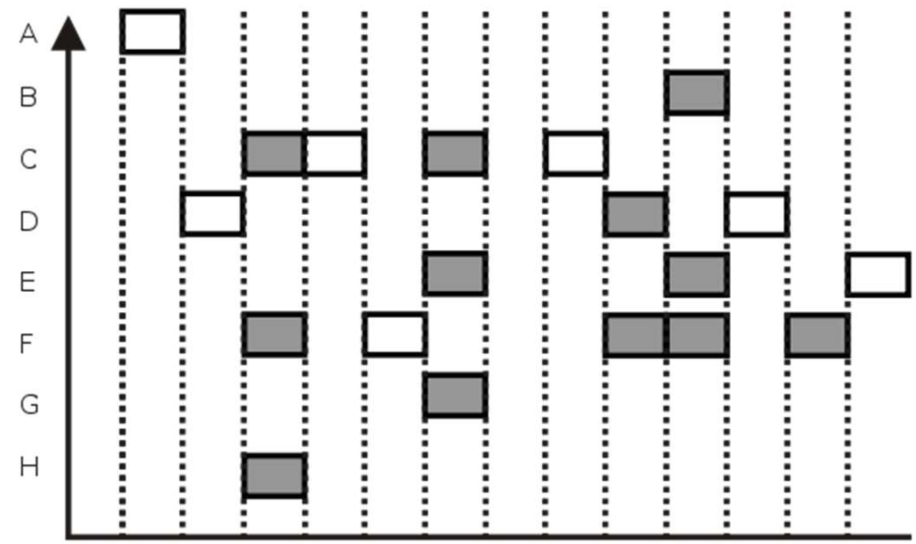
Random access: Pure Aloha

- Aloha is used in mobile network of 1G, 2.5G, 3G using GSM technology .
- Pure Aloha:
  - When one sender has data to send, just sends it
  - If while sending, the senders receive data from other stations → there is collision. All stations need to resend their data.
    - There are possibility to have collision when retransmit.
  - Problem: Sender does not check to see if the chanel is free before sending data
  - Grey package are having overlap in time→ causing collision



Random access: Slotted Aloha

- Times axe is divided into equal slots.
- Each station sends data only at the beginning of a time slot.
- ➔ Collision possibility is reduced
- Still have collision in grey package

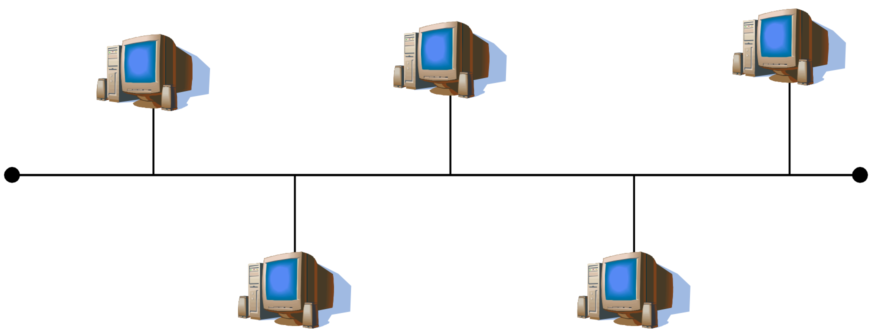


Slotted ALOHA protocol (shaded slots indicate collision)

Random access: CSMA

- CSMA: Carrier Sense Multiple Access
- CSMA idea is similar to what happens in a meeting.
- CSMA:
  - The sender "Listen before talk"
  - If the channel is busy, wait
  - If the chanel is free, transmit

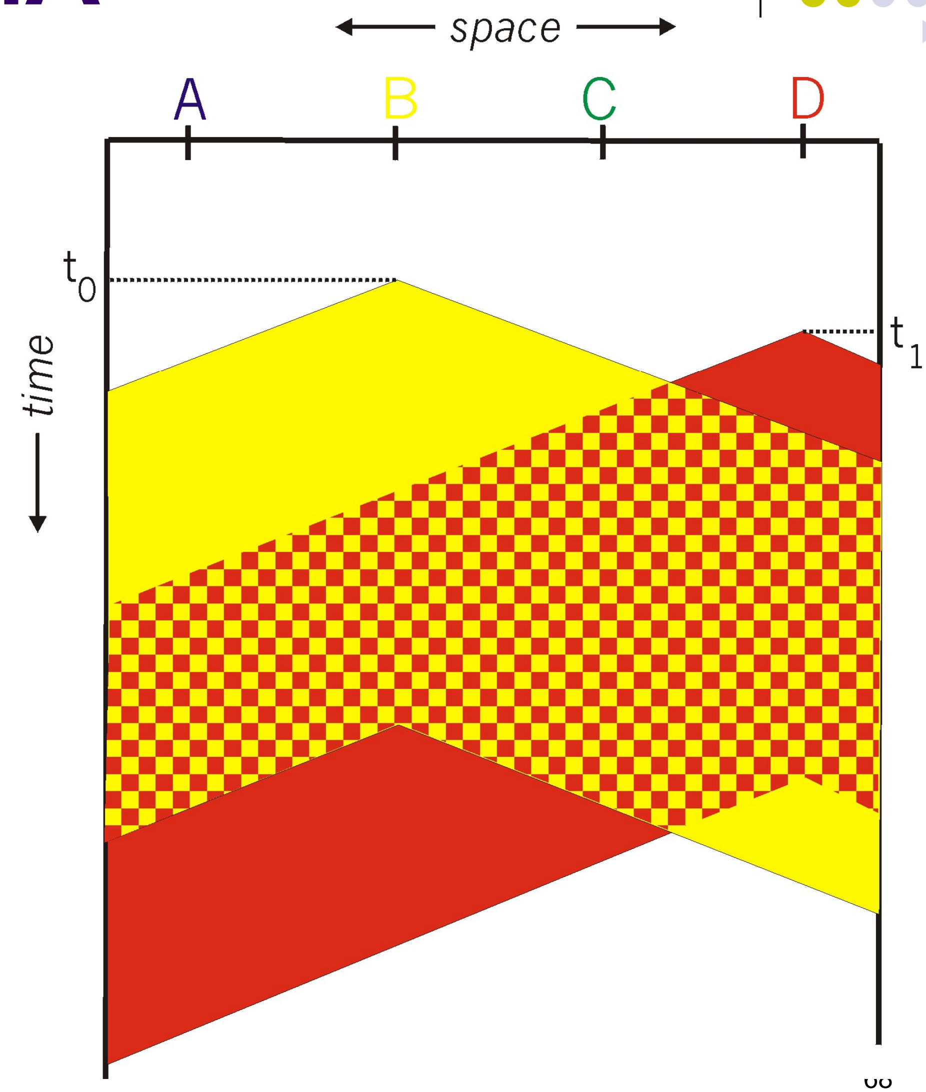
CSMA



- **CSMA:** Sender listens before transmission:
  - If the channel is free, send all the data
  - If the channel is busy, wait.
- Why there are still collision?
  - Due to propagation delay

Collision in CSMA

- Assume that there are 4 nodes in the channel
- The propagation of the signal from one node to the other requires a certain delay.
- Ex:
  - Transmissions from B and D cause collision



CSMA/CA

- CSMA/CA is used WIFI standard IEEE 802.11
- If two stations discover that the channel is busy, and both wait then it is possible that they will try to resend data in the same time.
  - → collision
- Solution CSMA/CA.
  - Each station wait for a random period → reduce the collision possibility

CSMA/CD

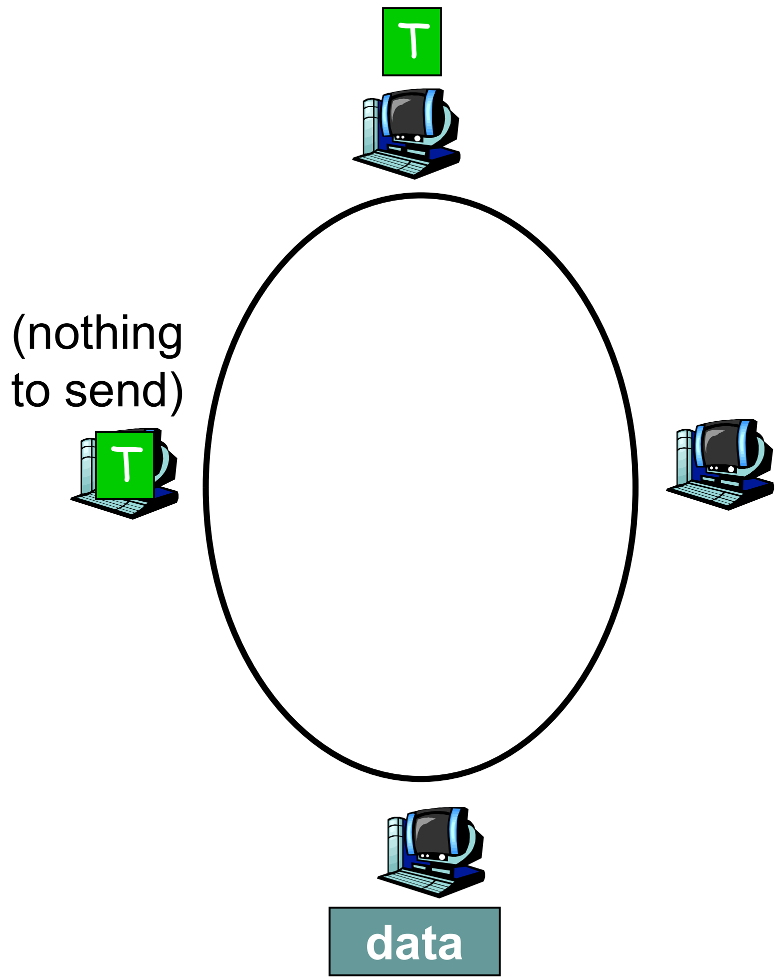
- Used in Ethernet
- CSMA with Collision Detection:
  - “Listen while talk”.
- A sender listen to the channel,
  - If the channel is free then transmit data
    - While a station transmit data, it listens to the channel. If it detects a collision then transmits a short signal warning the collision then stop
    - Do not continue the transmission even in collision as CSMA
  - If the channel is busy, wait then transmit with probability p
- Retransmit after a random waiting time.

Comparison between channel division and random access

- Channel division
  - Efficient, treat stations equally.
  - Waste of resources if one station has much smaller data to send than the others
- Random access
  - When total load is small: Efficient since each station can use the whole chanel
  - When total load is large: Collision possibility increases.
- Token control: compromise between the two above methods.

Token Ring

- A “tocken” is passed from one node to the other in a ring topo
- Only the token holder can transmit data
- After finishing sending data, the token need to be passed to next nodes.
- Some problem
  - Time consuming in passing token
  - Loss of token due to some reasons



Summary on Media access control mechanisms

- Channel division
- Random access
- Token
- What do you thinks about their advantages and weaknesses