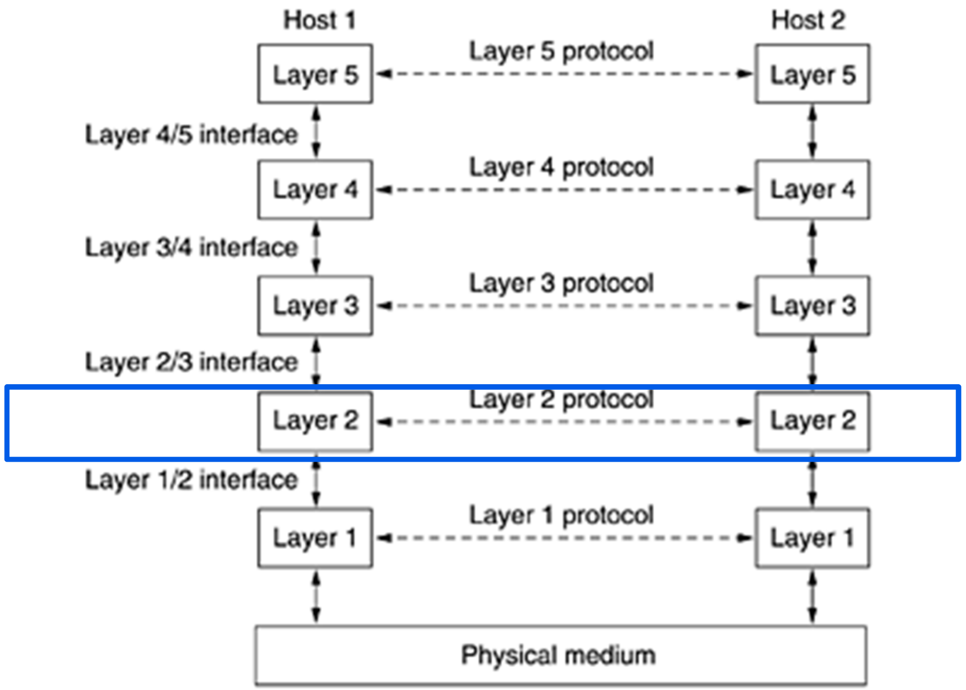
MIEEC

Computer Networks

Lecture note 4

The data link layer

Data link


Host 1
Host 2
Layer 5
Layer 5 protocol
Layer 5
Layer 4/5 interface
Layer 4
Layer 4 protocol
Layer 4
Layer 3/4 interface
Layer 3
Layer 3 protocol
Layer 3
Layer 2/3 interface
Layer 2
Layer 2 protocol
Layer 2
Layer 1/2 interface
Layer 1
Layer 1 protocol
Layer 1
Physical medium

Functions and services

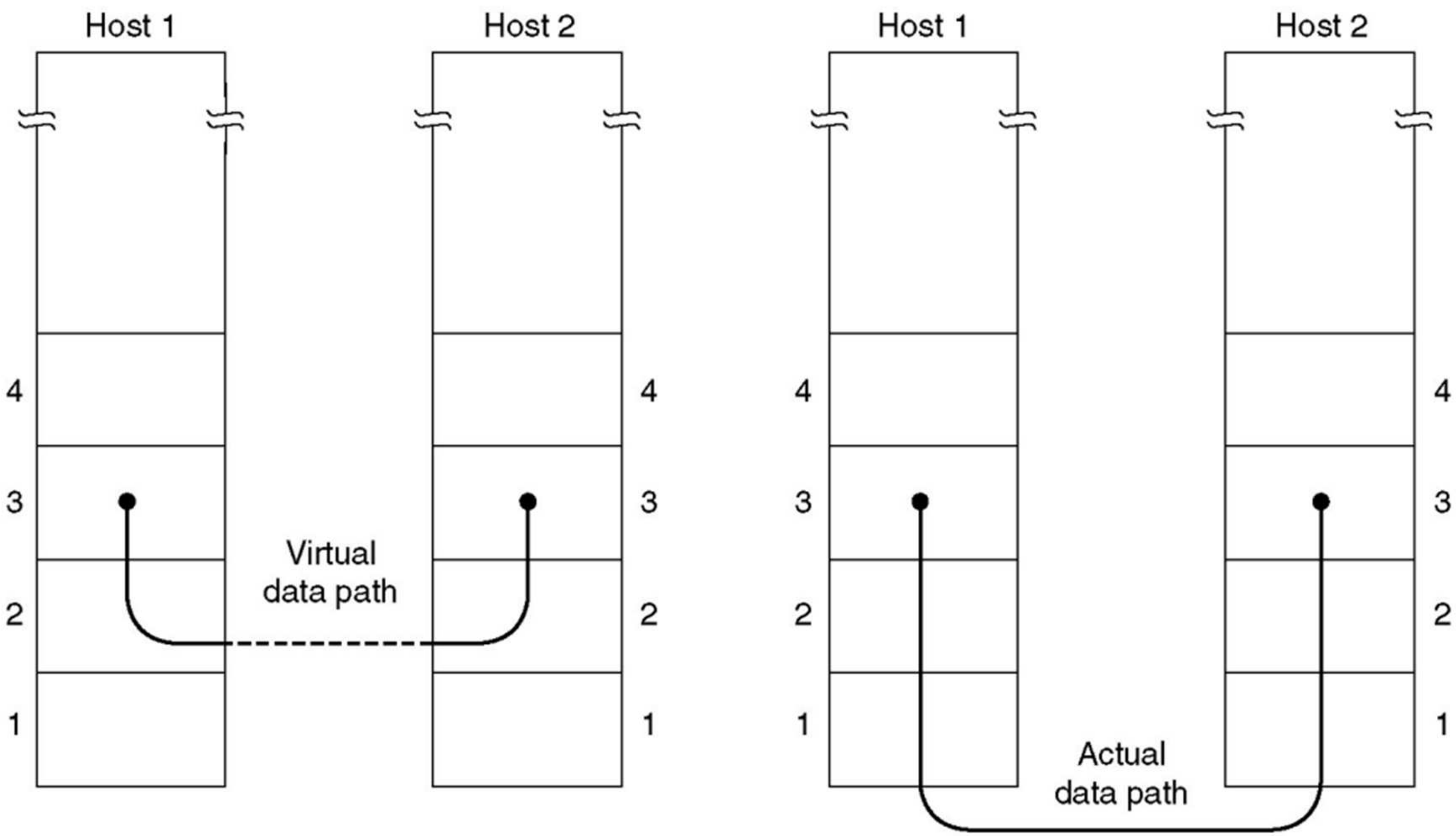
Main functions:

- Service interface to upper layer (network)
- Dealing with transmission errors
- Regulating data flow

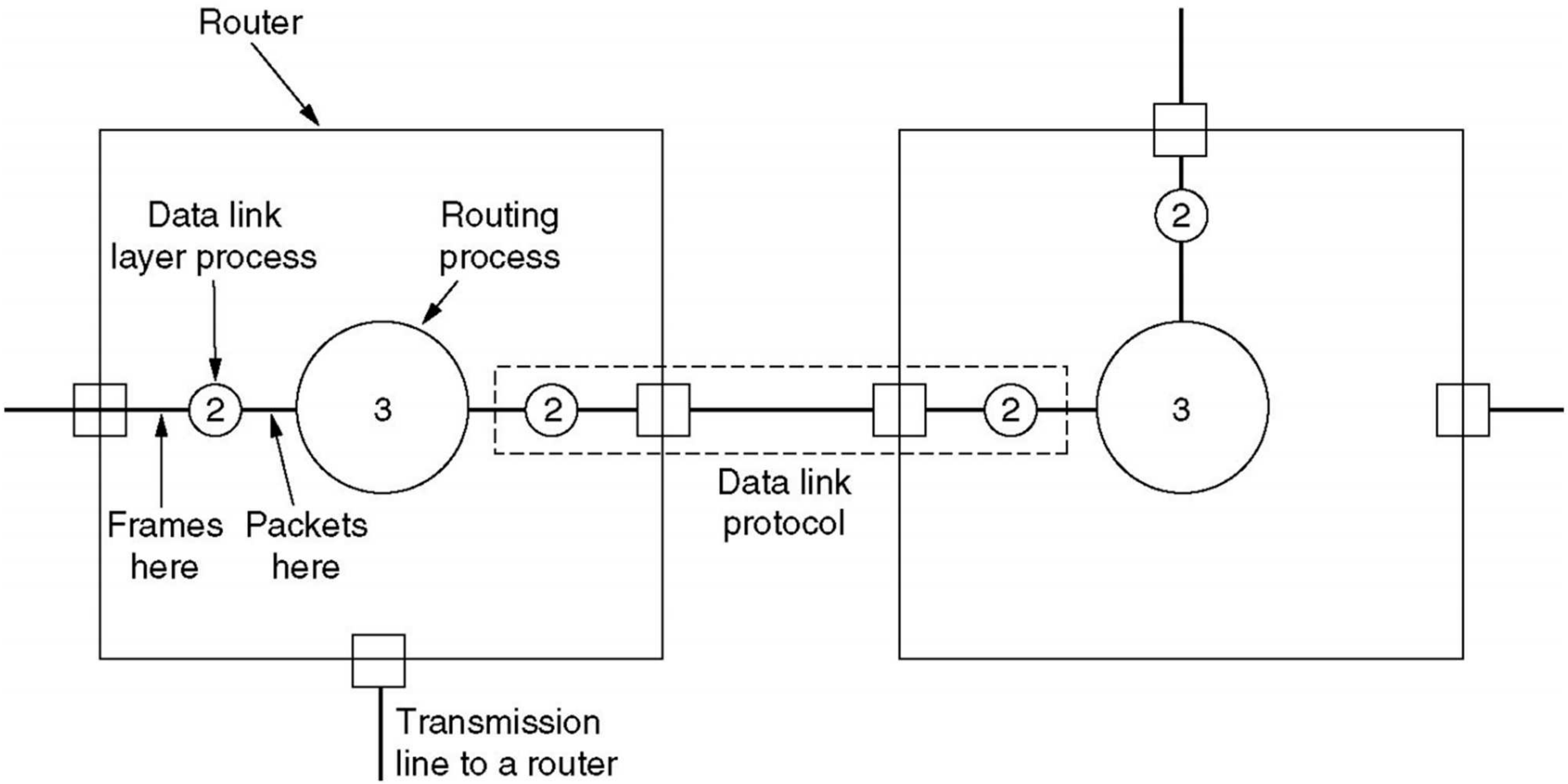
Services provided:

- Unacknowledged connectionless service
- Acknowledged connectionless service
- Acknowledged connection-oriented service

Service to the network layer



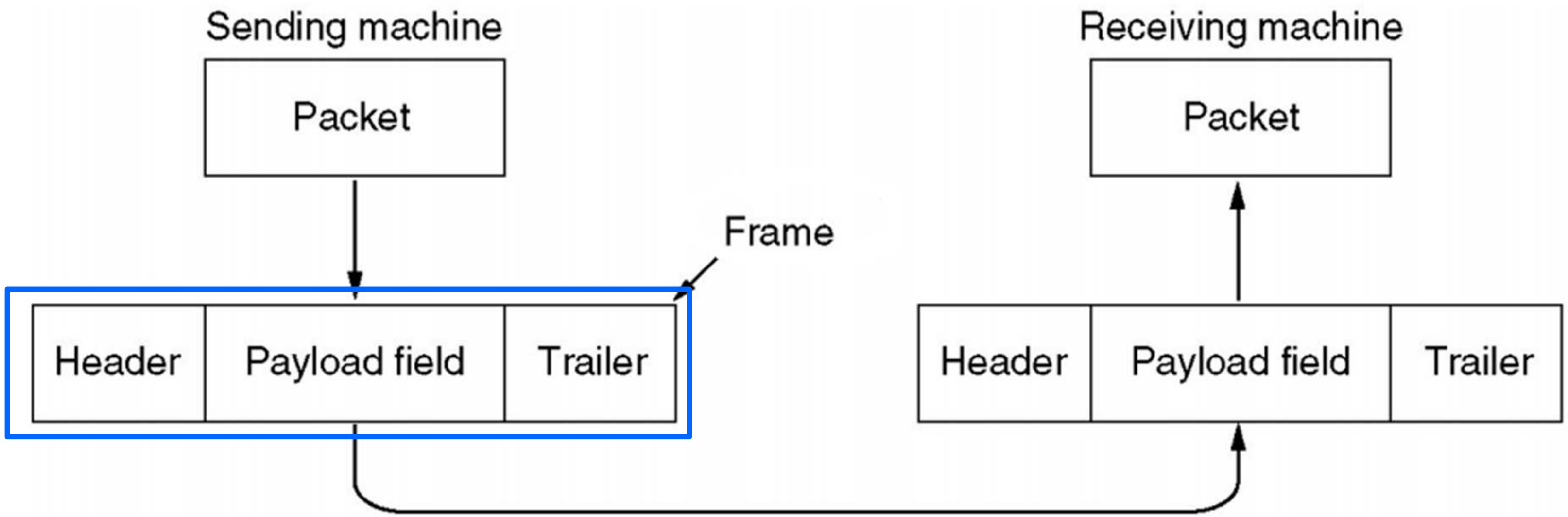
Data link: the big picture



Frames and packets

Layer 3

Layer 2



Framing

TO THINK

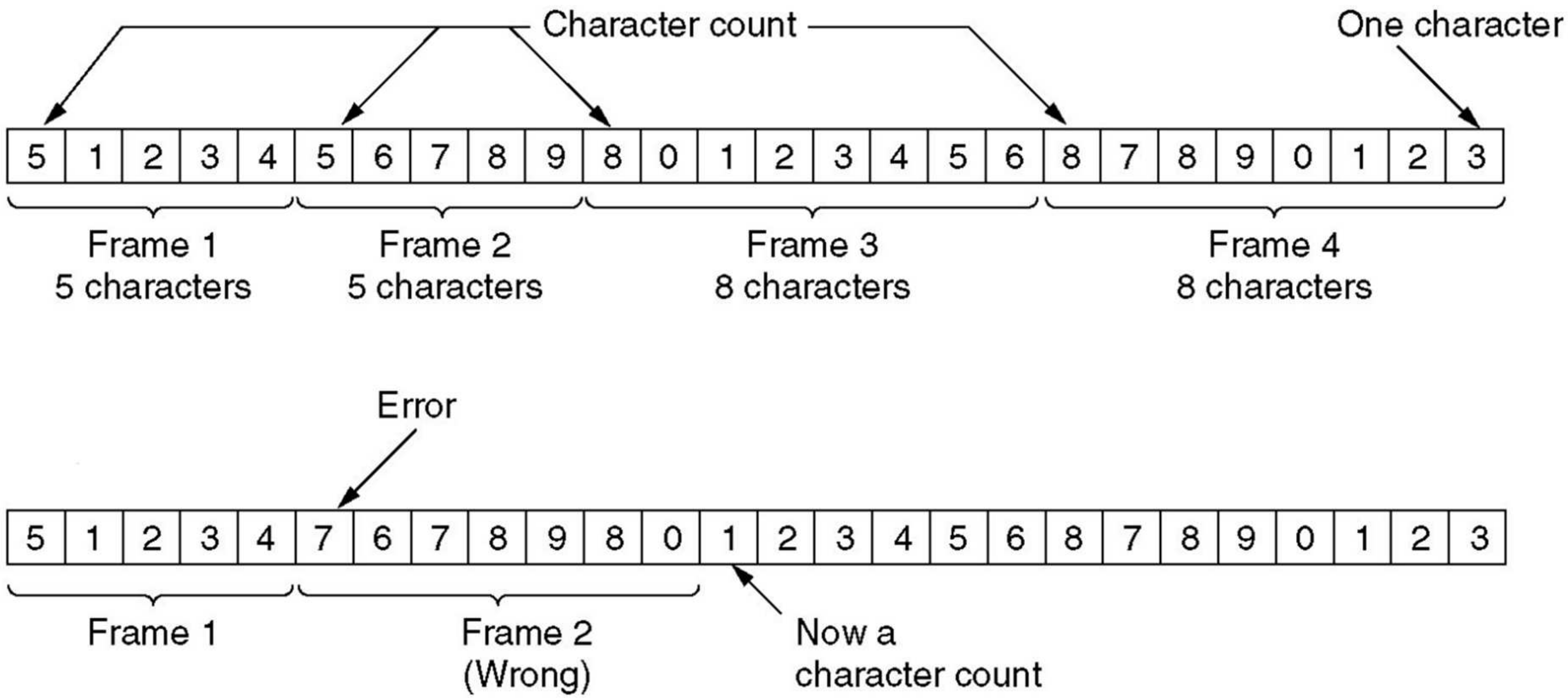
The data link gets a stream of 1s and 0s from the physical layer.

[sender] => ..00100100001011010111100101001011101000 => [receiver]

- Where does a frame start/stop?
  - Need to split a bit stream into frames

Character counting

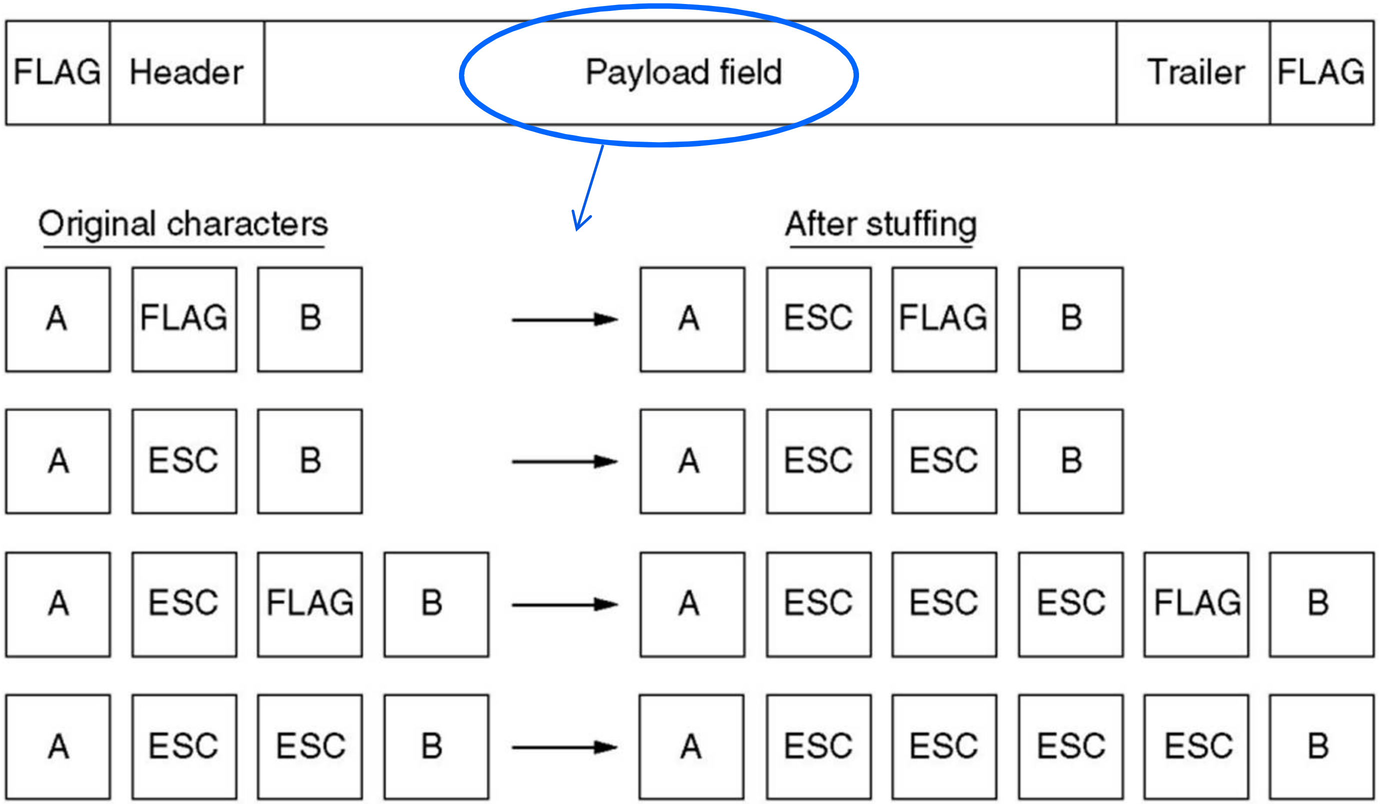
- Send the length of each frame



Start/Stop Flags

- Unique sequence as flag
  - What's the problem with flagging sequences?

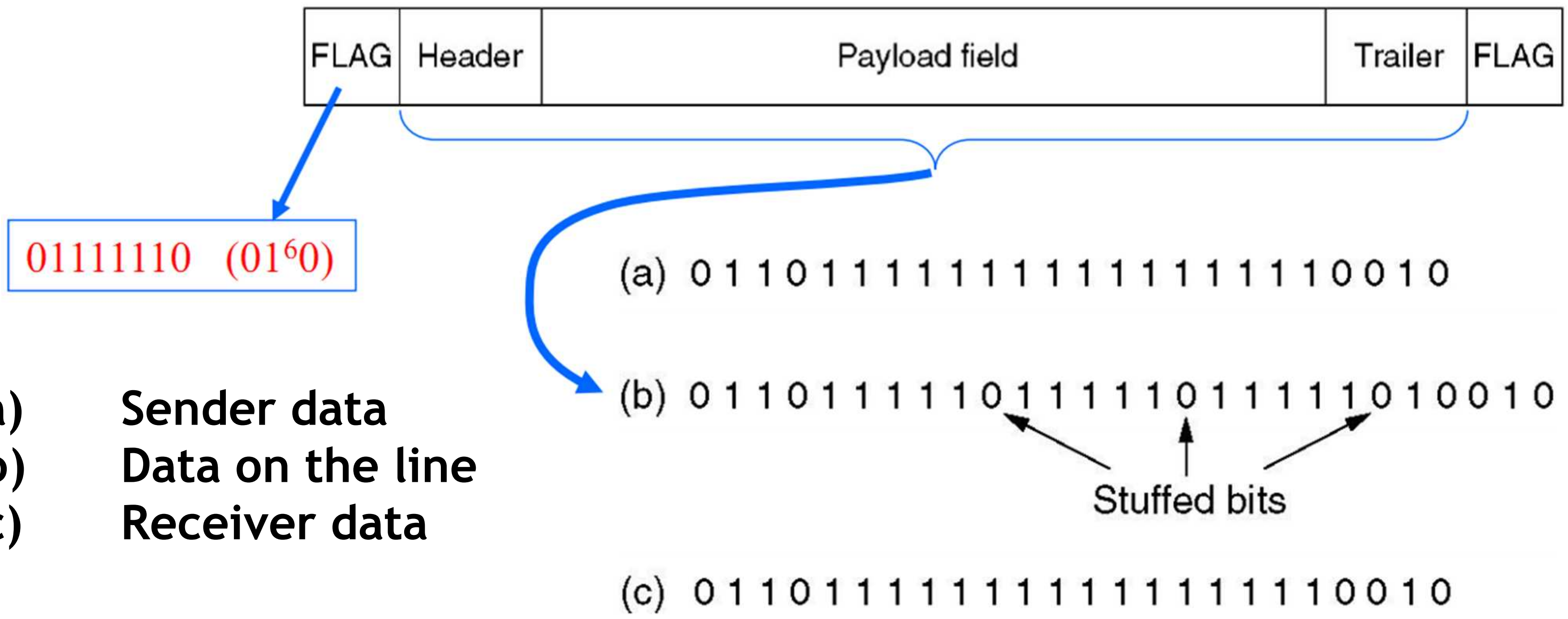
Flagging and stuffing: Byte stream



Flagging and stuffing

- How do you know where a byte starts?

Flagging and stuffing: Bit stream



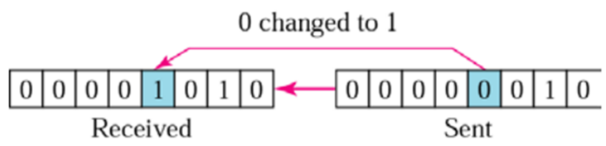
a) Sender data
b) Data on the line
c) Receiver data

a) => b) b) => c)

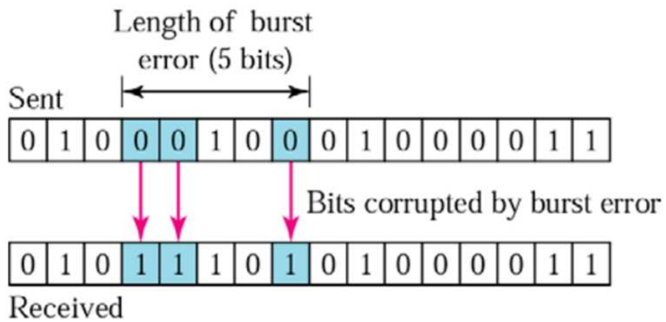
$1^5 \Rightarrow 1^5 0$ $1^5 0 \Rightarrow 1^5$

Error detection

Types of errors



Simple (individual) errors
- Random
- Independent from errors in other (previous) bits



Errors in bursts
- Random
- Not independent
- Burst: period with higher probability of error
- Burst length: [# last bit] – [# first bit] + 1

TO THINK

What's the math expression for:

- Probability frame has no errors
- Probability frame has errors

Assume

- p – bit error probability (Bit Error Ratio, BER)
- n – frame length
- Independent errors

TO THINK

Assume
- p – bit error probability, BER
- n – frame length
- Independent errors

Student A explains to student B why

$$P(frame\ has\ no\ errors) = (1-p)^n = (1-BER)^n$$

Student B explains to student A why

$$P(frame\ has\ errors) = 1-(1-p)^n = FER$$
$$= 1-(1-BER)^n$$

Error counting

Assume
- p - bit error probability, BER
- n - frame length
- Independent errors

- How many errors?
  - $P(frame\ has\ \boldsymbol{one\ or\ more}\ errors) = 1-(1-p)^n = FER$
- P(frame has i errors)
  - $i = 0 \Rightarrow P = (1-p)^n$
  - $i = 1 \Rightarrow P = \binom{n}{1} p(1-p)^{n-1}$
  - $i \Rightarrow P = \binom{n}{i} p^i (1-p)^{n-i}$

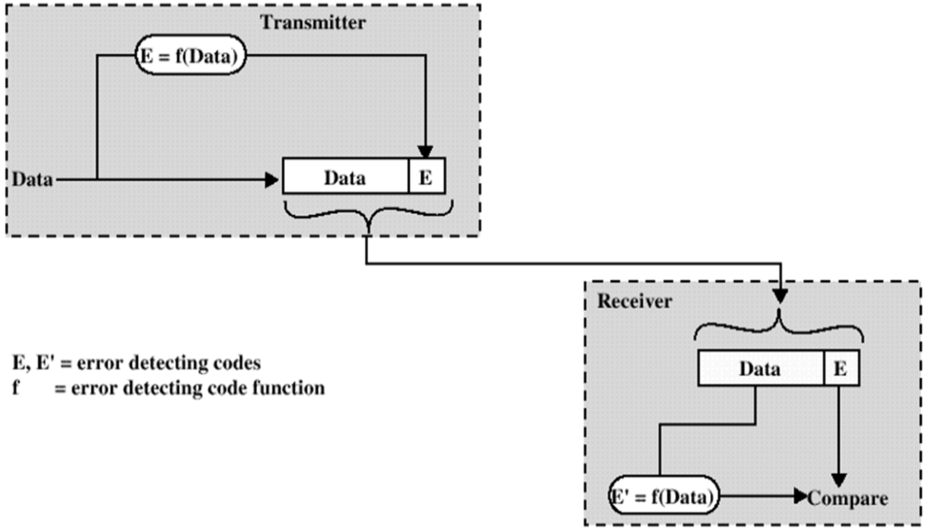
Good wired channel
- p = 10^{-7}
- n - 10^{-4} (Ethernet frame)
- $p(FER) = 1-(1-10^{-7})^{10^4} \sim 10^{-3}$

Wireless channel
- p = 10^{-3}
- n - 10^{-4} (Ethernet frame)
- $p(FER) = 1-(1-10^{-3})^{10^4} \sim 1$

Error detection techniques - effectiveness

- Minimum number **d** of bit errors detected in a block of n bits
  - If fewer than d errors, errors are detected
  - Maximum distance of code
- Maximum burst length of errors detected **B**

Error detection techniques - redundancy



- k => k + r
  - k data bits
  - r redundancy bits
- Check for errors
  - Using r and k
  - If error, ask to retransmit
- Techniques
  - Parity check
  - Cyclic Redundancy Check, CRC

Simple parity check

- Add 1 parity bit every k data bits
  - k+1 bits on the line
- Set parity bit to 1 or 0 to satisfy:
  - Total number of bits with value “1” is even
    - even parity
  - Total number of bits with value “1” is odd
    - odd parity
- 111010 101011 001010
  - Odd or even?

Simple parity check

- Allows detection of
  - Simple errors
  - Any odd number of errors in a k+1 block
- What goes by undetected?
  - Even number of errors
  - $P(undetected) = \sum_{even\ i} \binom{n}{i} p^i (1-p)^{n-1}$
- Used in character-oriented protocols

Assume
- p – bit error probability, BER
- n – frame length
- Independent errors

Bi-dimensional parity

- Blocks represented as rows in a matrix
- One parity bit per
  - Row
  - Column
- Can detect the exact location of single error or multiple even errors in single row/column
- Errors in rectangle configuration are undetectable
  - Minimum code distance d=4

1	0	0	1	0	1	0	1	
0	1	1	1	0	1	0	0	Horizontal
1	1	1	0	0	0	1	0	checks
1	0	0	0	1	1	1	0	
0	0	1	1	0	0	1	1	
1	0	1	1	1	1	1	0	

Vertical checks

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

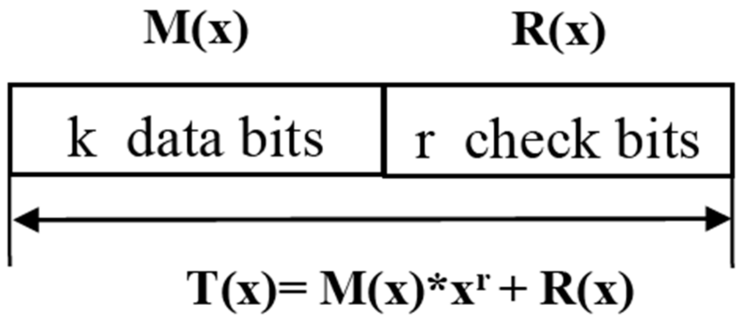
Internet (IP) Checksum

- Note: not layer 2!
- IP Checksum
  - Easily implementable in software
  - 1’s complement sum of 16 bit words
  - d=2
- 1’s complement sum
  - Mod-2 addition with carry-out
  - Carry-out bit is added to least significant bit
  - Take 1’s complement

	1010011
	0110110
carry-out	1 0001001
Carry wrap-around	0000001
	0001010
One's complement =	**1110101**

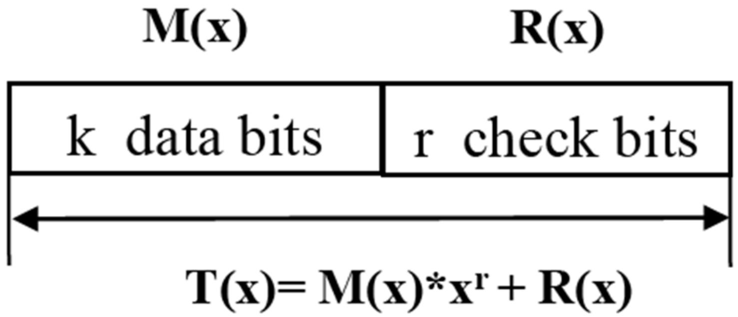
Cyclic redundancy check CRC

- Represent bit string as polynomial
  - $1101001 \Rightarrow x^6 + x^5 + x^3 + 1$
- T(x) : transmitted bits
- M(x) : message bits
- R(x) : check bits



- Goal: take T(x) and assess presence of errors
- G(x) : generator polynomial

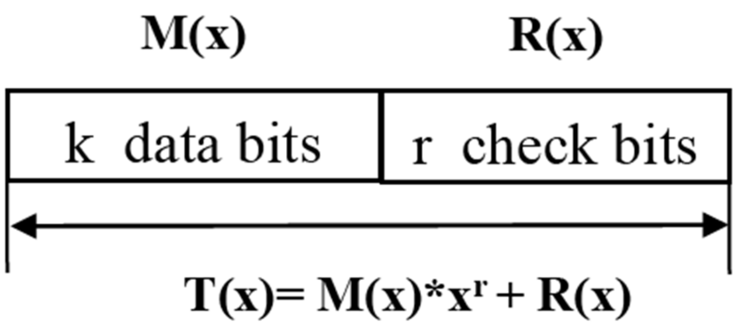
CRC - Generator Polynomial

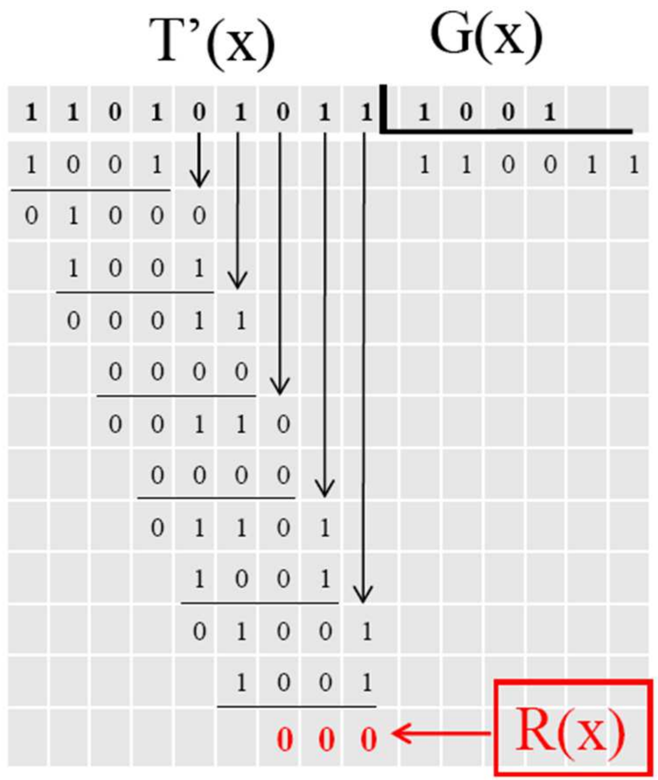


- G(x)
  - r+1 bits long
- Find R(x) such that
  - $T(x) = M(x)\,x^r + R(x) = A\,G(x)$
- This means that
  - $M(x)\,x^r = A\,G(x) + R(x)$ (mod-2 arithmetic)
  - $\mathrm{R}(x) = remainder\ of\ {}^{M(x)\,x^r}\!/_{AG(x)}$

CRC - checking at the receiver

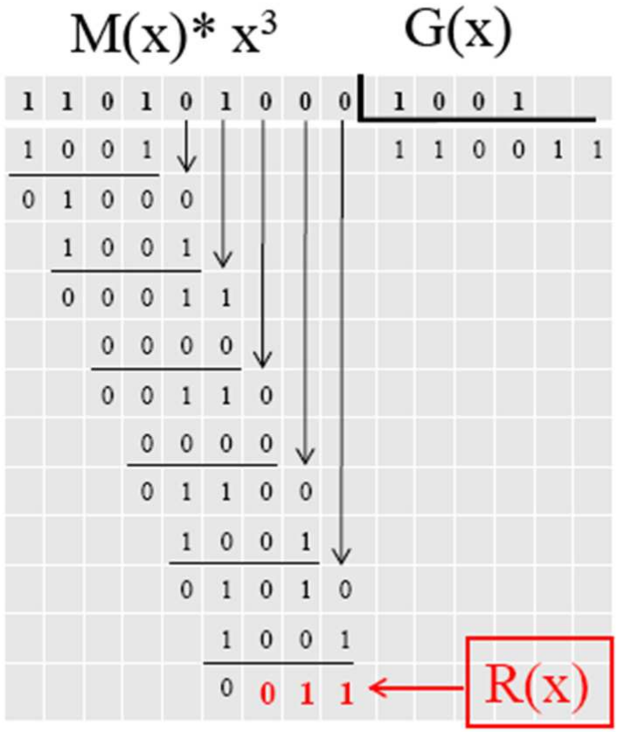
- Divide T(x) by G(x)
  - If remainder is zero => no errors
  - Errors if remainder non zero





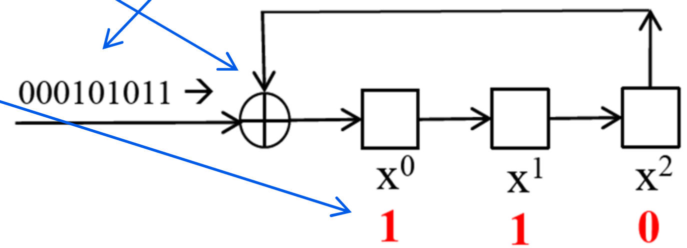
CRC - Generating R(x)

- $\mathrm{R}(x) = remainder\ of\ {}^{M(x)\,x^r}\!/_{AG(x)}$
- Example
  - $\mathrm{r} = 3, \mathrm{x}^{\mathrm{r}} = \mathrm{x}^3; \mathrm{G}(x) = x^3 + 1\ (\mathbf{1001})$
  - $\mathrm{M(x)} = \mathrm{x}^5 + \mathrm{x}^4 + \mathrm{x}^2 + 1(\mathbf{110101})$
  - $\mathrm{M(x)x}^3 = \mathrm{x}^8 + \mathrm{x}^7 + \mathrm{x}^5 + \mathrm{x}^3\ (\mathbf{110101})$
  - $R(\mathrm{x}) = \mathrm{x} + 1$
  - $\mathrm{T(x)} = \mathrm{M(x)x}^3 + \mathrm{x} + 1$
    $= \mathrm{x}^8 + \mathrm{x}^7 + \mathrm{x}^5 + \mathrm{x}^3 + \mathrm{x} + 1$
    $= 1101011$

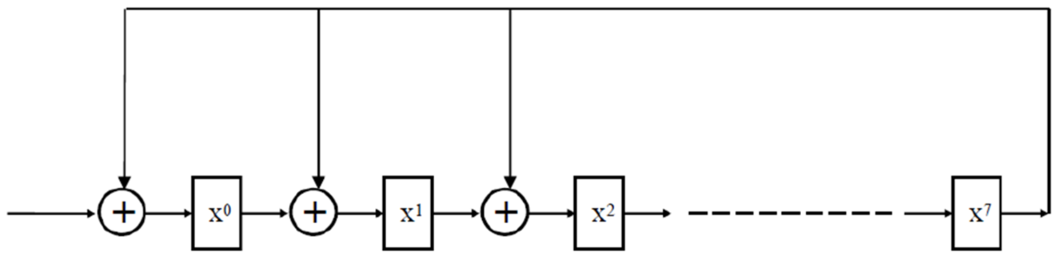


CRC - Generating R(x): hardware, shift registers

- $G(x) = x^3 + 1$
- $M(x)\,x^3 = x^8 + x^7 + x^5 + x^3$ (110101000)
- $R(x) = x + 1$ (011)



- $G(x) = x^8 + x^2 + x + 1$



CRC - Performance

- For r check bits CRC can detect:
  - All patterns of 1,2, or 3 errors (d > 3)
  - All bursts of errors of r or fewer bits
  - All errors consisting of an odd number of inverted bits
- ITU-16: r=16, G(x)=
  - $x_{16}+ x_{12}+ x_5+1$ (10001000000100001)
- ITU-32: r=32, G(x)= $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x^{1}+1$

Automatic Repeat Request (ARQ)

Motivation

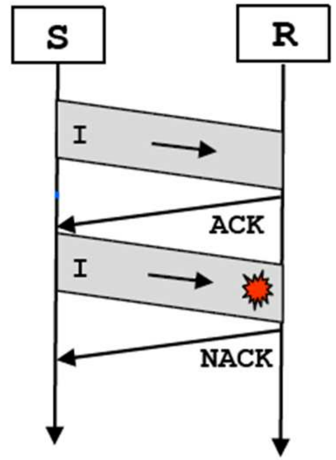
- When errors are detected in a frame
- Receiver asks for retransmission

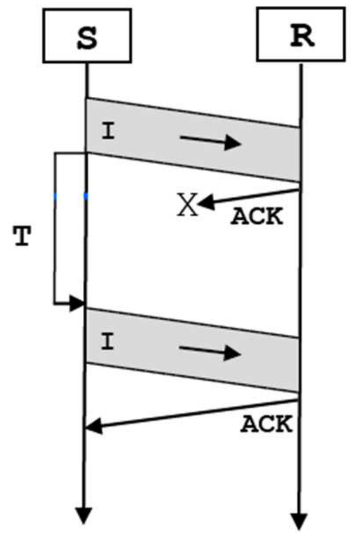
- ARQ
  - Automatic retransmission of:
    - Packets with errors
    - Missing packets

- Stop and wait
- Go back N
- Selective repeat

Stop and Wait ARQ

- Sender: sends frame (DATA)
  - Waits for acknowledgement
- Receiver: receives frame
  - If no errors in frame: send ACK
  - If errors: send NACK
- Sender:
  - receives ACK => transmit new frame
  - receives NACK => retransmit frame
- If frame, ACK, NACK is lost?
  - Timeout T

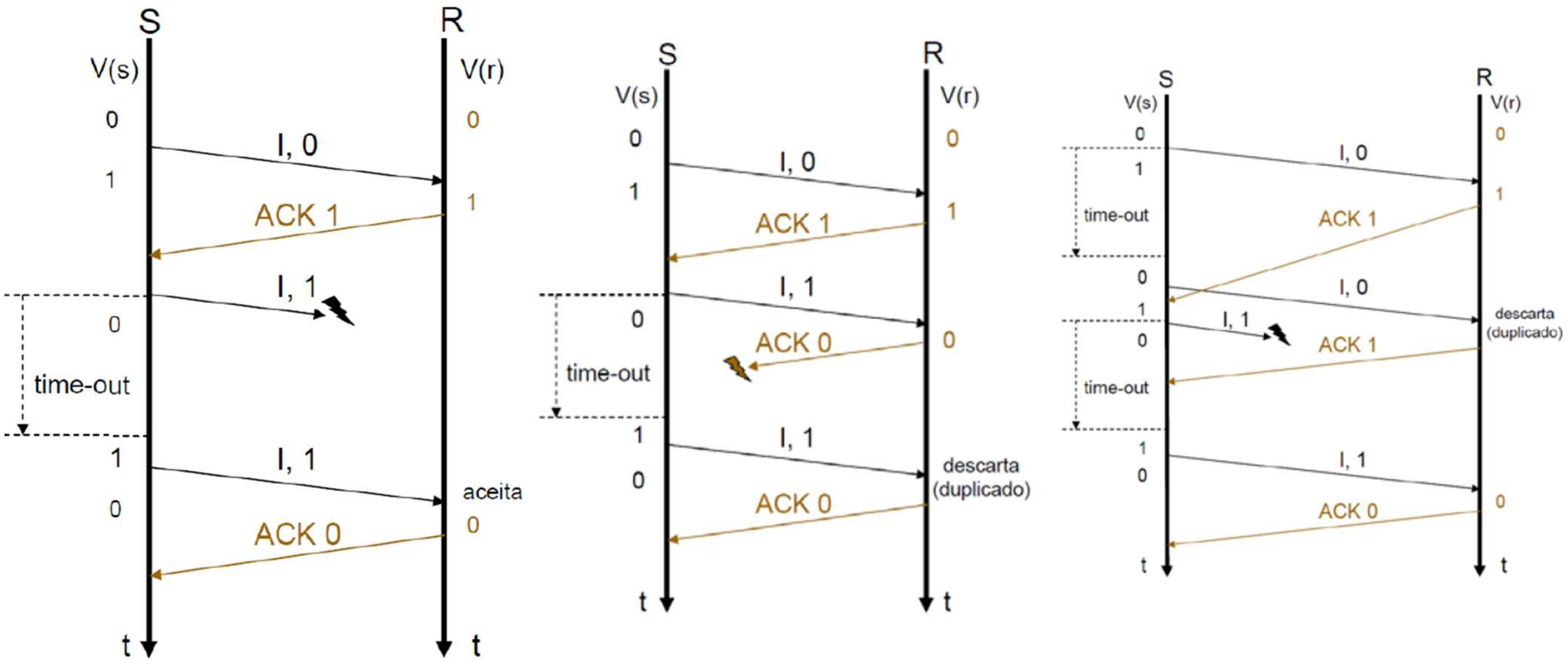




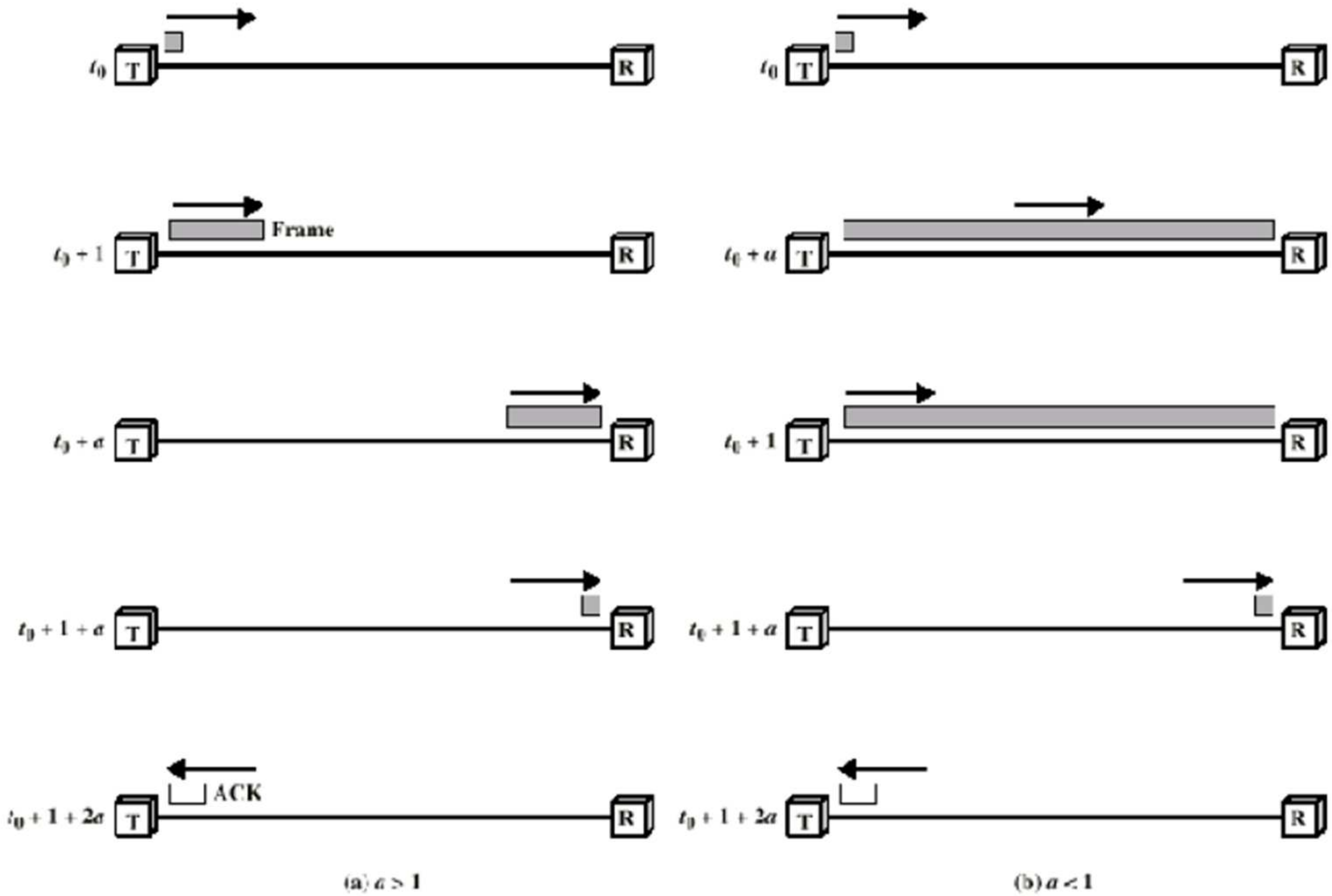
Sequence numbers

- Problems
  - Receiver: is this a new frame or retransmission?
  - Sender: which frame is ACKed / NACKed?
- Solution: sequence numbers
  - Frames numbered
  - ACK numbered
    - ACK(i) means receiver is waiting for frame i
  - No NACK required
  - Module 2 sequence numbers

Examples


S
R
V(s)
V(r)
0
0
I, 0
1
ACK 1
1
I, 1
0
time-out
1
I, 1
aceita
0
ACK 0
0
t
t
S
R
V(s)
V(r)
0
0
I, 0
1
ACK 1
1
I, 1
0
ACK 0
0
time-out
1
I, 1
0
descarta
(duplicado)
ACK 0
t
t
S
R
V(s)
V(r)
0
0
I, 0
1
time-out
ACK 1
1
0
I, 0
1
I, 1
descarta
(duplicado)
0
ACK 1
time-out
1
I, 1
0
ACK 0
0
t
t

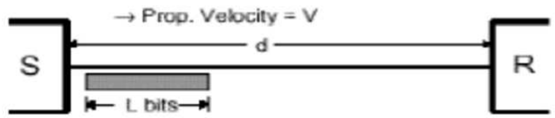
Efficiency



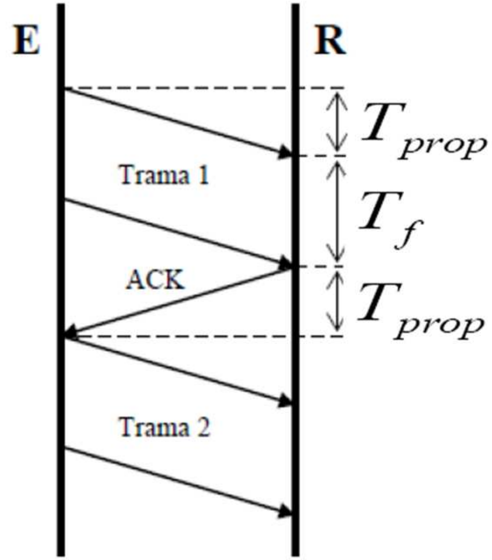
Stop-and-Wait Link Utilization (transmission time = 1; propagation time = a)

Efficiency

- Propagation and transmission
  - $T_{prop} = \tau = \frac{d}{V}$
  - $T_f = \frac{L}{R}$
- Efficiency
  - $a = \frac{T_{prop}}{T_f}$
  - $S = \frac{T_f}{T_{prop}+T_f+T_{prop}} = \frac{1}{1+2a}$



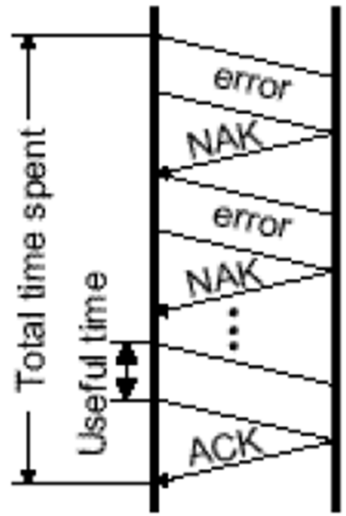
➔ R – *data rate* (bit/s)



Efficiency examples

- WAN ATM
  - $L = 424$ bit; $R = 155.52$ Mbit/s; $T_f = 2.7\ \mu s$
  - $d = 1000$ km; fiber optics $5\ \mu s/km \Rightarrow T_{prop}=5\ \mu s$
  - $a = 1852$, $S=0.0003$
- LAN
  - $L = 1000$ bit, $R=10$Mbit/s, $T_f = 100\ \mu s$
  - $d = 0.1\sim10$km; coax $4\ \mu s/km \Rightarrow T_{prop}=0.4\sim40\ \mu s$
  - $a = 0.004\sim0.4$; $S=0.55\sim0.99$

Efficiency with errors



- P_e = FER
- P(A=k)
  - Probability of k attempts required for successful transmission
  - $P(A=k) = p_e^{k-1}(1-p_e)$
- E(A)
  - Expected number of attempts
  - $E(A) = \sum_{k=1}^{+\infty} kP(A=k) = \frac{1}{1-p_e}$
- Efficiency
  - $S = \frac{T_f}{E(A)(T_f+2T_{prop})} = \frac{1}{E(A)(1+2a)} = \frac{1-p_e}{1+2a}$

TO THINK

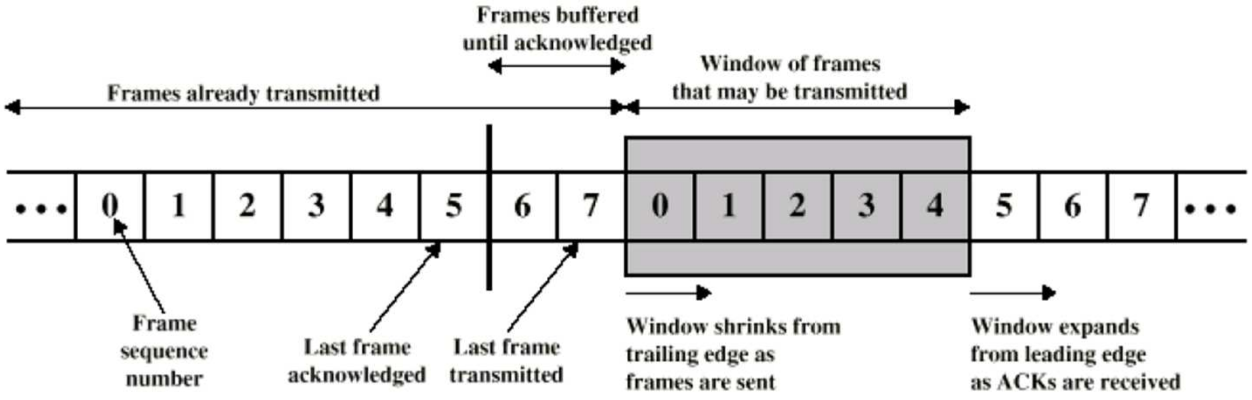
Assume sender and receiver are far apart

- Why is efficiency smaller?
- How can the efficiency be improved?

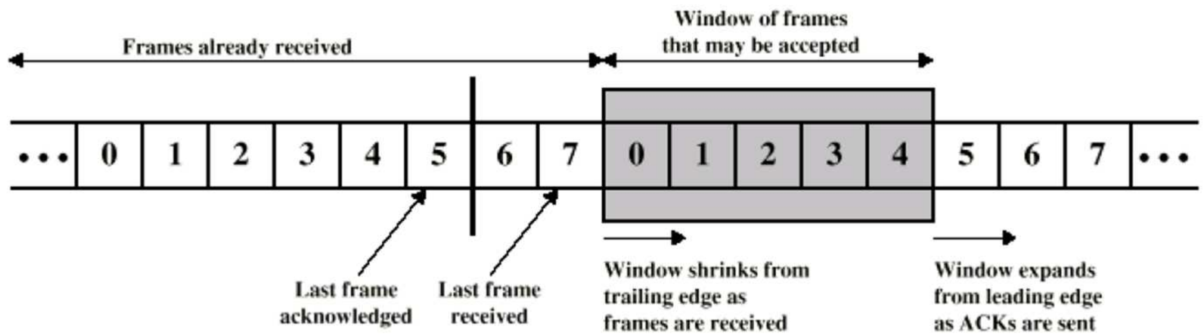
Go back N ARQ (AKA Sliding window)

- Stop and wait inefficient when
  - $T_{prop} > T_f$ (a>1)
  - Single frame per RTT
- Go back N
  - new packets sent before confirmation of previous packets
  - sliding window
    - range of packets that can be sent
    - window slides with acknowledgements

Sliding window model

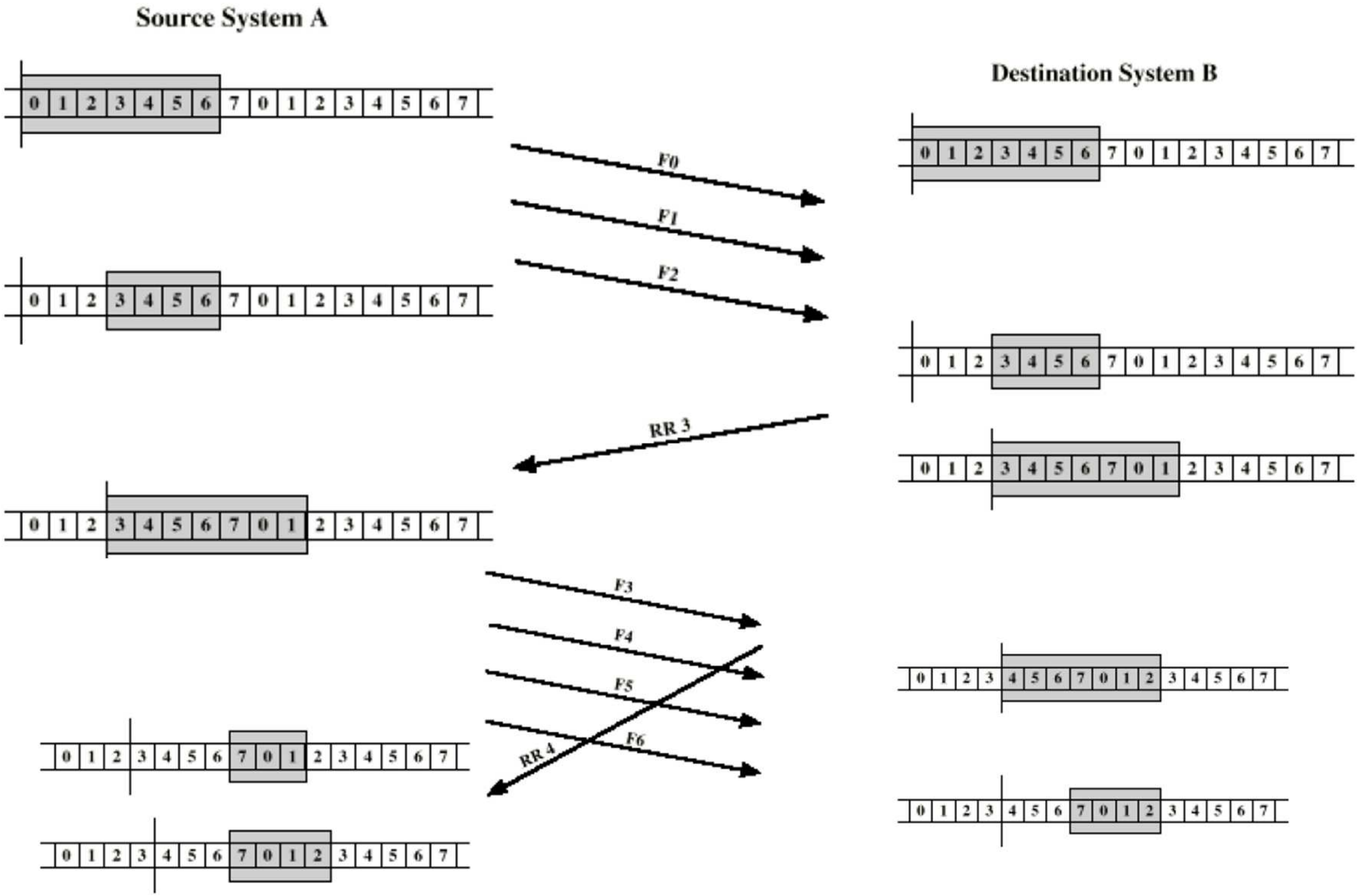


(a) Sender's perspective



(b) Receiver's perspective

Example



Source System A
Destination System B
F0
F1
F2
RR 3
F3
F4
F5
F6
RR 4

Go back N basic behavior

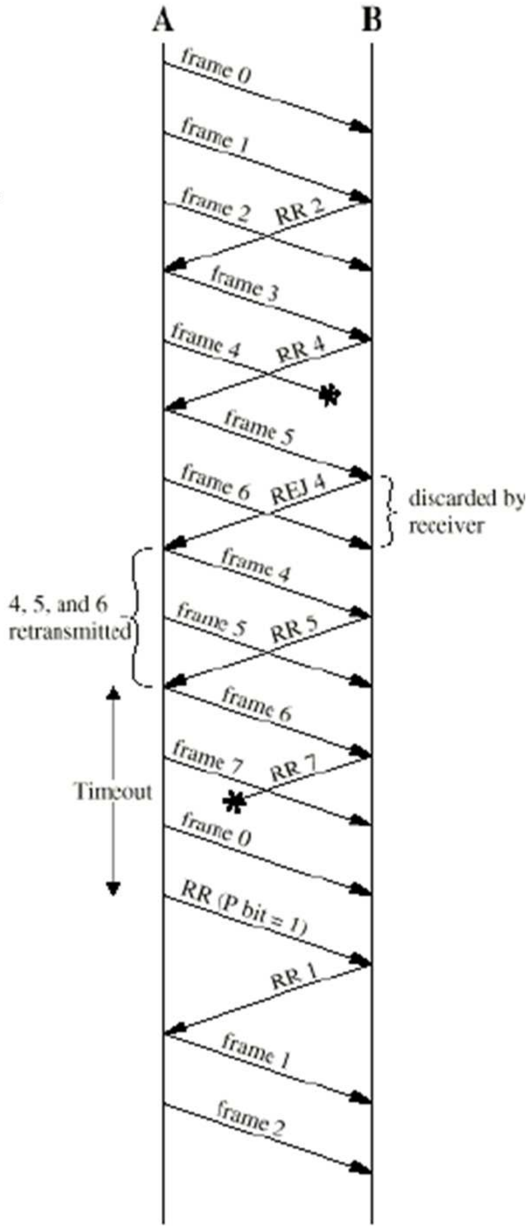
- Sender
  - May send up to W frames without RR (ack)
  - Frames I(k) are numbered sequentially
  - Cannot send I(k+W) until reception of RR(k)
- Receiver
  - Does not accept frames out of sequence
  - Sends RR(k) indicating
    - All packets up to k-1 have been received
    - The next expected frame is k

Go back N ARQ

- Sequence numbers
  - Module M (0..M-1)
  - n bits to represent in header
  - 2^k-1 maximum window
- Extensions to basic behavior
  - Piggy backing for bidirectional flows
  - RR sent in data packets of opposite direction

Go back N under errors

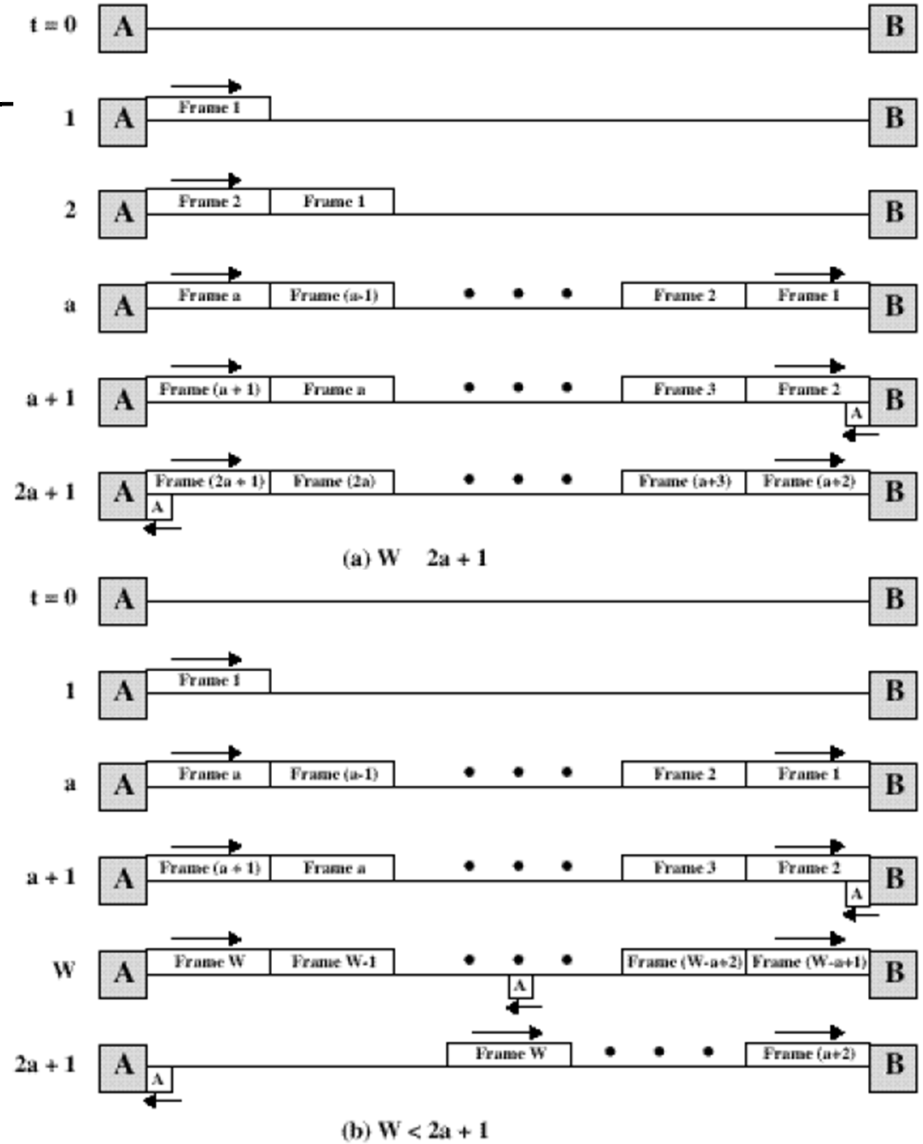
- Discards frames with errors
- Frame out of sequence
  - 1st: receiver sends REJ(k)
    - k is next in-sequence frame expected
  - Following: discard, no REJ
- Sender receives REJ(k)
  - Retransmits frames k, k+1, …
  - Continues with sliding window
- Upon sender timeout
  - Requests receiver to send RR



Efficiency

- $If\ W \geq 1 + 2a$
- $\Rightarrow S = 1$

- $If\ W < 1 + 2a$:
- $\Rightarrow S = W/(1 + 2a)$

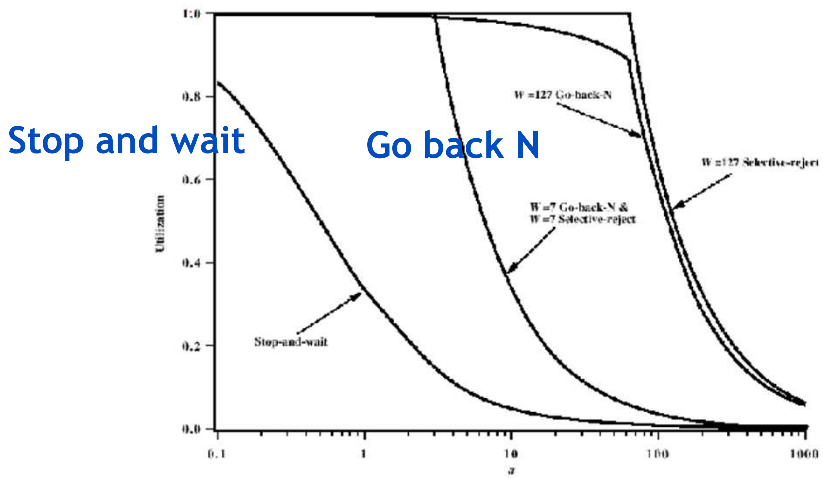


Selective reject ARQ

- Also uses sliding window
- Receiver
  - accepts out-of-sequence frames
  - Confirms negatively missing frames (SREJ)
  - Confirms blocks of frames with RR
- Sender retransmits only SREJ frames
- Suitable for large W
- Maximum window size is $W = \frac{M}{2} = 2^{k-1}$

Efficiency under errors

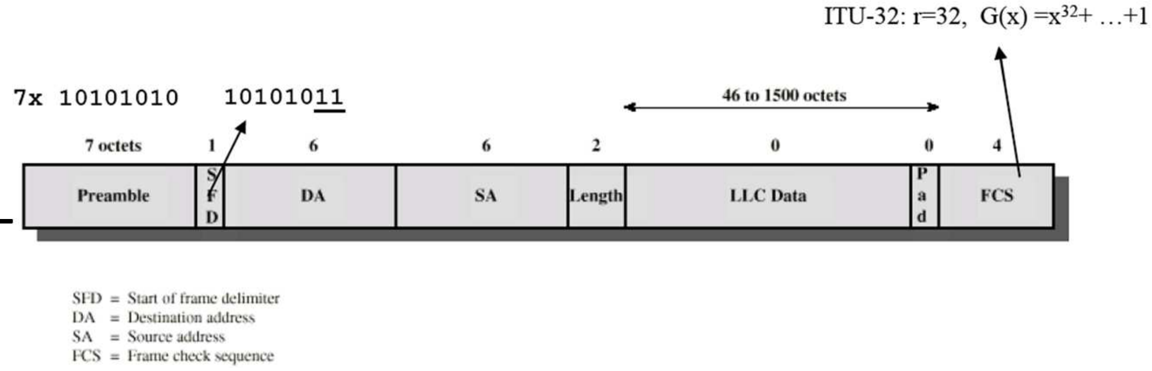
- p_e=FER
- Go back N
  - $S = \frac{(1-p_e)}{1+2ap_e}, W \geq 1 + 2a$
  - $S = \frac{W(1-p_e)}{(1+2a)(1-p_e+Wp_e)}, W < 1 + 2a$
- Selective reject
  - $S = (1 - p_e), W \geq 1 + 2a$
  - $S = \frac{W(1-p_e)}{(1+2a)}, W < 1 + 2a$



ARQ Utilization as a Function of a ($P = 10^{-3}$)

Framing, error detection, and ARQ in common networks

Ethernet



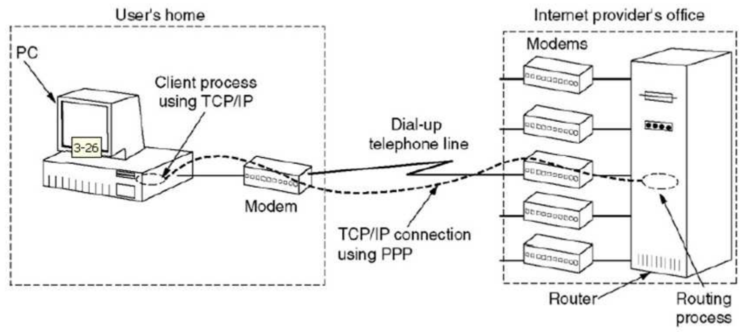
- Framing
  - Preamble + start frame delimiter
  - End of frame: end of Manchester transitions
- Error detection
  - Frame check sequence, CRC ITU-32
- No ARQ
  - Very low BER, low FER
  - Strong CRC, discards frames with errors

PPP, Point-to-Point protocol

- Framing
  - Flags 0x7E, ESC 0x7D

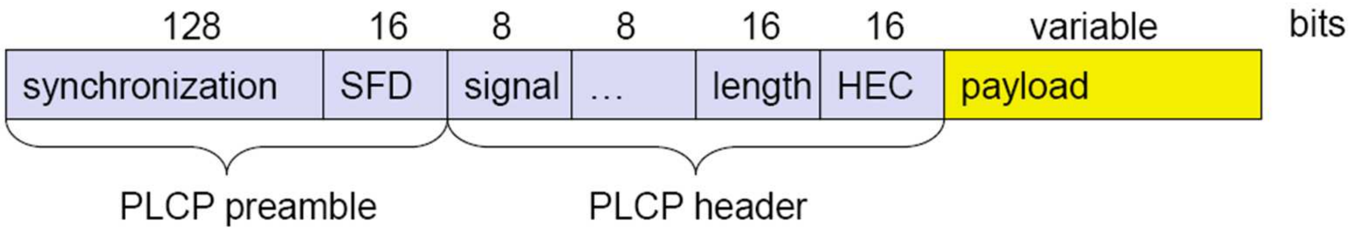
Bytes	1	1	1	1 or 2	Variable	2 or 4	1
	Flag 01111110	Address 11111111	Control 00000011	Protocol	Payload	Checksum	Flag 01111110

Byte stuffing

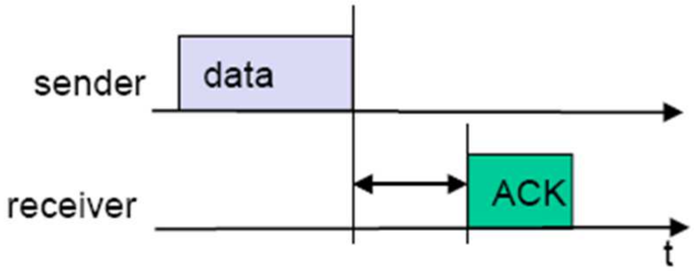


- Error detection negotiable
- No ARQ

Wireless LAN



- Framing
  - 1010101010… synchronization
  - Start frame delimiter
  - Length
- Header error chec
  - ITU-16 CRC
- ARQ
  - Modified Stop and Wait



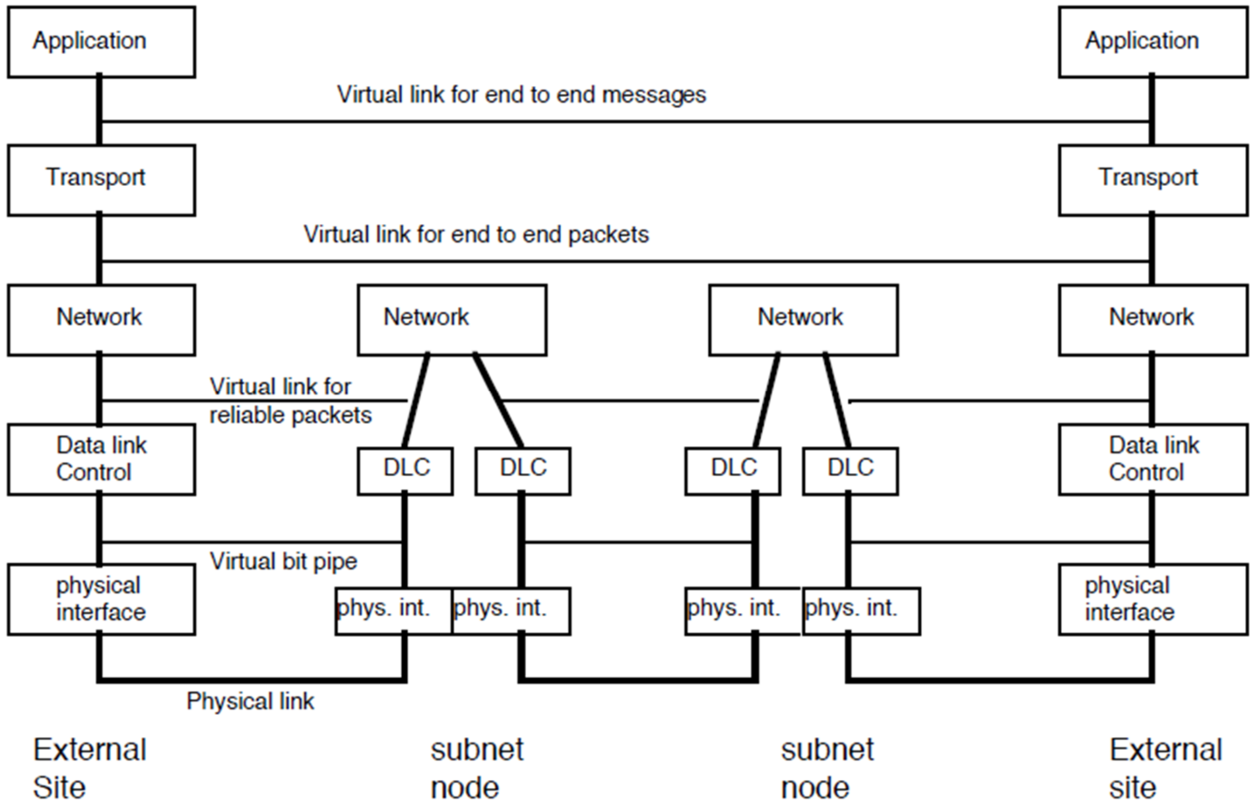
High-Level Data Link Control

- HDLC, bit oriented
- Framing
  - Flags
  - Bit stuffing
- Error detection
  - ITU-16 CRC
- ARQ
  - Selective reject
- Basis for GSM/GPRS/UMTS, …

Bits	8	8	8	≥ 0	16	8
	0 1 1 1 1 1 1 0	Address	Control	Data	Checksum	0 1 1 1 1 1 1 0

Reliability in the protocol stack

Reliability in the protocol stack



FEC, ARQ

ARQ

ARQ

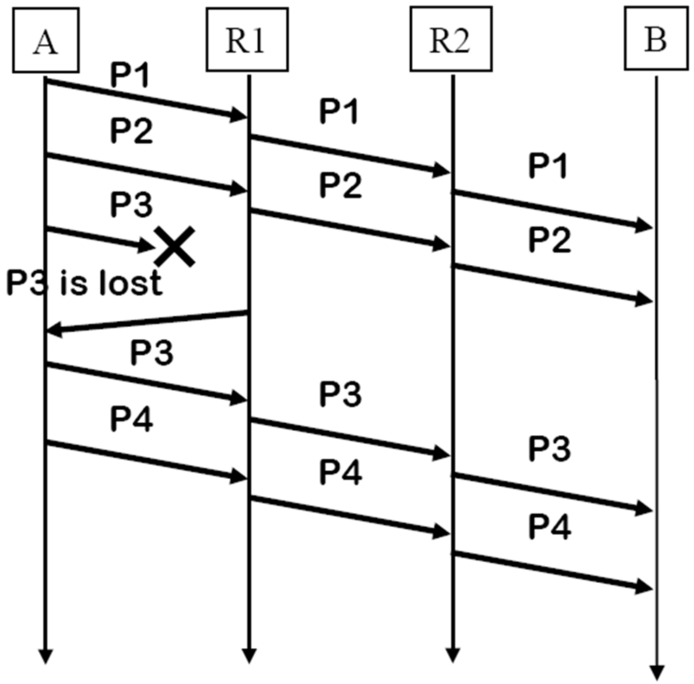
FEC

Forward Error Correction, channel coding

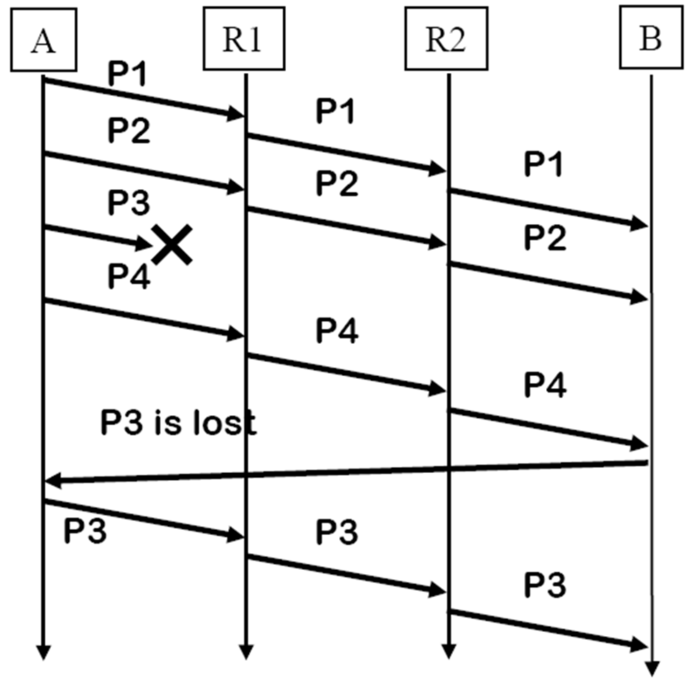
BER and FER

- Layer 2 service either:
  - Delivers packets without errors
  - Discards packets
- Layer 2 transforms bit errors into frame errors, BER => FER
- ARQ solutions for FER
- Two strategies
  - Link-by-link ARQ
  - End-to-end ARQ

Link-by-link vs. End-to-end ARQ



Link-by-link



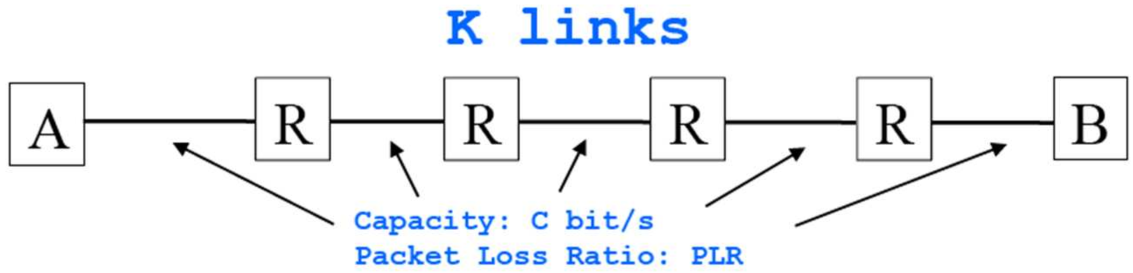
End-to-end

TO THINK

Link-by-link vs. End-to-end ARQ

- Which provides lower delay?
- Which provides higher bitrates?

End-to-end capacity



- Assume no losses in queues
  - Packet Loss Ratio (PLR) = FER
- Link capacity = C*(1-PLR)
- End-to-end
  - LL ARQ, CLL=C*(1-PLR)
  - EE ARQ, CEE=C*(1-PLR)^k
- EE ARQ is inefficient

K	PLR	C_{EE}	C_{LL}
10	0.05	0.6 C_{EE}	0.95 C_{LL}
10	0.0001	0.9990 C_{EE}	0.9999 C_{LL}

Complexity

- LL ARQ is more complex
  - Requires intermediary nodes in the network to process individual flows
  - And store frames in case of retransmission request

ARQ in TCP/IP model

- Packet losses are repaired:
  - At the DLL on lossy channels (e.g. wireless)
  - At the end systems, transport/application

HOMEWORK

- Review slides
- Read:
  - Tanenbaum 3.1, 3.2, 3.6, 3.7
  - Bertsekas – 2.3, 2.4
- Do your Moodle homework