Computer Networks

Lecture 7: Data Link layer

Based on slides from D. Choffnes Northeastern U. and P. Gill from StonyBrook University
Revised Autumn 2015 by S. Laki

Cyclic Redundancy Check (CRC)

- Uses field theory to compute a semi-unique value for a given message

- Much better performance than previous approaches
  - Fixed size overhead per frame (usually 32-bits)
  - Quick to implement in hardware
  - Only 1 in 2^{32} chance of missing an error with 32-bit CRC

CRC (Cyclic Redundancy Check)

- Polynomial code
  - Treating bit strings as representations of polynomials with coefficients of 0 and 1.
- CRC
  - Add k bits of redundant data to an n-bit message.
  - Represent n-bit message as an n-1 degree polynomial;
    - e.g., MSG=10011010 corresponds to $M(x) = x^7 + x^4 + x^3 + x^1$.
  - Let k be the degree of some divisor polynomial G(x);
    - e.g., $G(x) = x^3 + x^2 + 1$.
    - Generator polynomial
      - Agreed upon it in advance

CRC

- Transmit polynomial P(x) that is evenly divisible by G(x), and receive polynomial P(x) + E(x);
  - E(x)=0 implies no errors.
- Recipient divides (P(x) + E(x)) by G(x);
  - the remainder will be zero in only two cases:
    - E(x) was zero (i.e. there was no error),
    - or E(x) is exactly divisible by C(x).
- Choose G(x) to make second case extremely rare.

A basic example with numbers

- Make all legal messages divisible by 3
- If you want to send 10
  - First multiply by 4 to get 40
  - Now add 2 to make it divisible by 3 = 42
- When the data is received ..
  - Divide by 3, if there is no remainder there is no error
  - If no error, divide by 4 to get sent message
- If we receive 43, 44, 41, 40, then error
- 45 would not be recognized as an error

Mod 2 arithmetic

- Operations are done modulo 2

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A - B
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A · B
0	0	0
0	1	0
1	0	0
1	1	1

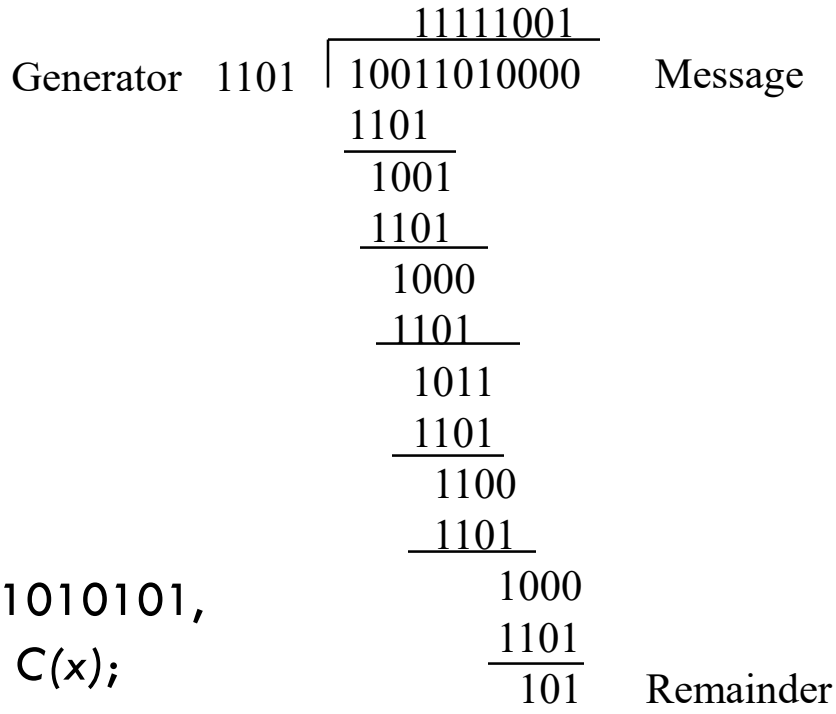
```
  0110111011
+ 1101010110
= 1011101101
```

```
   1111
  +1010
  =====
   0101
```

```
    11001
  x  101
  =====
    11001
+   11001
=========
  1111101
```

A basic example with polynomials

- Sender:
  - multiply $M(x) = x^7 + x^4 + x^3 + x^1$ by x^k; for our example, we get
    - $x^{10} + x^7 + x^6 + x^4$ (10011010000);
  - divide result by $C(x)$ (1101);



Send 10011010000 + 101 = 10011010101,
since this must be exactly divisible by $C(x)$;

Further properties

- Want to ensure that $G(x)$ does not divide evenly into polynomial $E(x)$.
- All single-bit errors, as long as the x^k and x^0 terms have non-zero coefficients.
- All double-bit errors, as long as $G(x)$ has a factor with at least three terms.
- Any odd number of errors, as long as $G(x)$ contains the factor $(x + 1)$.
- Any "burst" error (i.e sequence of consecutive errored bits) for which the length of the burst is less than k bits.
- Most burst errors of larger than k bits can also be detected.

Even Parity

Actually consists of using x+1 polynomial

Given message 0111, multiply by x to get 01110

Now divide by x+1=11

```
     0101
   ------
11 | 01110
       11
       0010
          11
            1=remainder
```

Message = 01110+1=01111 even parity

- Common polynomials for $C(x)$:

CRC	$C(x)$
CRC-8	$x^8+x^2+x^1+1$
CRC-10	$x^{10}+x^9+x^5+x^4+x^1+1$
CRC-12	$x^{12}+x^{11}+x^3+x^2+x^1+1$
CRC-16	$x^{16}+x^{15}+x^2+1$
CRC-CCITT	$x^{16}+x^{12}+x^5+1$
CRC-32	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

Error control

- Error Control Strategies
  - Error Correcting codes (Forward Error Correction (FEC))
  - Error detection and retransmission Automatic Repeat Request (ARQ)

Error control

- Objectives
  - Error detection
    - with correction
      - Forward error correction
    - without correction -> e.g. drop a frame
      - Backward error correction
      - The erroneous frame needs to be retransmitted
  - Error correction
    - without error detection
      - e.g. in voice transmission

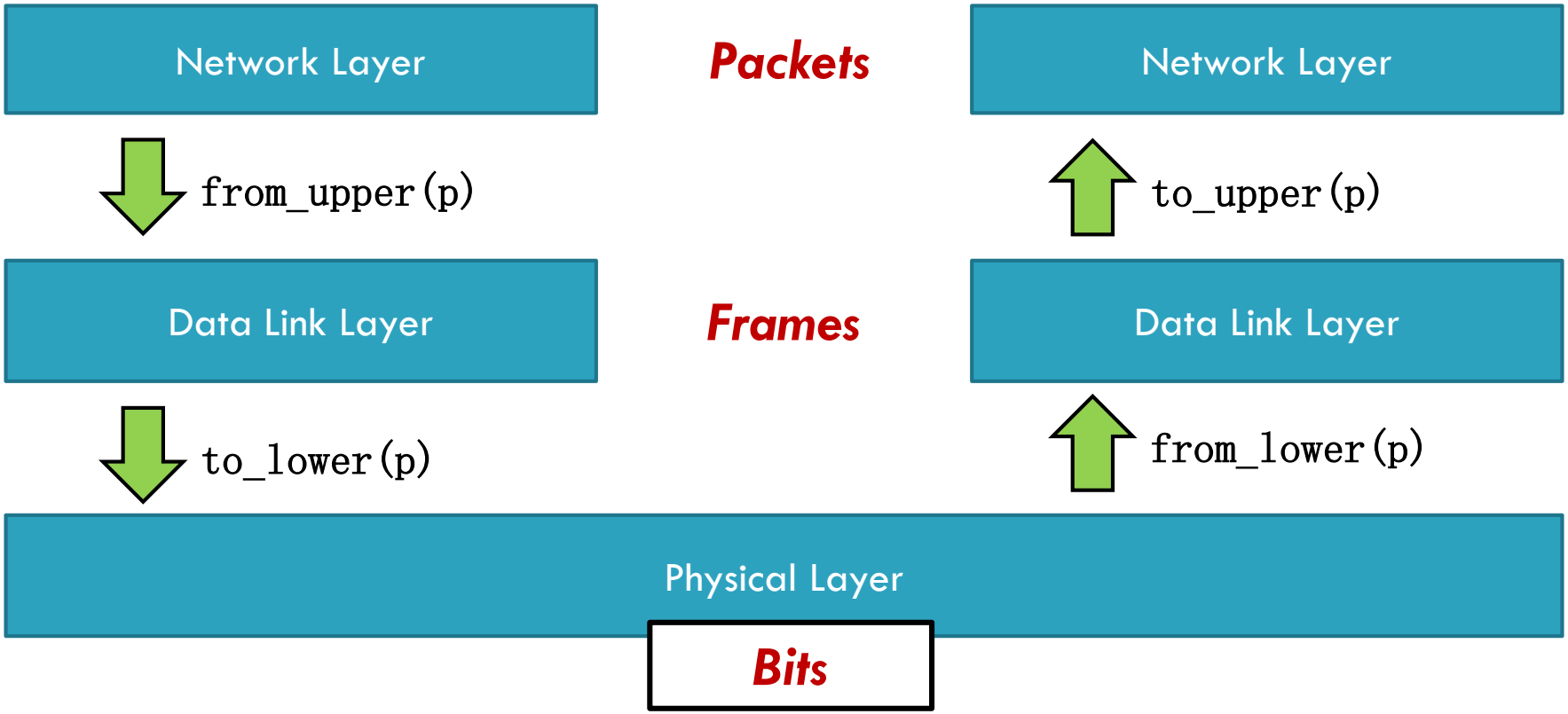
Should We Error Check in the Data Link?

- Recall the End-to-End Argument
- Cons:
  - Error free transmission cannot be guaranteed
  - Not all applications want this functionality
  - Error checking adds CPU and packet size overhead
  - Error recovery requires buffering
- Pros:
  - Potentially better performance than app-level error checking
- Data link error checking in practice
  - Most useful over lossy links
  - Wifi, cellular, satellite

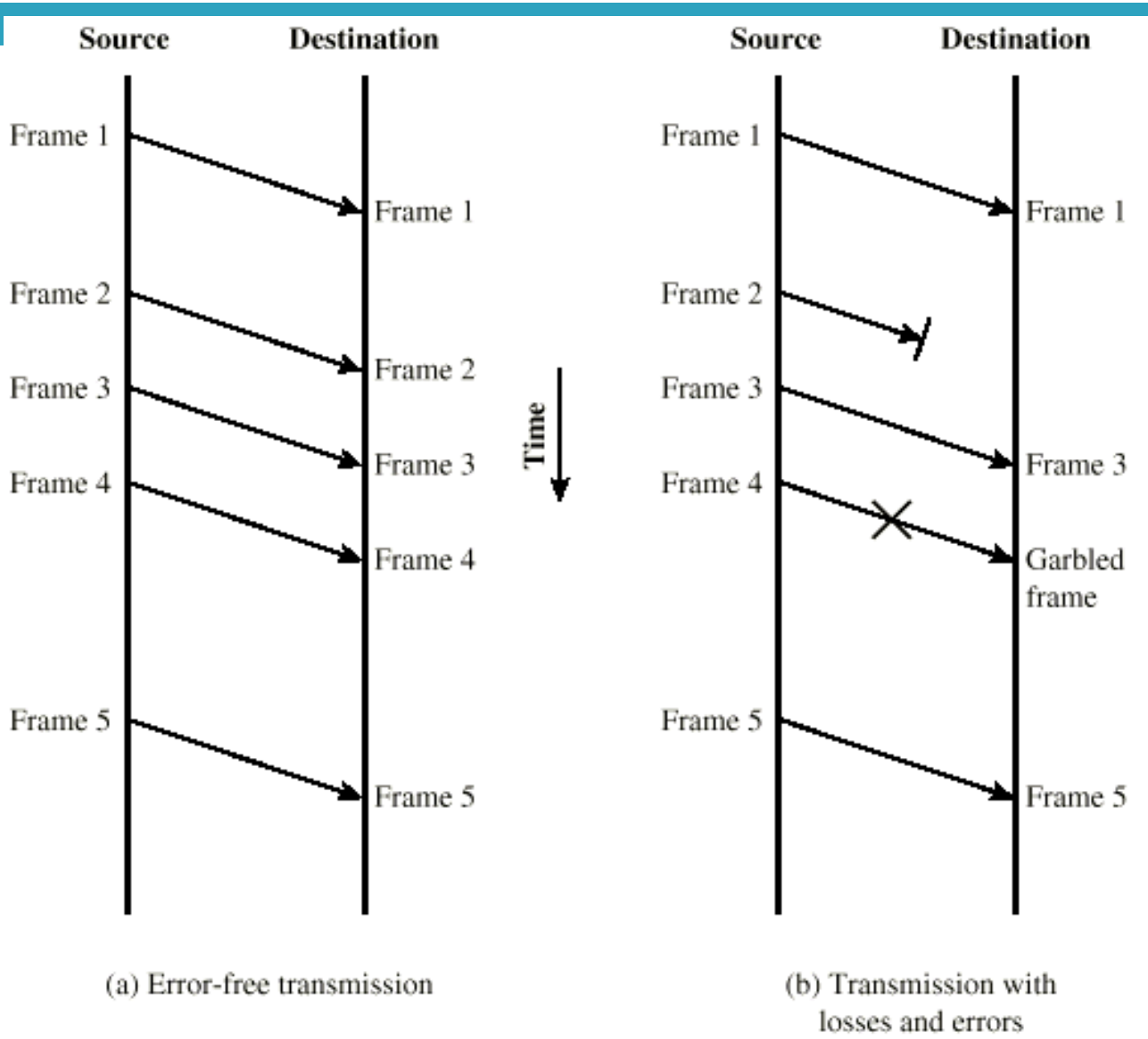
Backward Error Correction

Backward error correction

- Error detection at the receiver side
- The sender retransmits a frame until it received by the other side correctly.



Model of Frame Transmission



(a) Error-free transmission

(b) Transmission with losses and errors

Elementary Data Link Protocols

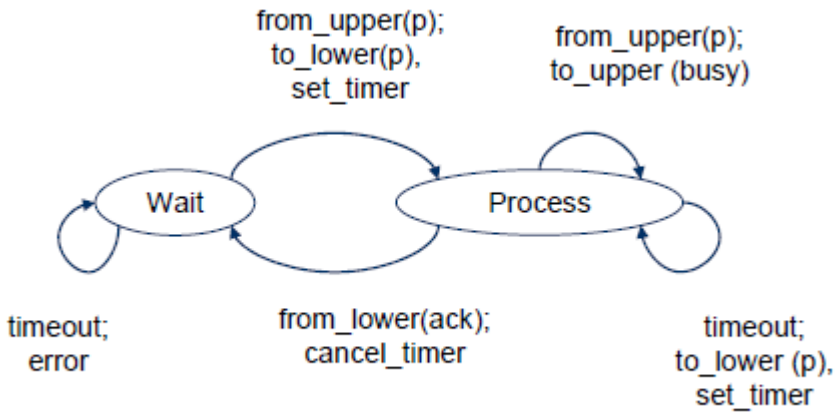
- Simplex Stop-and-Wait Protocol
- Alternate Bit Protocol
- Sliding Window Protocol

Simple Stop-and-Wait Protocol

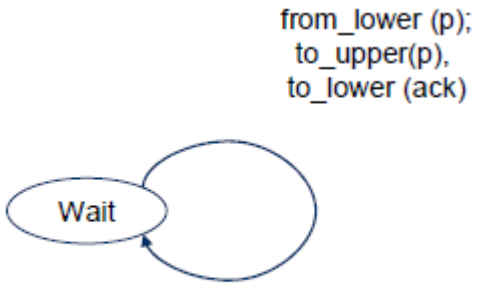
- A sends a message to B
- A stops and waits for an answer from B
  - Acknowledgement message (ACK)
- After receiving the message B sends an ACK back to the sender.
- A retransmits the message until it receives an ACK from B
- If the ACK arrived, the next message may be sent.

Simplex Stop-and-Wait Protocol

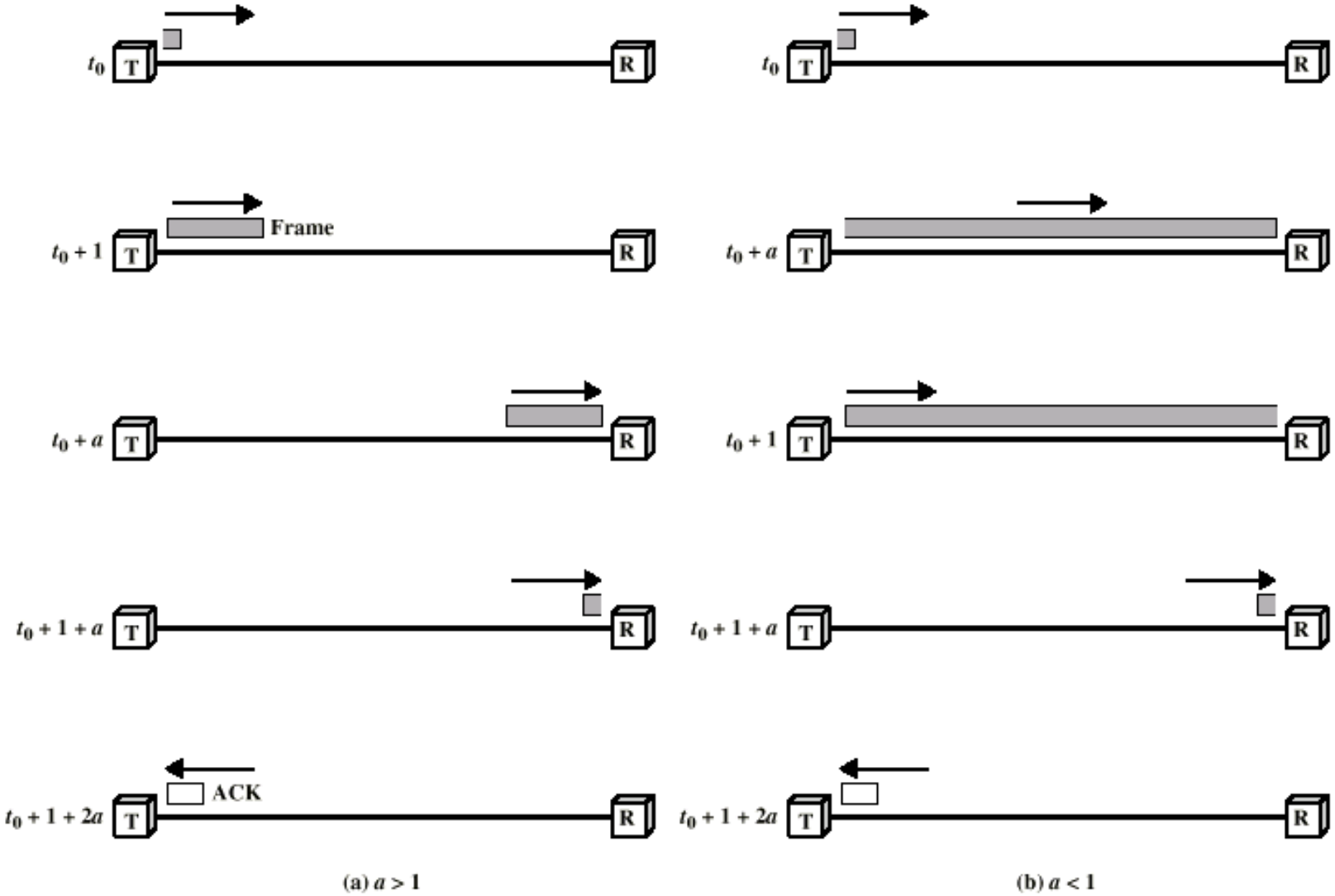
Sender



Receiver



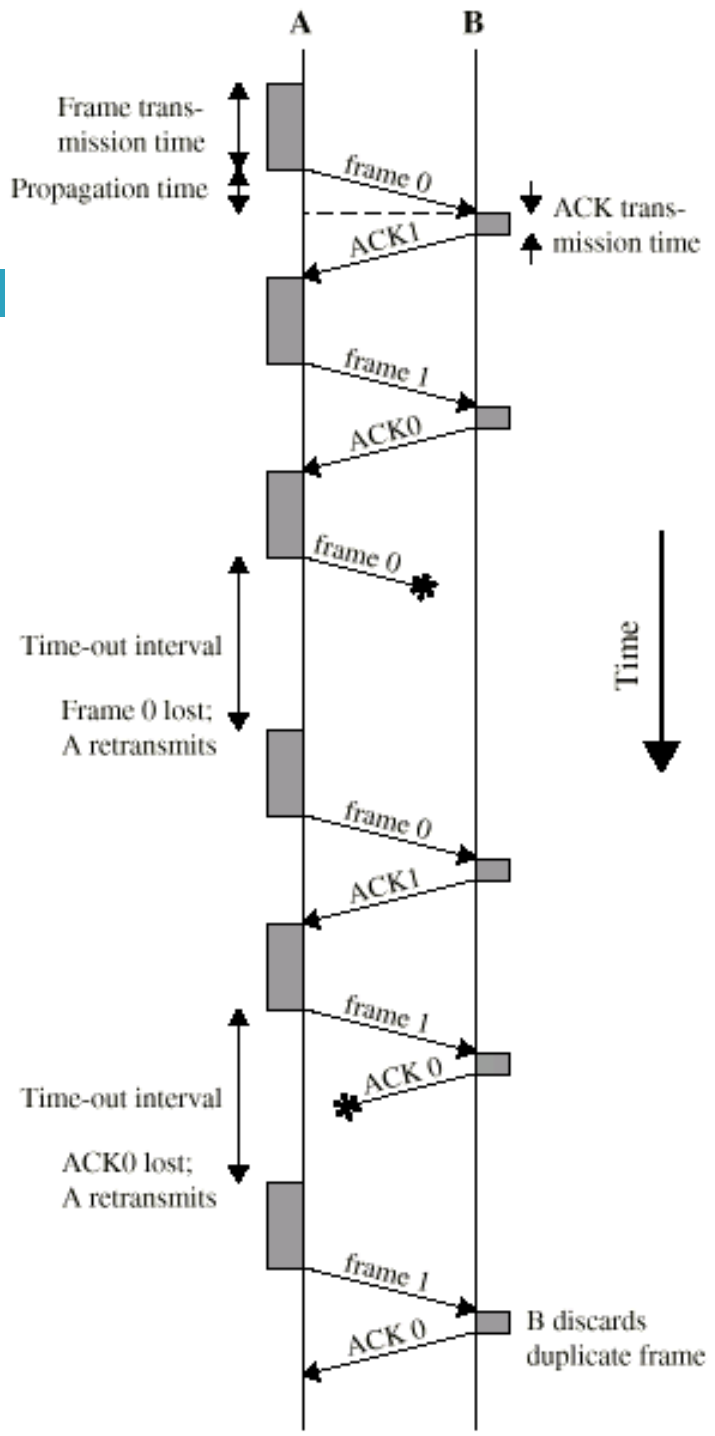
Stop-and-Wait Link Utilization



Stop-and-Wait Diagram

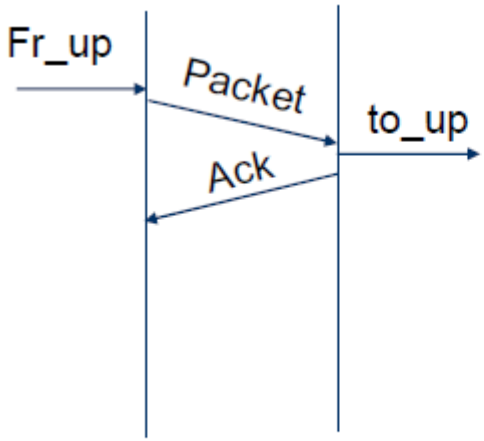
Simple, but inefficient for long distance and high speed applications.

We can use sliding-window technique to improve the efficiency.

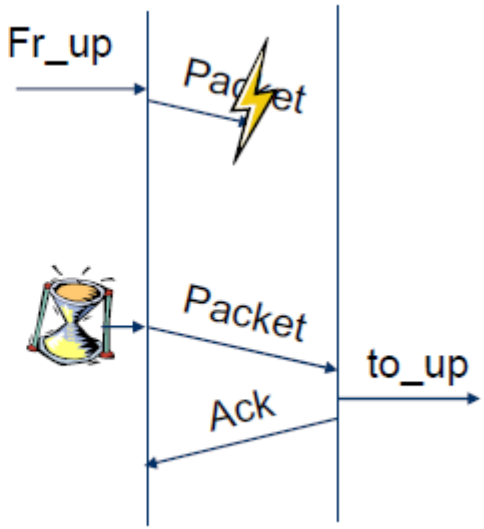


What's the problem?

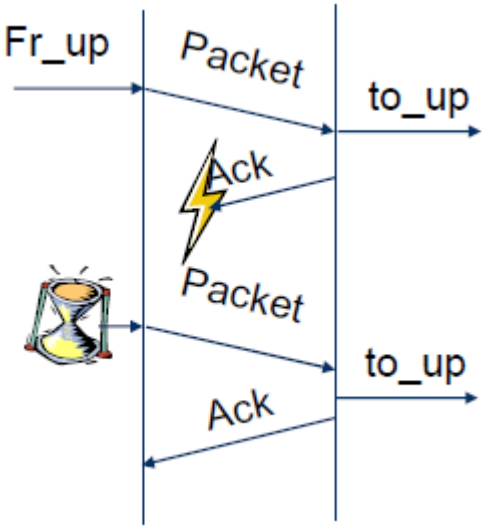
Usually



Packet loss



ACK loss

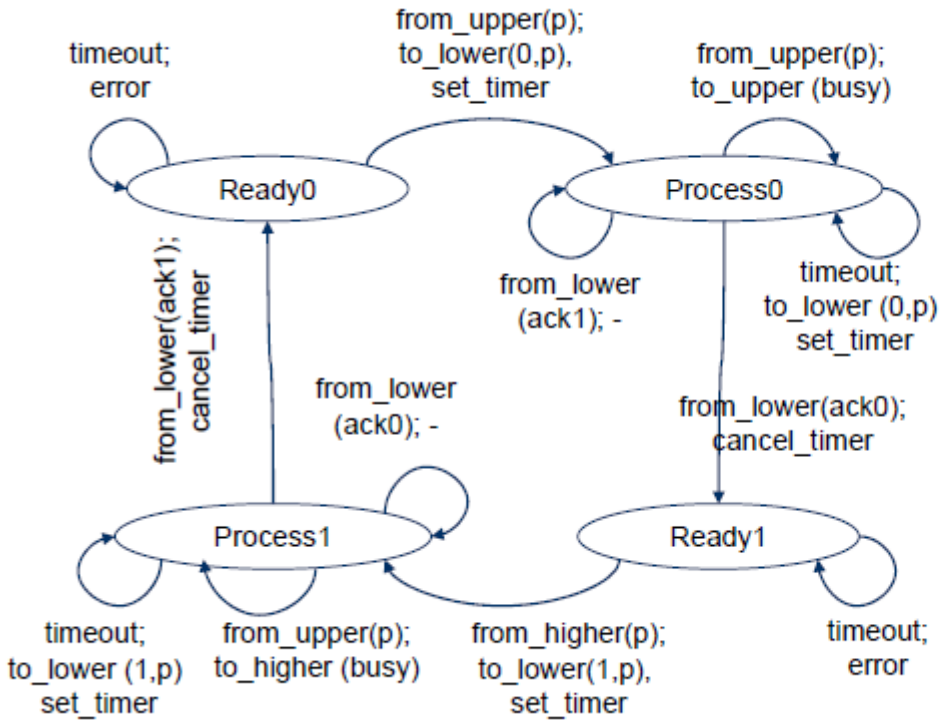


Alternating Bit Protocol (ABP)

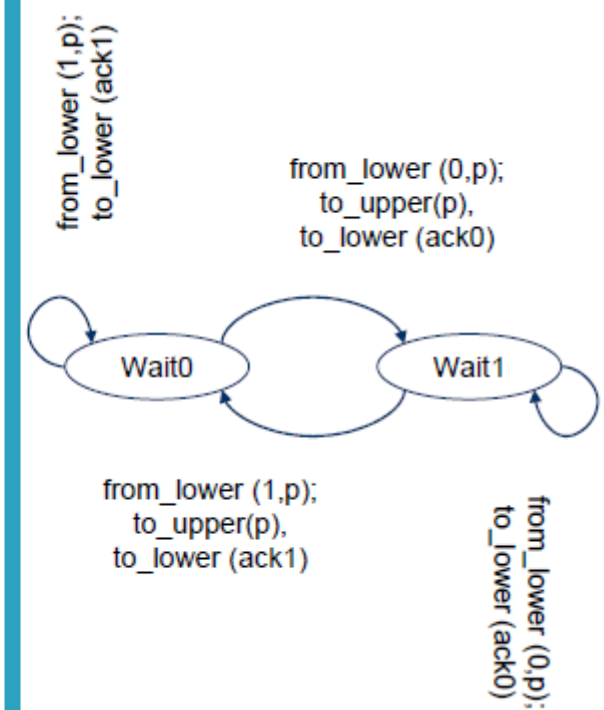
- **Let**
  - A be the sender
  - B be the receiver
- **A and B maintain internal one-bit counter**
  - A value that is 0 or 1
- **Each message from A to B contains**
  - a data part and
  - a one-bit sequence number
    - E.g. a value that is 0 or 1
- After receiving A's message, **B sends an ACK back to A**
  - which also contains a one-bit sequence number
- **Retransmission until A receives an ACK from B with the same sequence number**
  - Then A complements its sequence number
    - 0->1 or
    - 1->0

Alternating Bit Protocol (ABP)

Sender



Receiver

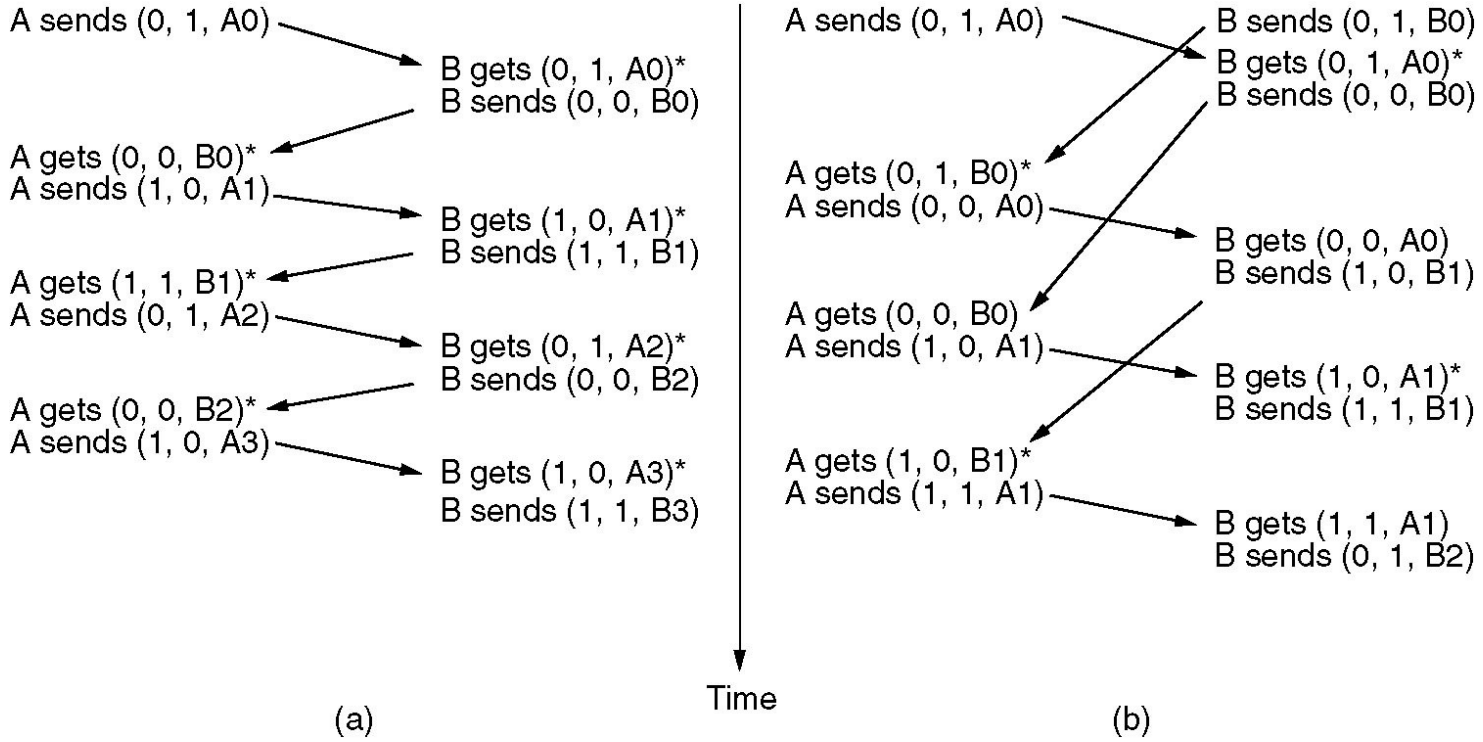


Alternating Bit Protocol (ABP)

- A reliable data transport over a noisy channel
- Basic flow control
  - The sender has to wait for the ACK from the receiver before sending the next message
- Automatic Repeat reQest (ARQ) protocol

- An acknowledgement
  - marks that the new message has been delivered.
  - allows the sender to tranmit the next frame.

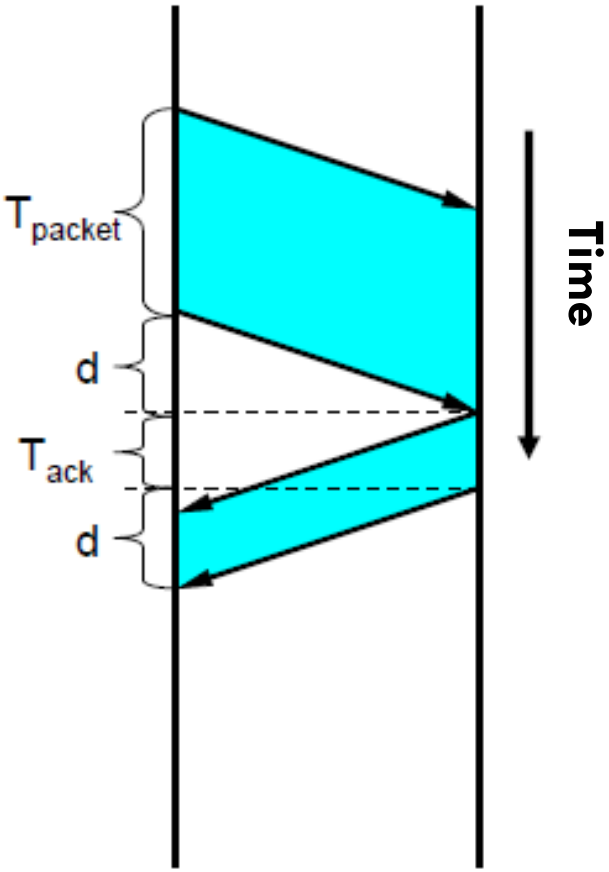
Alternating Bit Protocol



Two scenarios for ABP. (a) Normal case. (b) Abnormal case. The notation is (seq, ack, packet number). An asterisk indicates where a network layer accepts a packet.

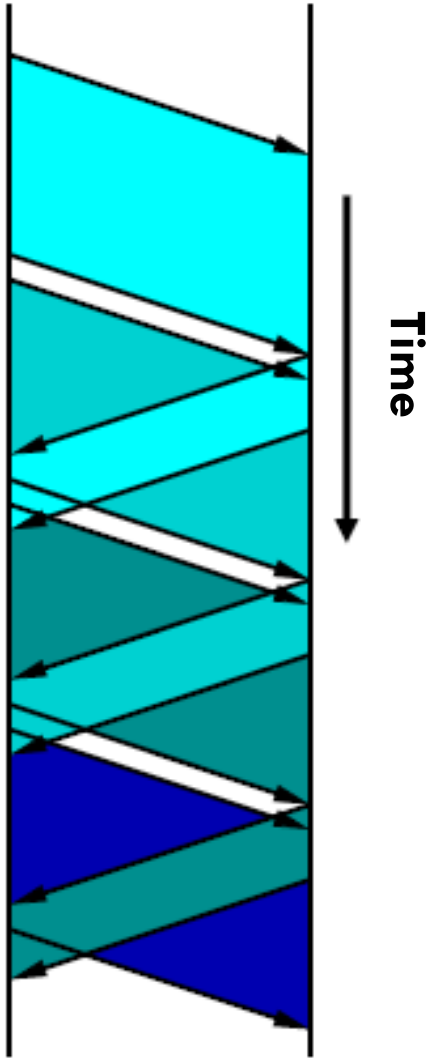
ABP – Channel utilization

- Utilization (η) is the ratio of
  - The time needed for the transmission of a frame (T_{packet})
  - The time ellapsed until the next frame can be transmitted
    - In the fig.: (T_{packet} + d + T_{ack} + d)
- Now:
  - $\eta = T_{packet} / (T_{packet} + d + T_{ack} + d)$
- If the propagation delay is large, the ABP is not efficient.



How to improve the efficency?

- The sender transmit frames continously one after another
  - More frames are sent out, but not acknowledged.
  - Pipeline technique
- Introduce sequence numbers



Sliding Window Protocols

- Similar to ABP
- but allow multiple frames to transmit
  - Receiver has a buffer of W frames
  - Sender can send up to W frames without receiving ACK
- Bidirectional

- Each outgoing frame contains a seq. number from 0 to 2^n-1.
  - So it fits in an n-bit field
  - ABP uses n=1

- Each ACK carries the sequence number of the next expected frame by the receiver

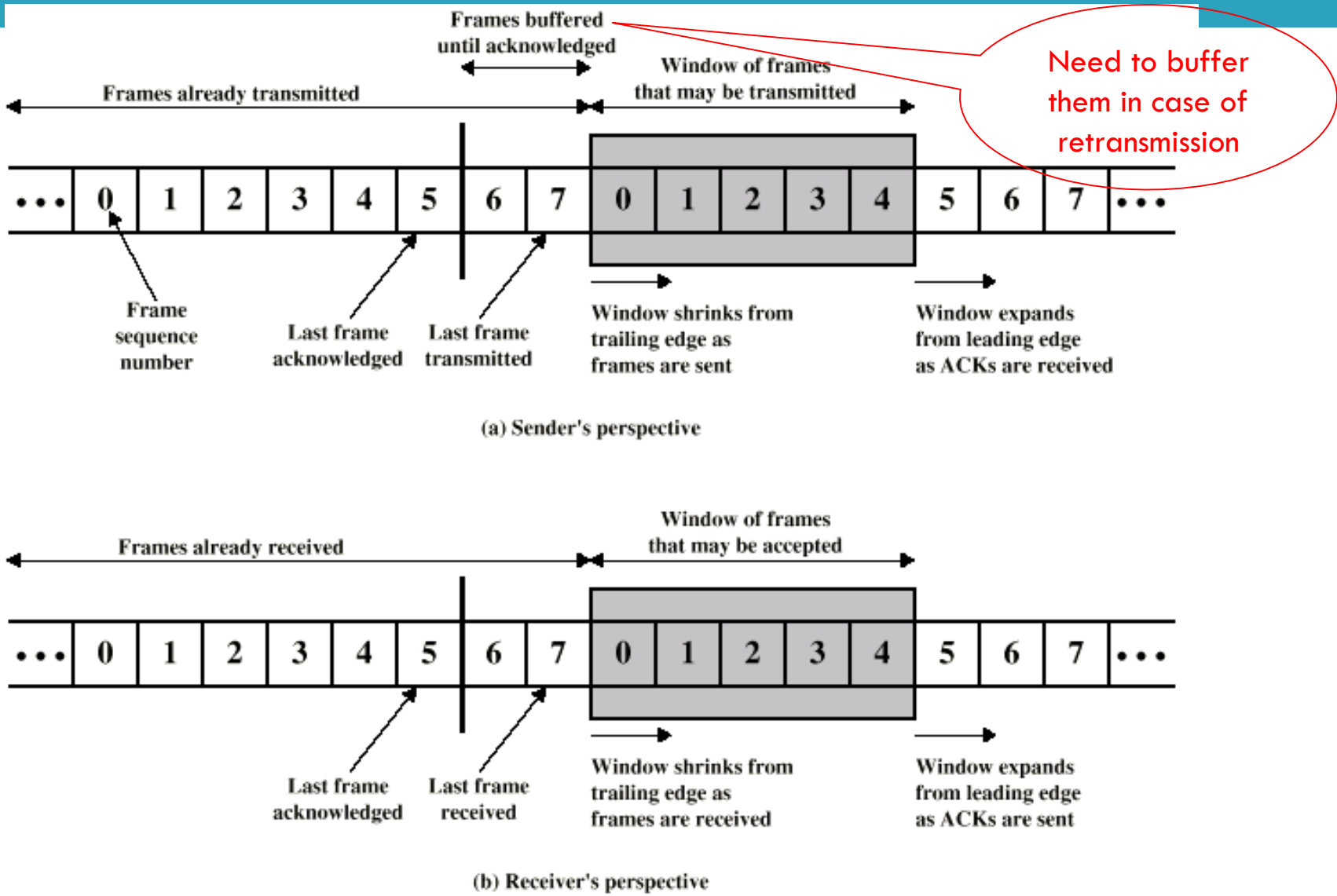
Sliding Window Protocols

- **At the sender**
  - Sending window
    - a set of sequence numbers corresponding to frames being under transmission (finite range of numbers)
- **At the receiver**
  - Receiving window
    - Sequence numbers for frames it is permitted to accept (finite range of numbers)

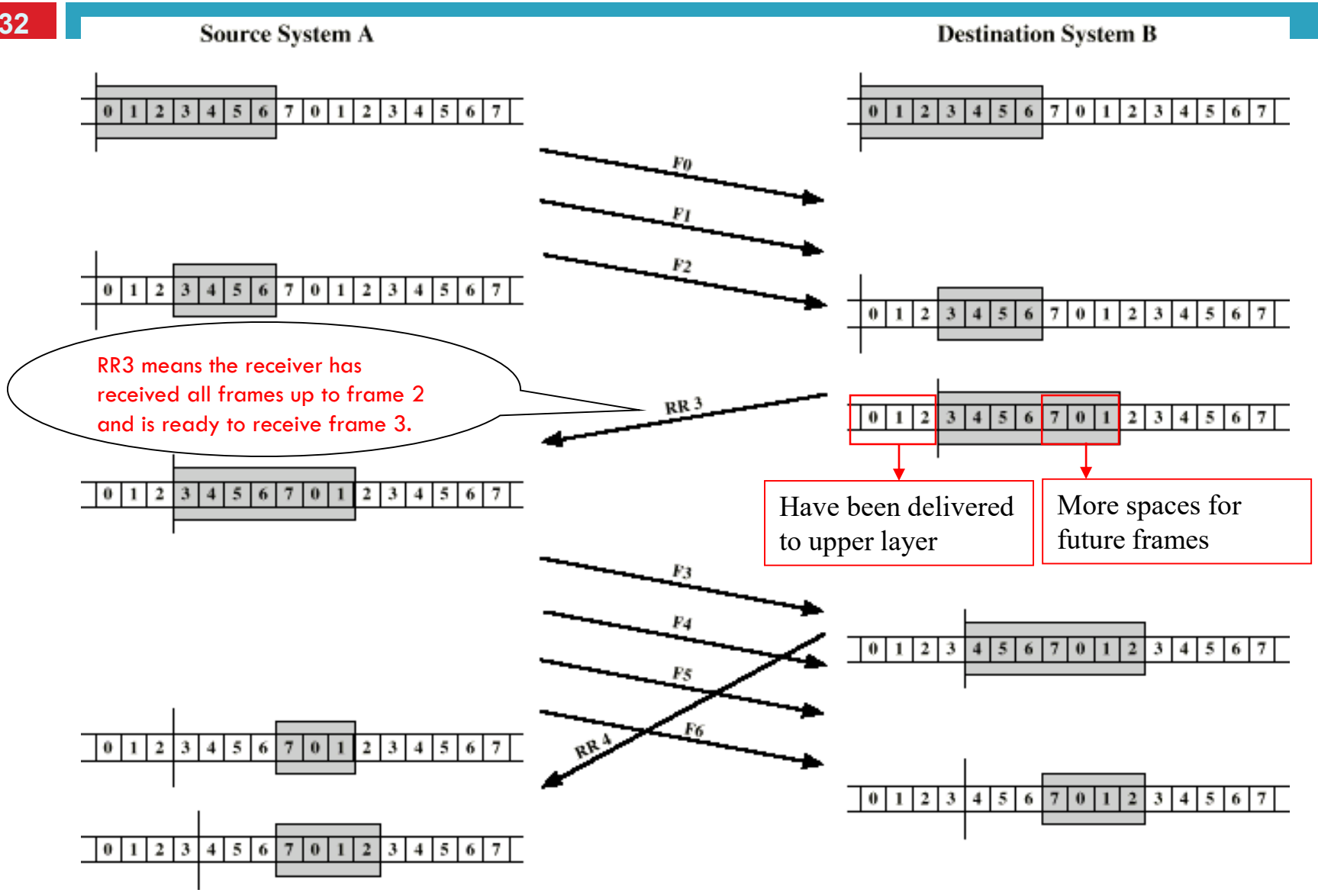
- The sender's and receiver's windows need
  - not have tha same lower and upper bounds and
  - even have the same size.

- The window size can be
  - fixed or
  - grow or shrink over the course of time as frames are sent and received

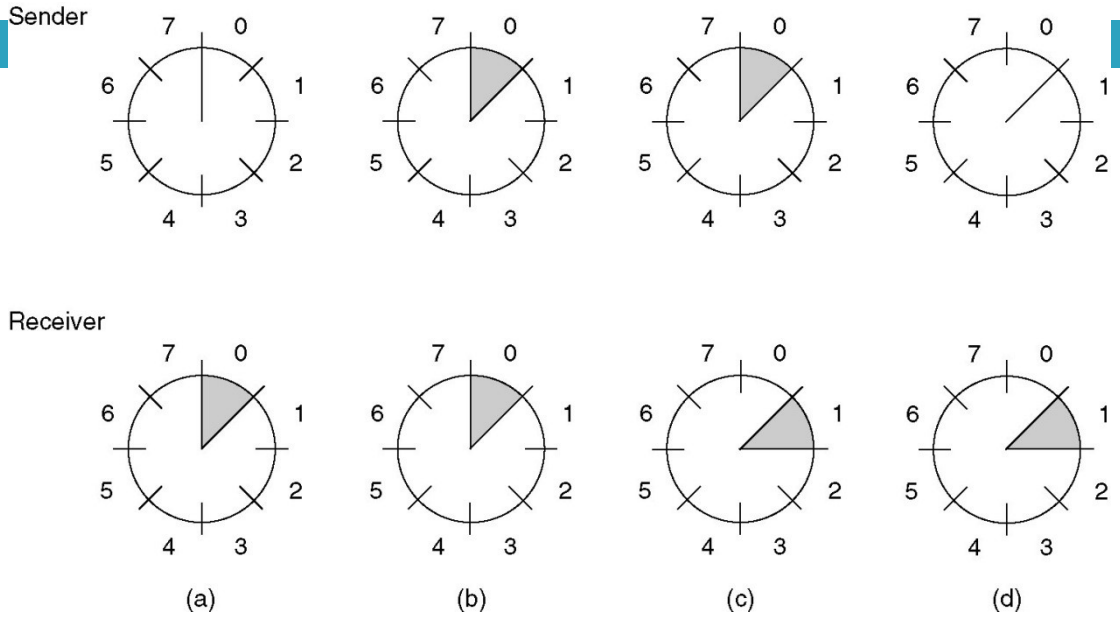
Sliding-Window Diagram



Example Sliding-Window



Sliding Window Protocols



A sliding window of size 1, with a 3-bit sequence number.

(a) Initially.

(b) After the first frame has been sent.

(c) After the first frame has been received.

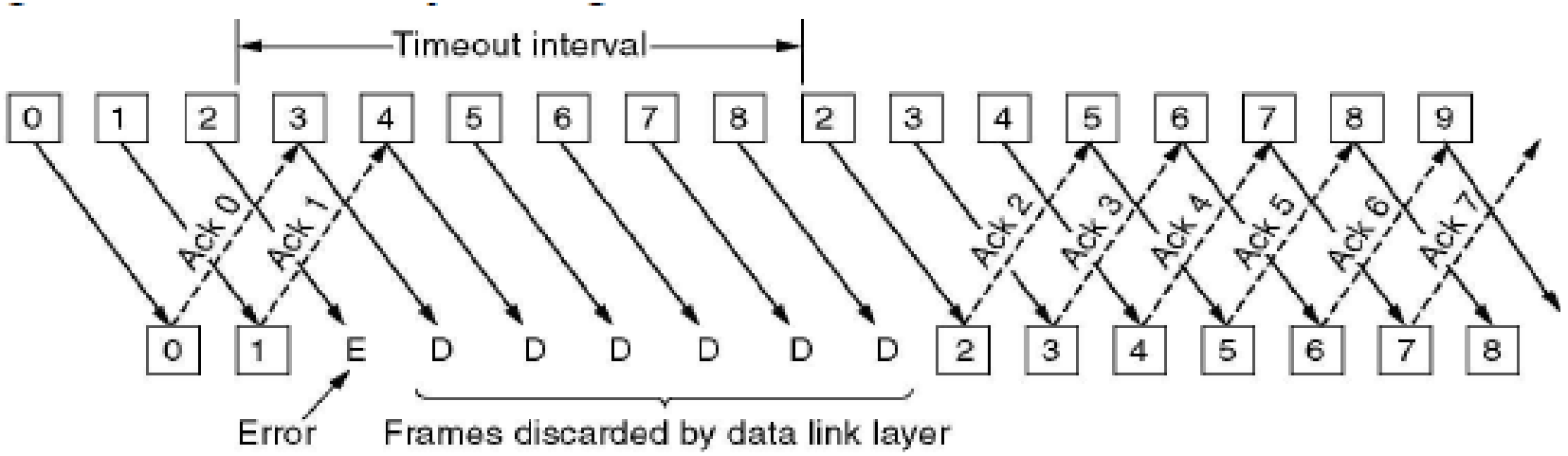
(d) After the first acknowledgement has been received.

Go-Back-N

- A sliding window protocol where
  - the receiver's window size is fixed to 1,
  - while the sender has window size > 0.

- After receiving a damaged frame
  - Receiver discards all subsequent frames
  - Sender retransmits the damaged frame and all its successors after the times out

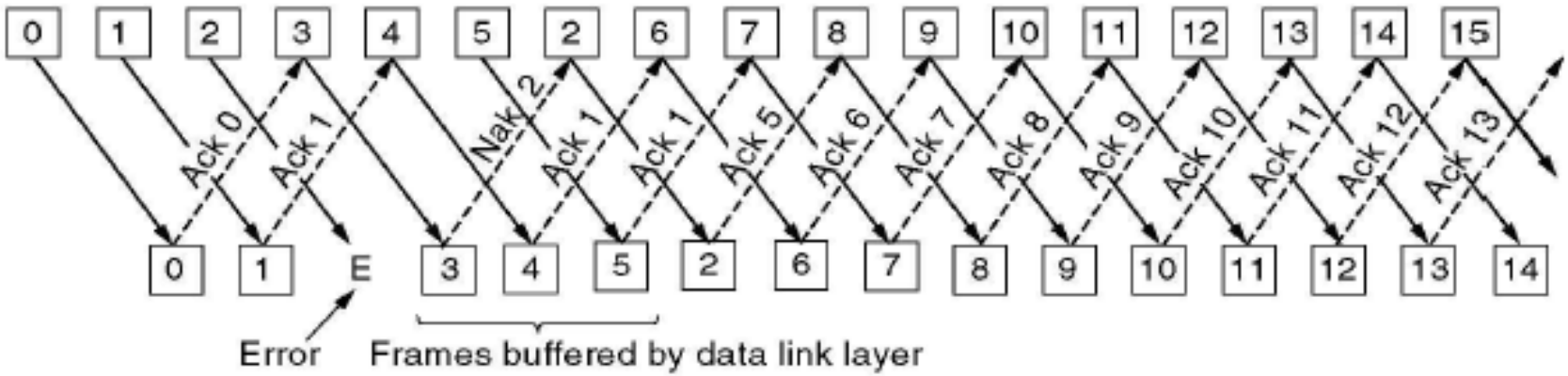
Go-Back-N


Timeout interval
0 1 2 3 4 5 6 7 8 2 3 4 5 6 7 8 9
Ack 0
Ack 1
Ack 2
Ack 3
Ack 4
Ack 5
Ack 6
Ack 7
0 1 E D D D D D D 2 3 4 5 6 7 8
Error
Frames discarded by data link layer

Selective Repeat

- Receiver's window size is n (n >1)
  - At most n frames can be buffered
- Receiver stores all the correct frames following the bad one
- The sender retransmits only the bad frame not all its successors

Selective Repeat



Communication channels and piggybacking

- Simplex
  - Communication in one direction only
- Half-duplex
  - Communication in both directions, but only one direction at a time, not simultaneously.
- Full-duplex
  - Communication in both directions simultaneously

- The previous protocols assumed
  - a simplex channel to the upper (network) layer and
  - a (half-)duplex channel to the physical layer

- If we use duplex channel to the upper layers
  - Transmitting data packet and acknowledgements in both directions separately
  - Or using piggybacking
    - The header of a data packet sent in the opposite direction carries the acknowledgement back to the other side
    - widely applied in practice

Ethernet frame

802.3 Ethernet frame structure

Preamble	Start of frame delimiter	MAC destination	MAC source	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)	Payload	Frame check sequence (32-bit CRC)	Interframe gap
7 octets	1 octet	6 octets	6 octets	(4 octets)	2 octets	42[note 2]–1500 octets	4 octets	12 octets
		64–1522 octets						
72–1530 octets								
84–1542 octets								

Outline

- Framing
- Error Checking and Reliability
- Media Access Control
  - 802.3 Ethernet
  - 802.11 Wifi

What is Media Access?

- Ethernet and Wifi are both multi-access technologies
  - Broadcast medium, shared by many hosts
  - Simultaneous transmissions cause collisions
    - This destroys the data
- Media Access Control (MAC) protocols are required
  - Rules on how to share the medium
  - Strategies for detecting, avoiding, and recovering from collisions

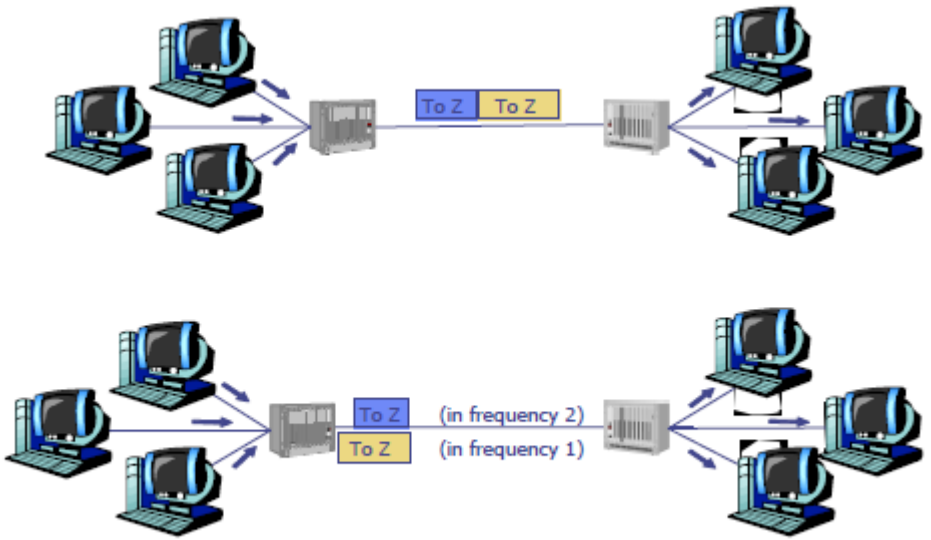
Strategies for Media Access

- Channel partitioning
  - Divide the resource into small pieces
  - Allocate each piece to one host
  - Example: Time Division Multi-Access (TDMA) cellular
  - Example: Frequency Division Multi-Access (FDMA) cellular
- Taking turns
  - Tightly coordinate shared access to avoid collisions
  - Example: Token ring networks

Channel Partitioning

- Frequency Division Multiplexing
  - E.g. a telephone trunk
- Time Division Multiplexing

- Are they good solutions?
  - if data rates are fixed
  - if the link utilization is good



What's the problem?

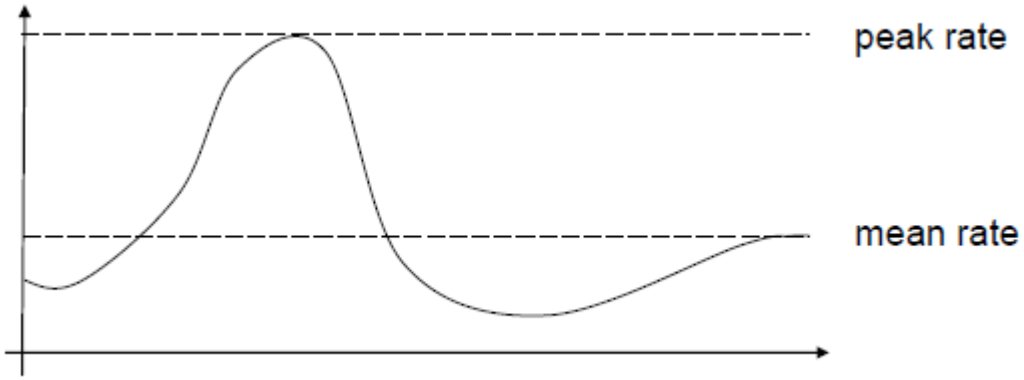
The number of senders is large

continuously varying

bursty traffic

Bursty traffic

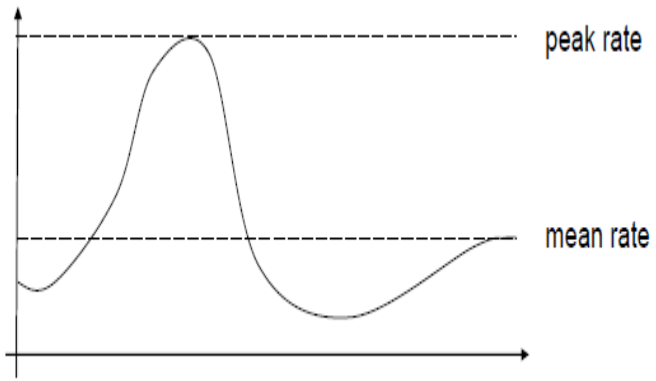
- If there's **huge difference** between the **peak** rate and the **mean** (or **average**) rate
- In computer networks it is not rare
  - peak rate/mean rate = 1000/1

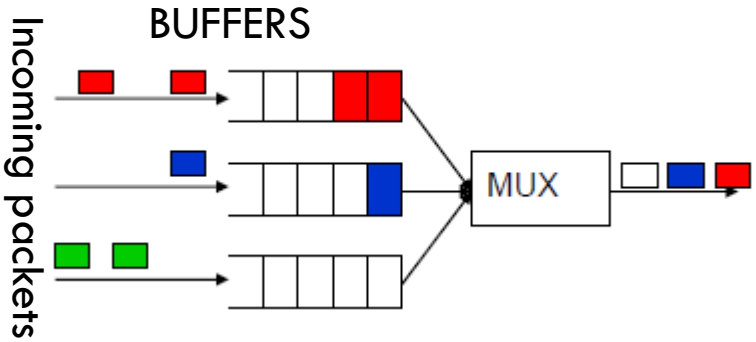


Bursty traffic with Static Channel Allocation

The capacity of the channels must be

- Either quite large to handle peak rates
  - Waste of resources
  - The mean rate is much less than the peak one
- Or based on the mean rate
  - We need buffers/queues to store packets coming faster than the mean rate
  - Queuing delays



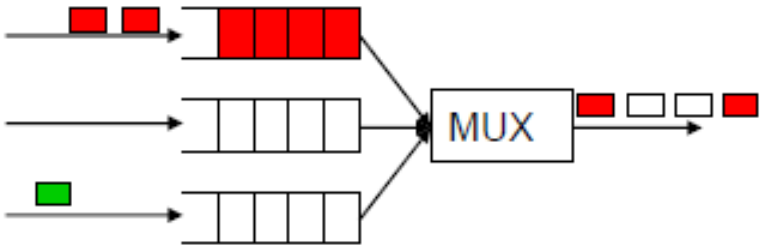


What's about delays?

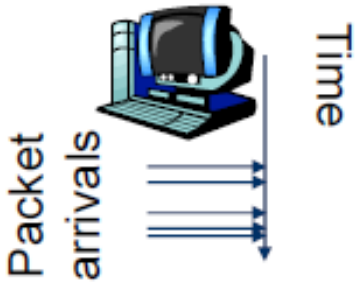
- **If there's no multiplexing**
  - A source has data rate of p bps
  - The capacity of the link is C bps
  - The delay is T
- **Bursty traffic with static channel allocation**
  - Dividing the channel into N static subchannels
    - With sending rate p/N bps and capacity C/N
  - The delay is N T

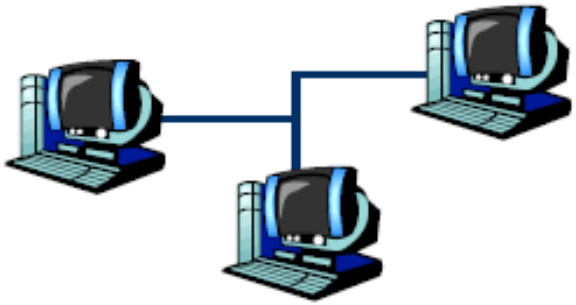
Static channel allocation increases the packet delays

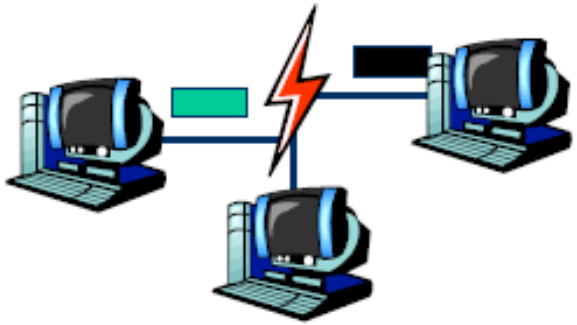
- Because of the idle subchannels



Dynamic Channel Allocation in LANs and MANs







1. Station Model.
   - N terminals/hosts
   - The prob. of a frame being generated in Δt is $\lambda\Delta t$, where the arrival rate is λ.

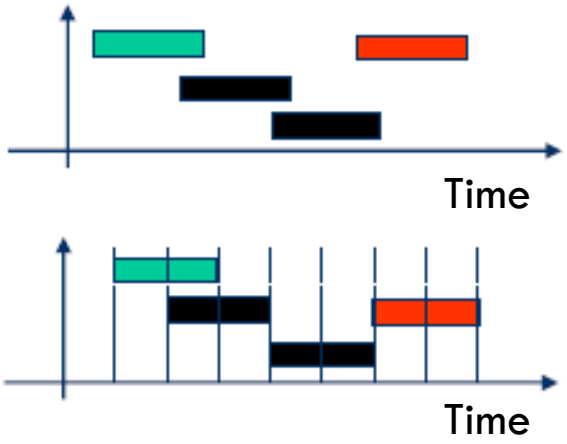
2. Single Channel Assumption.
   - All stations are equivalent
   - A single channel is available for all communications

3. Collision Assumption.
   - If two frames are transmitted simultaneously, they overlap in time which results a garbled signal
   - This event is called collision

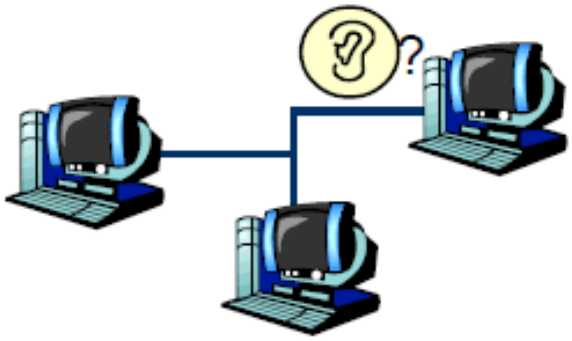
4. Continuous Time VS Slotted Time.

5. Carrier Sense VS No Carrier Sense.

Dynamic Channel Allocation in LANs and MANs



4. Continuous Time VS Slotted Time.



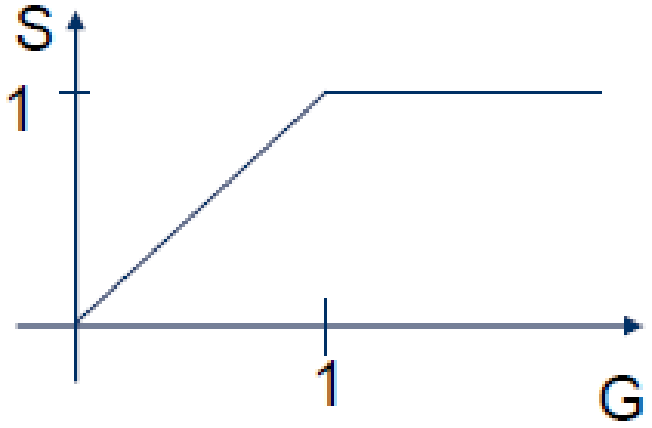
5. Carrier Sense VS No Carrier Sense.

How can the efficiency be measured?

- **Throughput (S)**
  - Number of packets/frames transmitted in a time unit (successfully)
- **Delay**
  - The time needs for transmitting a packet
- **Fairness**
  - All the terminals are treated as equals

Throughput and offered load

- **Offered load (G)**
  - The number of packets in a time unit that the protocol must handle
  - G>1: overloading



- **An ideal protocol**
  - If G<1, S=G
  - If G≥1, S=1
  - where sending out a packet takes 1 time unit.

Strategies for Media Access

- Channel partitioning
  - Divide the resource into small pieces
  - Allocate each piece to one host
  - Example: Time Division Multi-Access (TDMA) cellular
  - Example: Frequency Division Multi-Access (FDMA) cellular
- Taking turns
  - Tightly coordinate shared access to avoid collisions
  - Example: Token ring networks
- Contention
  - Allow collisions, but use strategies to recover
  - Examples: Ethernet, Wifi

Contention MAC Goals

- Share the medium
  - Two hosts sending at the same time collide, thus causing interference
  - If no host sends, channel is idle
  - Thus, want one user sending at any given time
- High utilization
  - TDMA is low utilization
  - Just like a circuit switched network
- Simple, distributed algorithm
  - Multiple hosts that cannot directly coordinate
  - No fancy (complicated) token-passing schemes

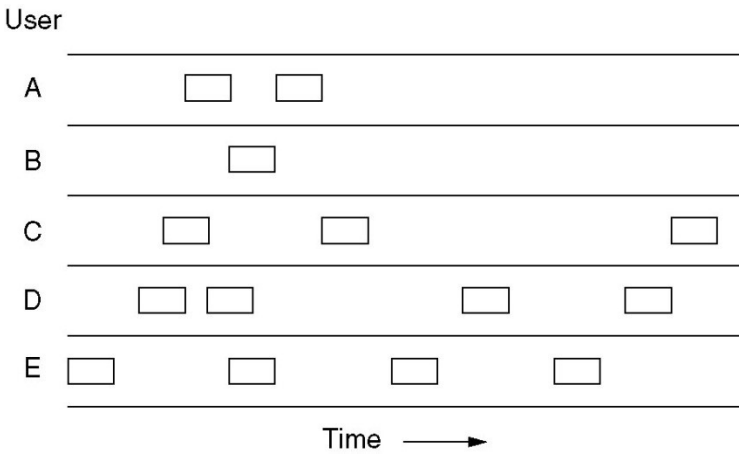
Contention Protocol Evolution

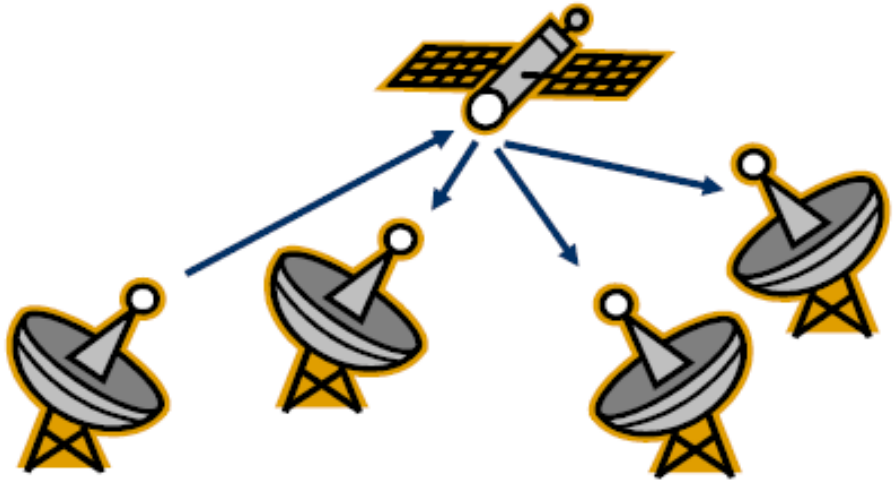
- ALOHA
  - Developed in the 70's for packet radio networks
  - Stations transmit data immedately
    - If there is a collision, it retransmits the packet later.
- Slotted ALOHA
  - Start transmissions only at fixed time slots
  - Significantly fewer collisions than ALOHA
- Carrier Sense Multiple Access (CSMA)
  - Start transmission only if the channel is idle
- CSMA / Collision Detection (CSMA/CD)
  - Stop ongoing transmission if collision is detected

Pure ALOHA

- The goal was to use low-cost commercial radio equipment to connect users on Oahu and the other Hawaiian islands with a central time-sharing computer on the main Oahu campus.

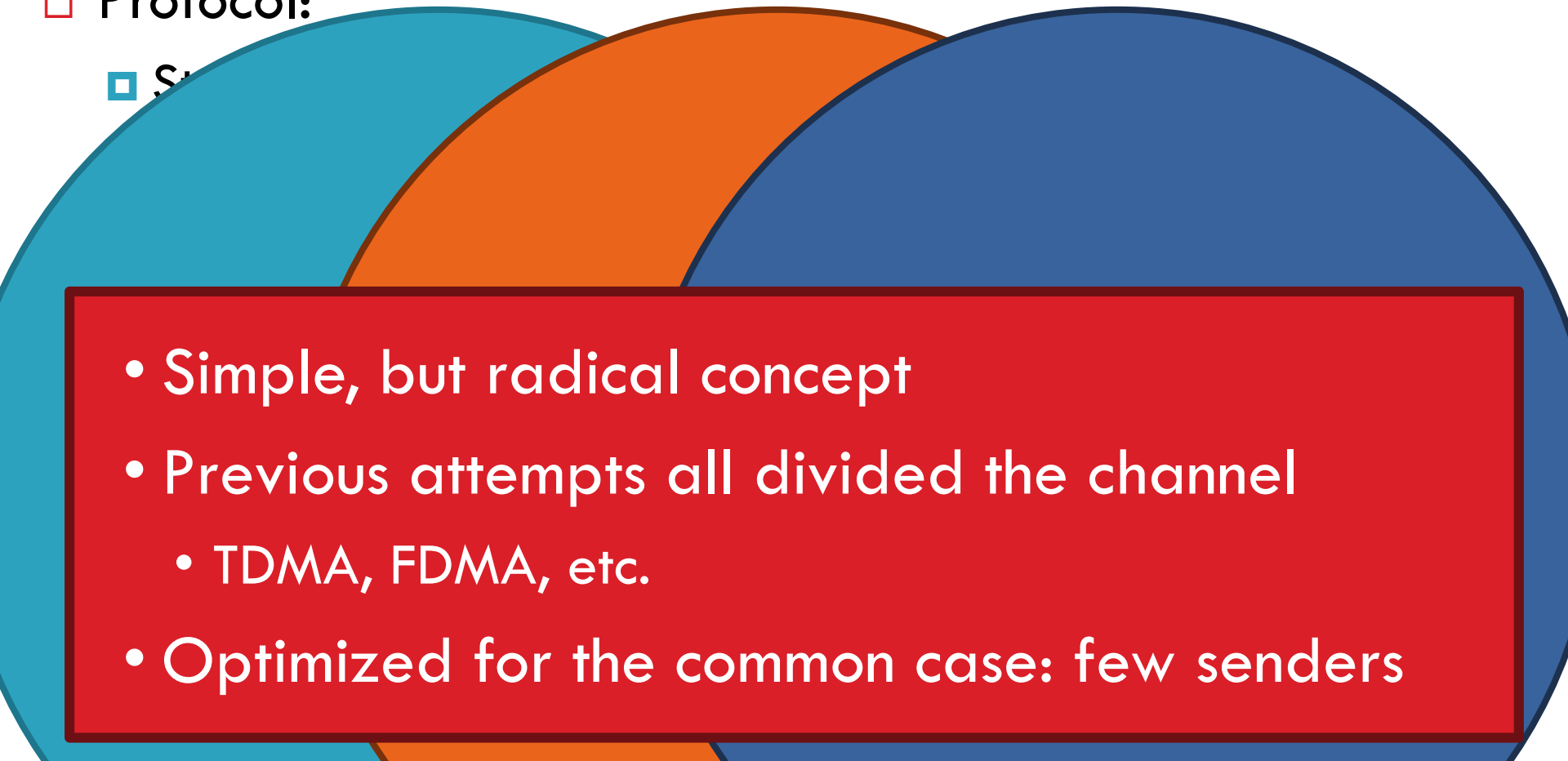
- Algorithm was developed by Uni. of Hawaii
  - **If you have data to send, send the data**
  - Low-cost and very simple





ALOHA

- Topology: radio broadcast with multiple stations
- Protocol:
  - S



- Simple, but radical concept
- Previous attempts all divided the channel
  - TDMA, FDMA, etc.
- Optimized for the common case: few senders

Performance analysis -Poisson Process

- The Poisson Process is a celebrated model used in Queuing Theory for "random arrivals". Assumptions leading to this model include:
  - The probability of an arrival during a short time interval Δt is proportional to the length of the interval, and does not depend on the origin of the time interval (memory-less property)
  - The probability of having multiple arrivals during a short time interval Δt approaches zero.

Performance analysis - Poisson Distribution

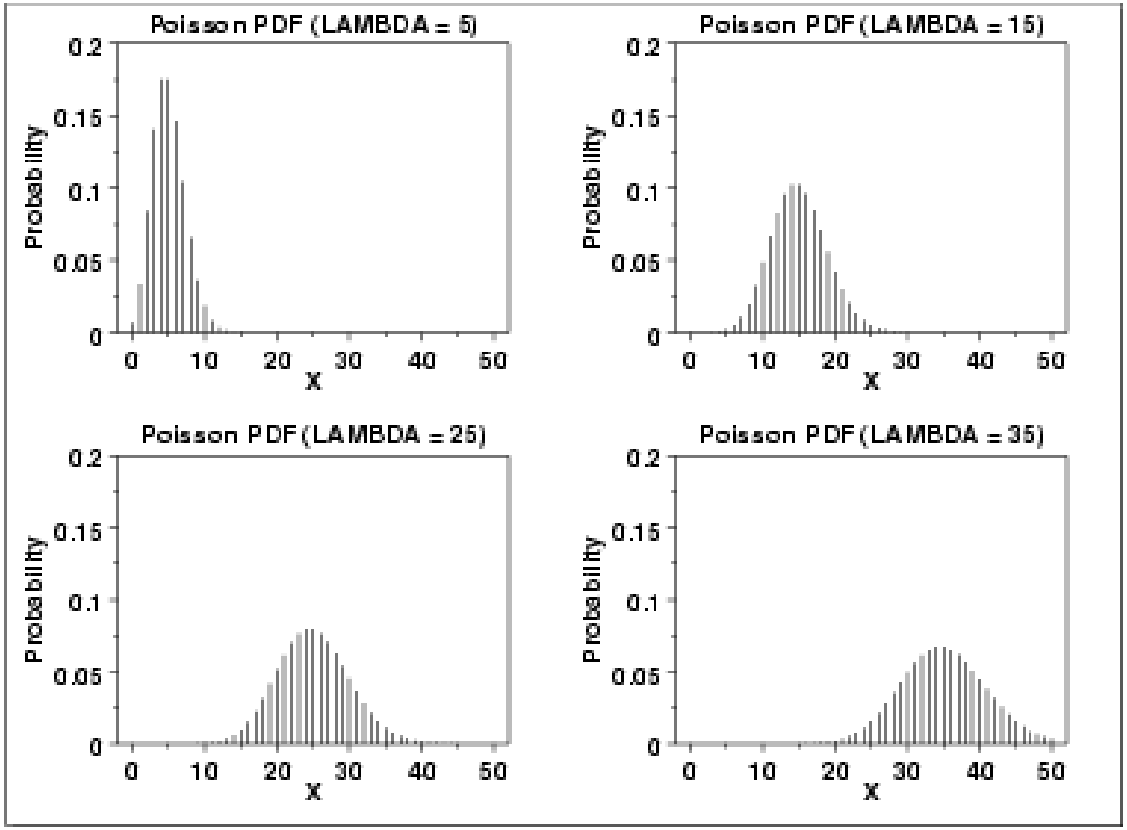
The probability of having k arrivals during a time interval of length t is given by:

$$P_k(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

where λ is the arrival rate. Note that this is a single-parameter model; all we have to know is λ.

FYI: Poisson Distribution

- The following is the plot of the Poisson probability density function for four values of λ.



Analysis of Pure ALOHA

- Notation:
  - T_f = frame time (processing, transmission, propagation)
  - *S*: Average number of successful transmissions per T_f ; that is, the *throughput*
  - *G*: Average number of total frames transmitted per T_f
  - *D*: Average delay between the time a packet is ready for transmission and the completion of successful transmission.
- We will make the following assumptions
  - All frames are of constant length
  - The channel is noise-free; the errors are only due to collisions.
  - Frames do not queue at individual stations
  - The channel acts as a Poisson process.

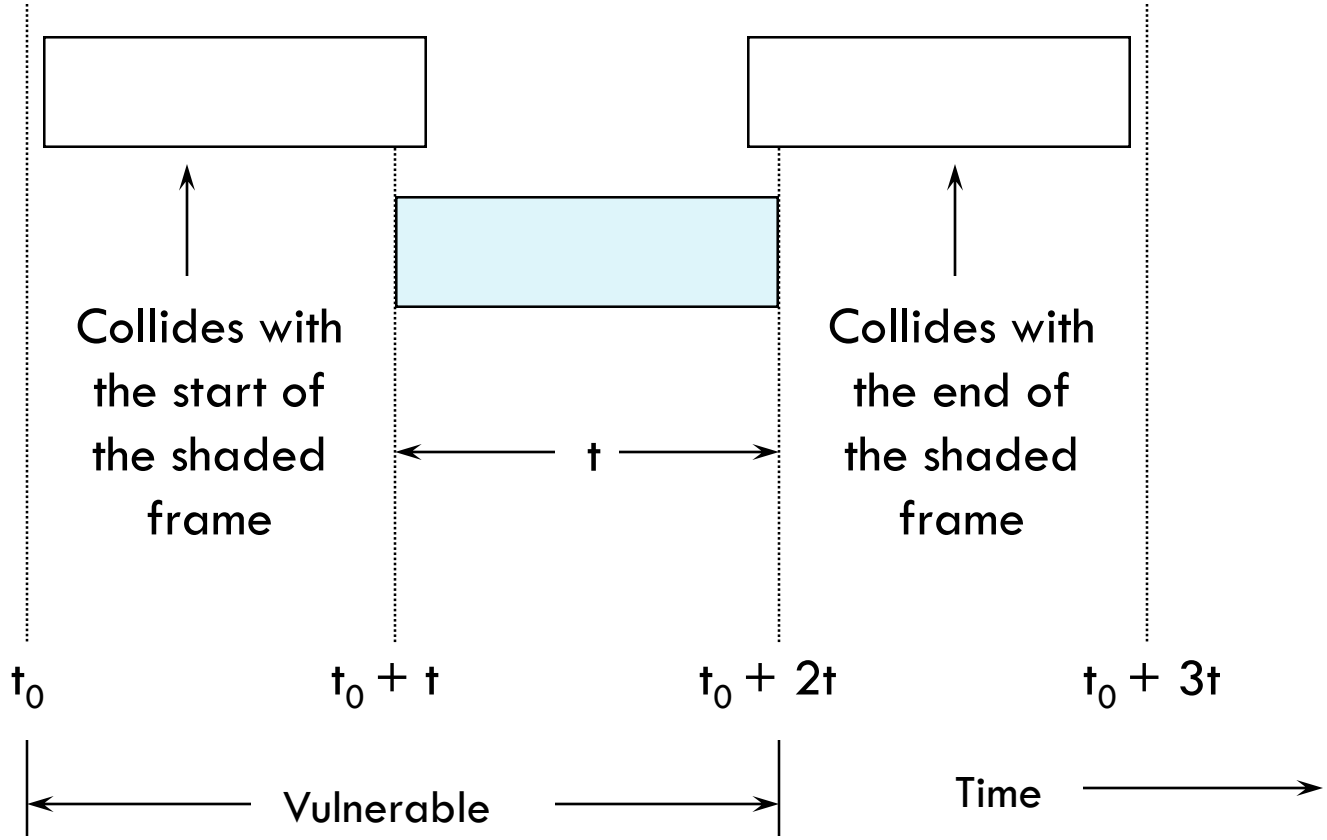
Analysis of Pure ALOHA…

- Since S represents the number of "good" transmissions per *frame time*, and G represents the total number of attempted transmissions per *frame time*, then we have:

$$S = G \times (\text{Probability of good transmission})$$

- The vulnerable time for a successful transmission is $2T_f$
- So, the probability of good transmission is not to have an "arrival" during the vulnerable time .

Analysis of Pure ALOHA…



Vulnerable period for the shaded frame

Analysis of Pure ALOHA…

Using:

$$P_k(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

And setting $t = 2\mathrm{T}_f$ and $k = 0$, we get

$$P_0(2T_f) = \frac{(\lambda \cdot 2T_f)^0 e^{-\lambda 2T_f}}{0!} = e^{-2G}$$

becasue $\lambda = \frac{G}{T_f}$. Thus, $S = G \cdot e^{-2G}$

Analysis of Pure ALOHA…

- If we differentiate $S = Ge^{-2G}$ with respect to G and set the result to 0 and solve for G, we find that the maximum occurs when

$$G = 0.5,$$

and for that $S = 1/2e = 0.18$. So, the maximum throughput is only 18% of capacity.

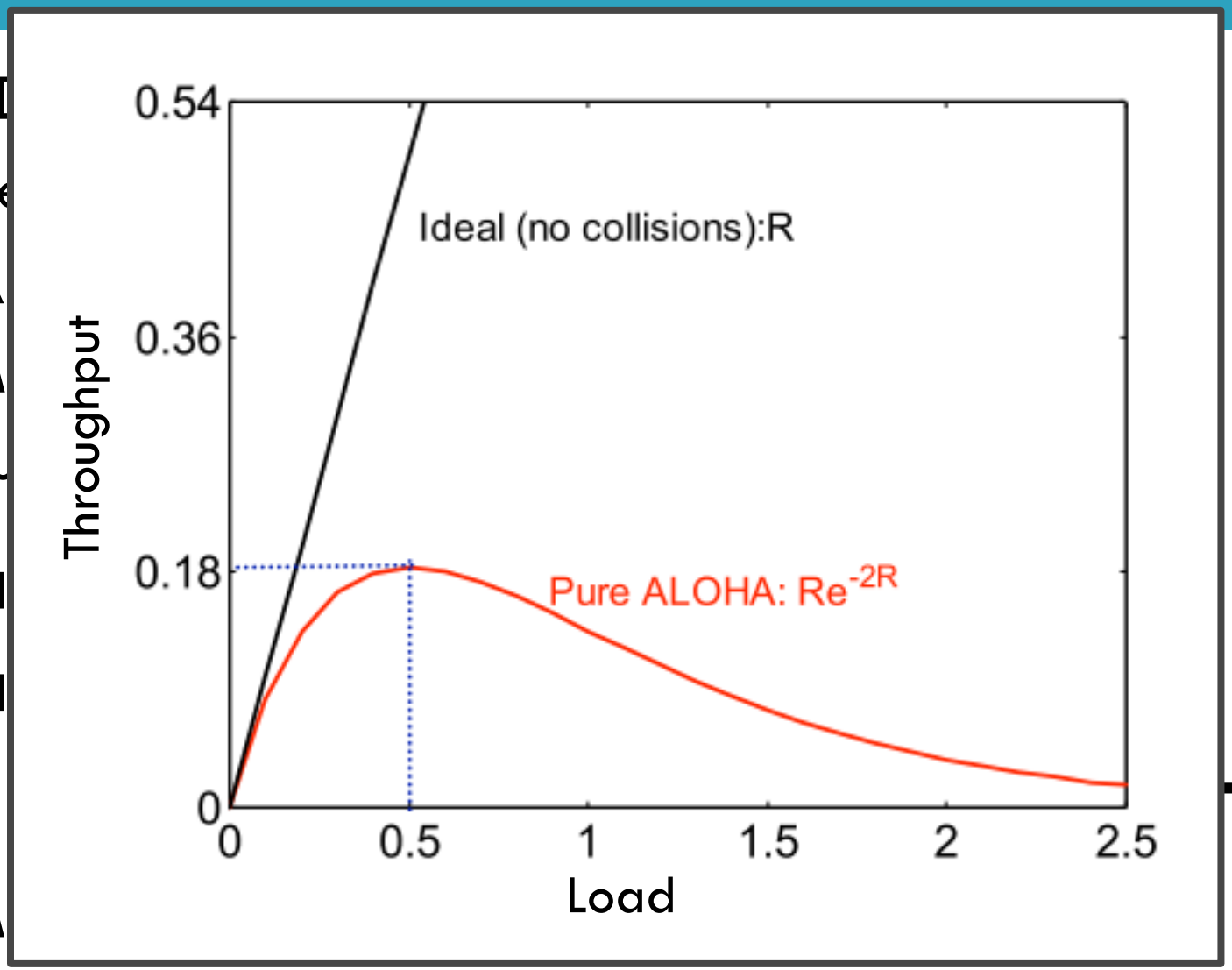
Tradeoffs vs. TDMA

- In TD
  - De
- In A
  - M
  - Bu

Send

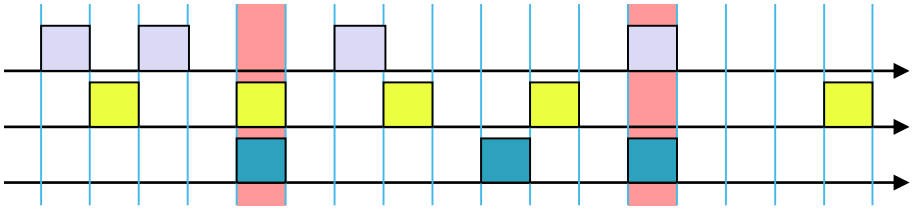
Send

  - M



Slotted ALOHA

- Channel is organized into uniform slots whose size equals the frame transmission time.
- Transmission is permitted only to begin at a slot boundary.



- Here is the procedure:
  - While there is a new frame A to send do

    Send frame A at (the next) slot boundary

Analysis of Slotted ALOHA

- Note that the vulnerable period is now reduced in half. Using:

$$P_k(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

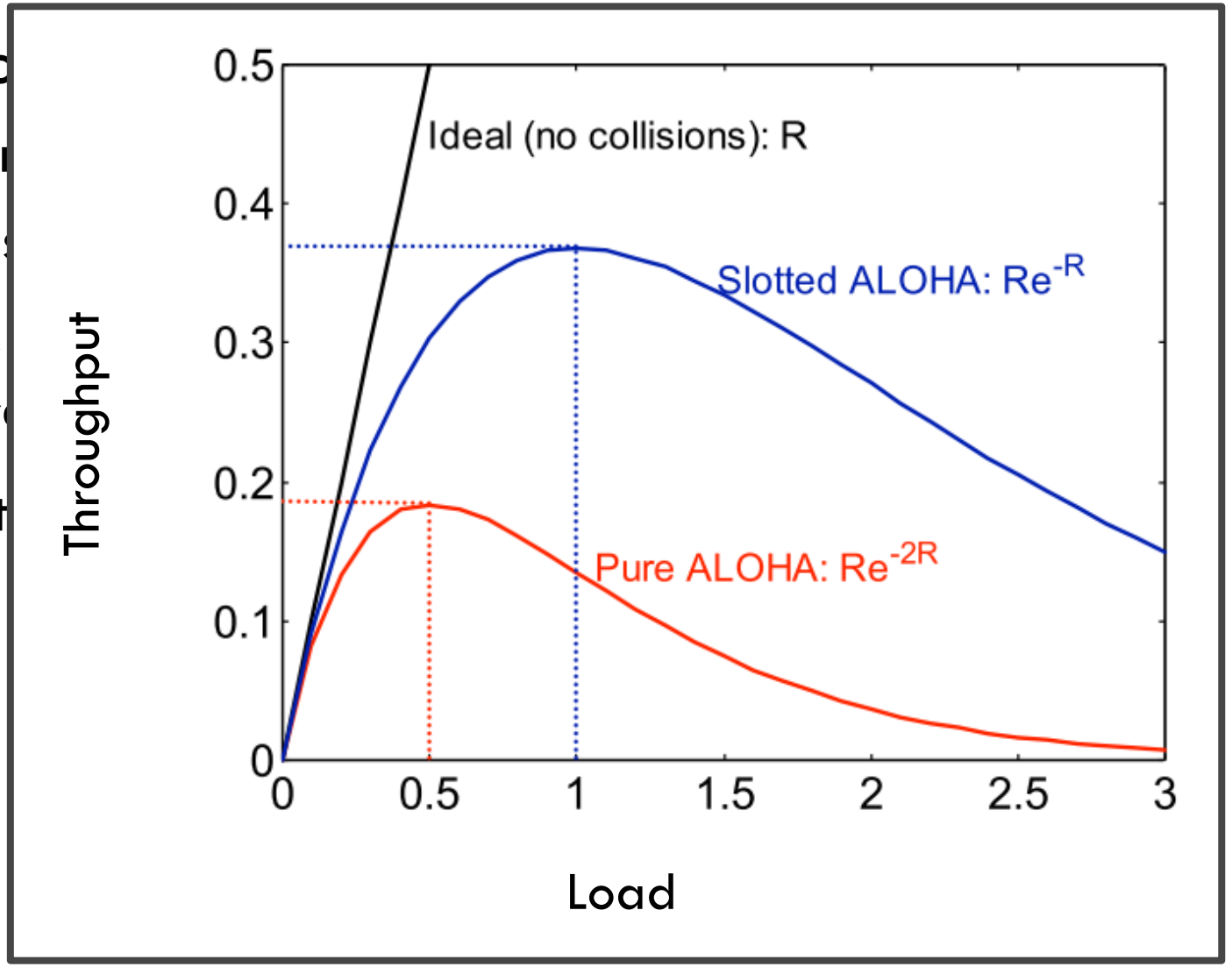
And setting $t = \mathrm{T}_f$ and $k = 0$, we get

$$P_0(T_f) = \frac{(\lambda \cdot T_f)^0 e^{-\lambda T_f}}{0!} = e^{-G}$$

because $\lambda = \frac{G}{T_f}$. Thus, $S = G \cdot e^{-G}$

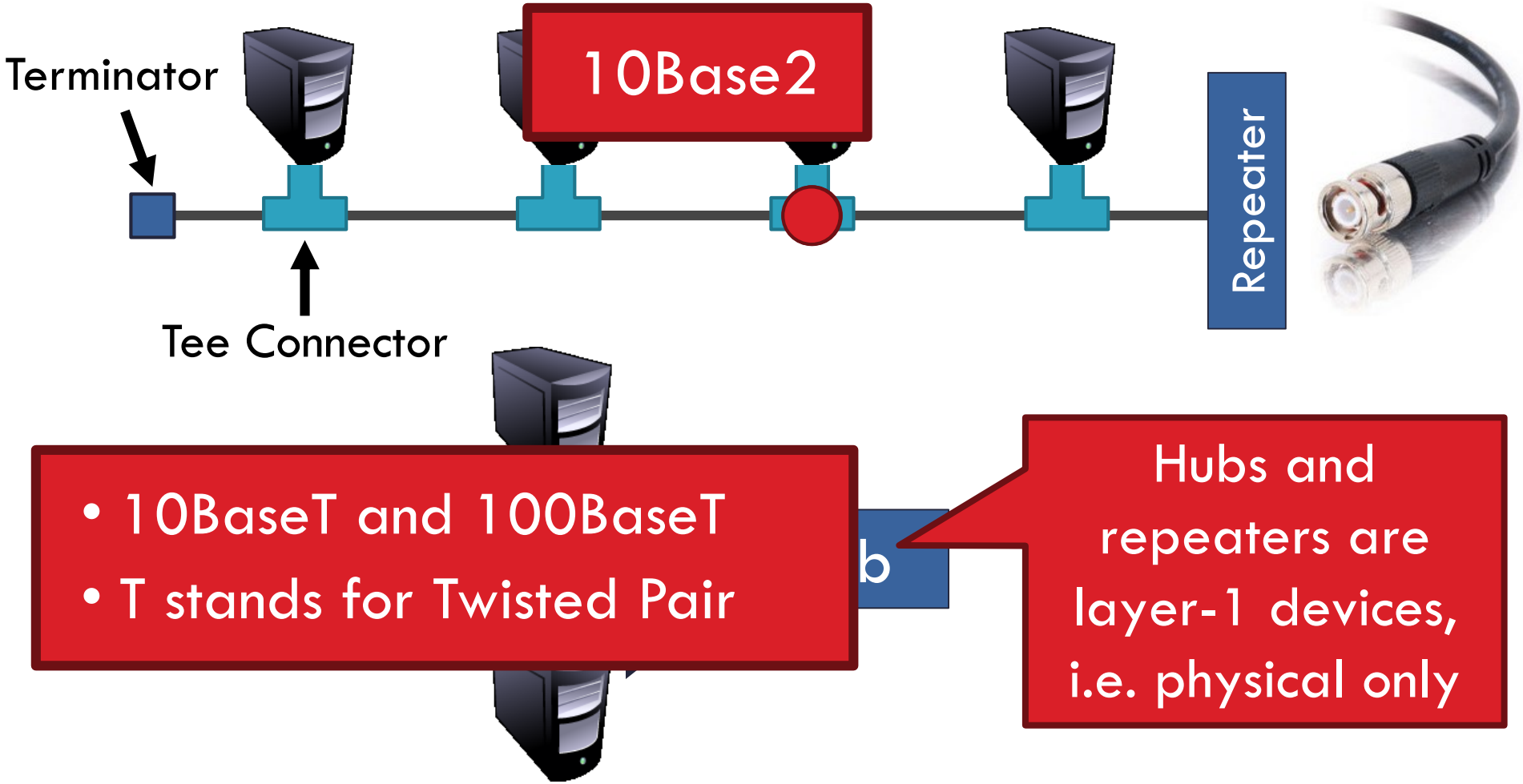
Slotted ALOHA

- Proto
  - Sa
  - Ho
- Thus,
  - 37
  - But



Broadcast Ethernet

- Originally, Ethernet was a broadcast technology



Carrier Sense Multiple Access (CSMA)

- Additional assumption:
  - Each station is capable of sensing the medium to determine if another transmission is underway

Non-persistent CSMA

While there is a new frame A to send

1. Check the medium
2. If the medium is busy, wait some time, and go to 1.
3. (medium idle) Send frame A

1-persistent CSMA

While there is a new frame A to send

1. Check the medium
2. If the medium is busy, go to 1.
3. (medium idle) Send frame A

p-persistent CSMA

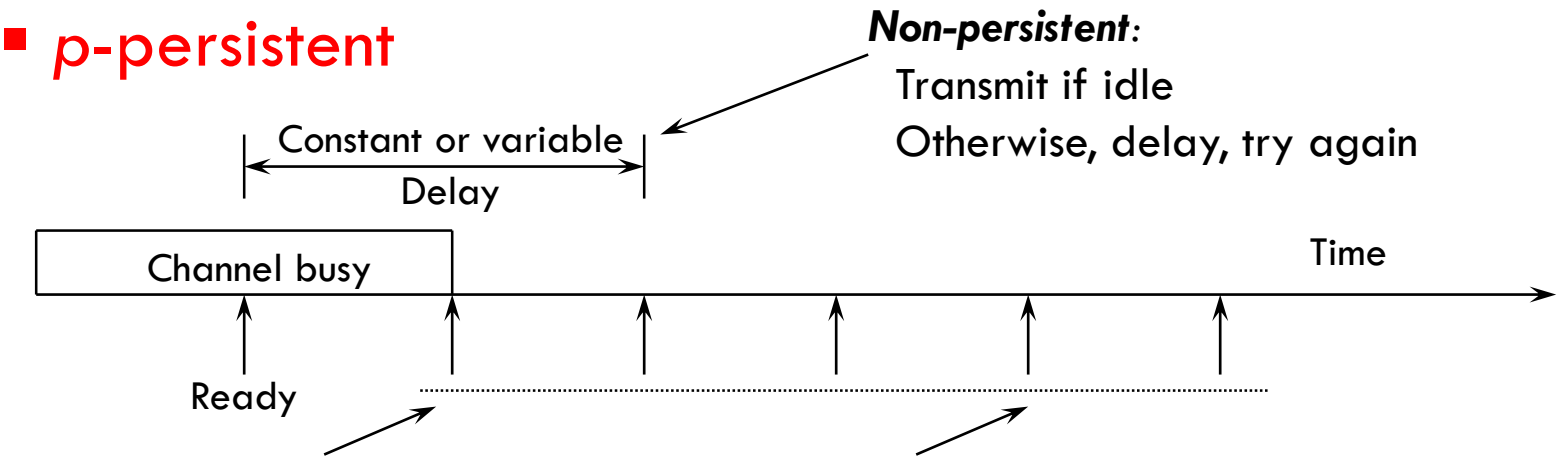
While there is a new frame A to send

1. Check the medium
2. If the medium is busy, go to 1.
3. (medium idle) With probability *p* send frame A, and probability (1- *p*) delay one time slot and go to 1.

CSMA Summary

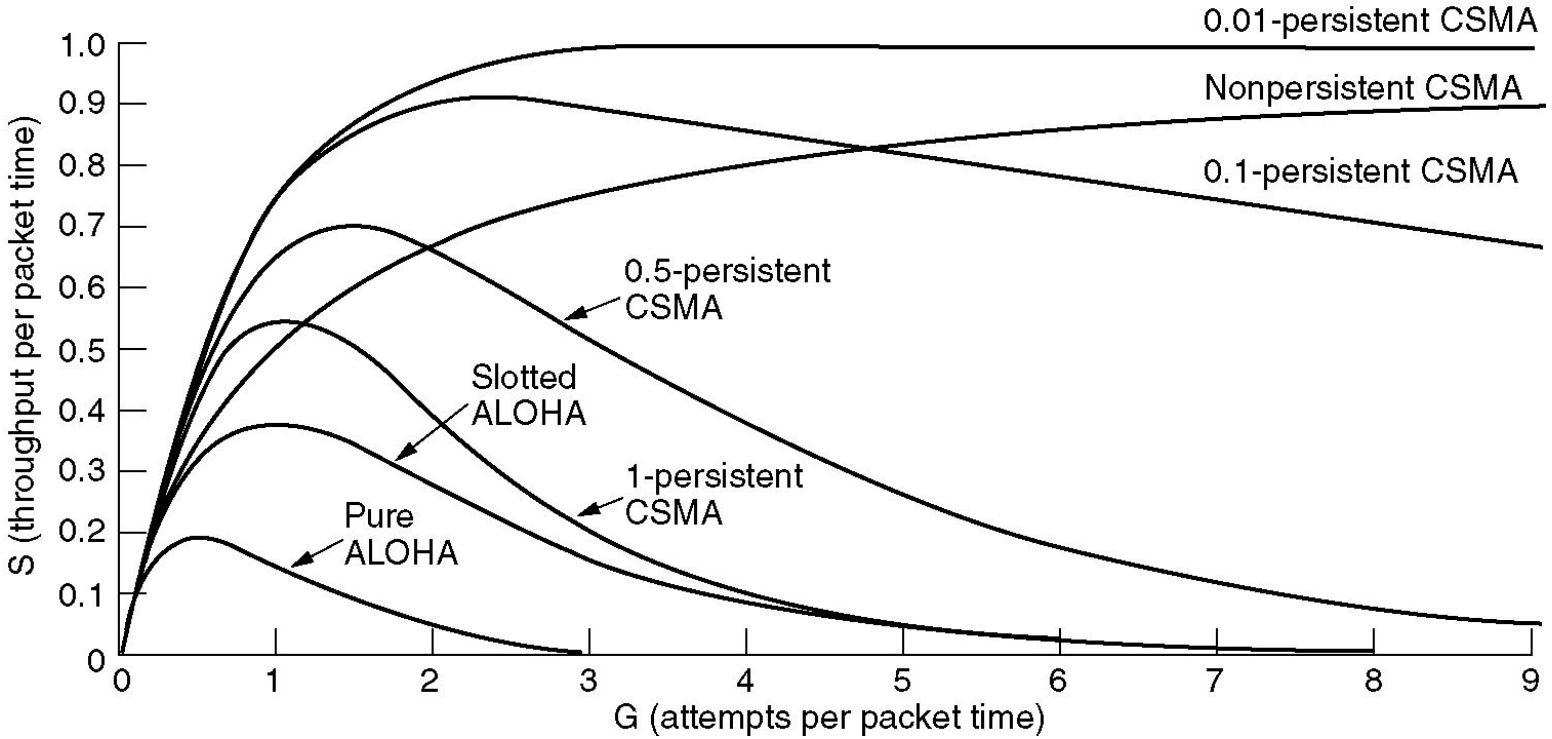
- Nonpersistent
- 1-persistent
- *p*-persistent

CSMA persistence and backoff



Persistent and Non-persistent CSMA

Comparison of throughput versus load for various random access protocols.



CSMA with Collision Detection

- Stations can sense the medium while transmitting
- A station aborts its transmission if it senses another transmission is also happening (that is, it detects collision)
- Question: When can a station be sure that it has *seized* the channel?
  - Minimum time to detect collision is the time it takes for a signal to traverse between two farthest apart stations.

CSMA/CD

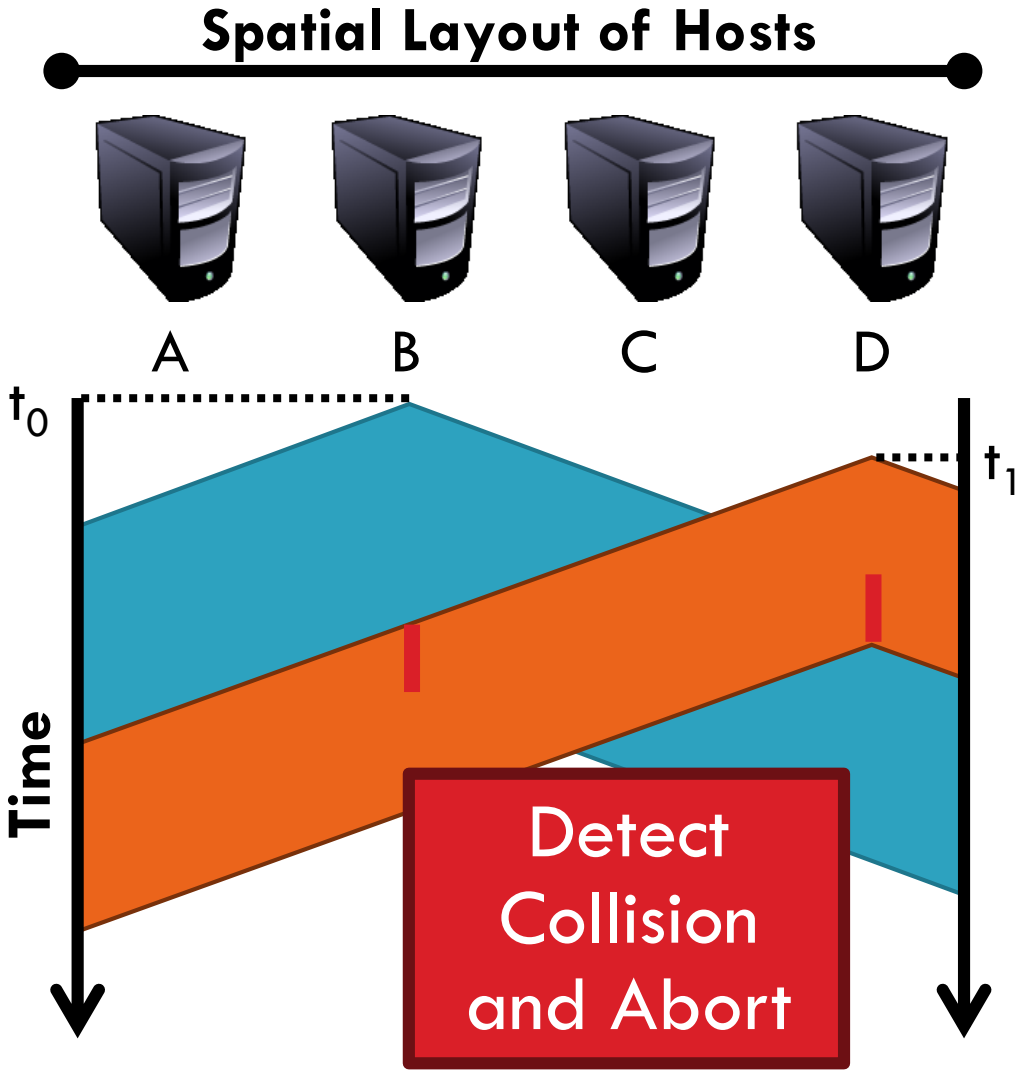
- A station is said to *seize* the channel if all the other stations become aware of its transmission.
- There has to be a lower bound on the length of each frame for the *collision detection* feature to work out.
- Ethernet uses CSMA/CD

CSMA/CD

- Carrier sense multiple access with collision detection
- Key insight: wired protocol allows us to sense the medium
- Algorithm
  1. Sense for carrier
  2. If carrier is present, wait for it to end
     - Sending would cause a collision and waste time
  3. Send a frame and sense for collision
  4. If no collision, then frame has been delivered
  5. If collision, abort immediately
     - Why keep sending if the frame is already corrupted?
  6. Perform exponential backoff then retransmit

CSMA/CD Collisions

- Collisions can occur
- Collisions are quickly detected and aborted
- Note the role of distance, propagation delay, and frame length



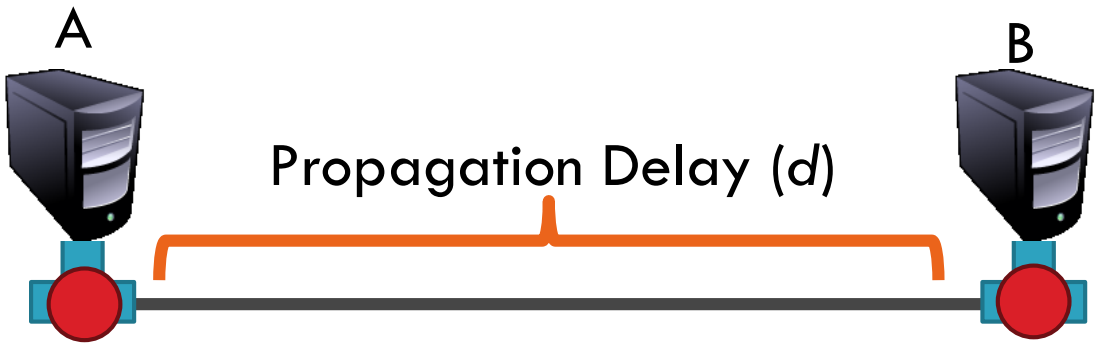
Exponential Backoff

- When a sender detects a collision, send "jam signal"
  - Make sure all hosts are aware of collision
  - Jam signal is 32 bits long (plus header overhead)
- Exponential backoff operates:
  - Select $k \in [0, 2^n - 1]$ unif. rnd., where n = number of collisions
  - Wait k time units (packet times) before retransmission
  - n is capped at 10, frame dropped after 16 collisions
- Backoff time is divided into contention slots

Minimum Packet Sizes

- Why is the minimum packet size 64 bytes?
  - To give hosts enough time to detect collisions
- What is the relationship between packet size and cable length?

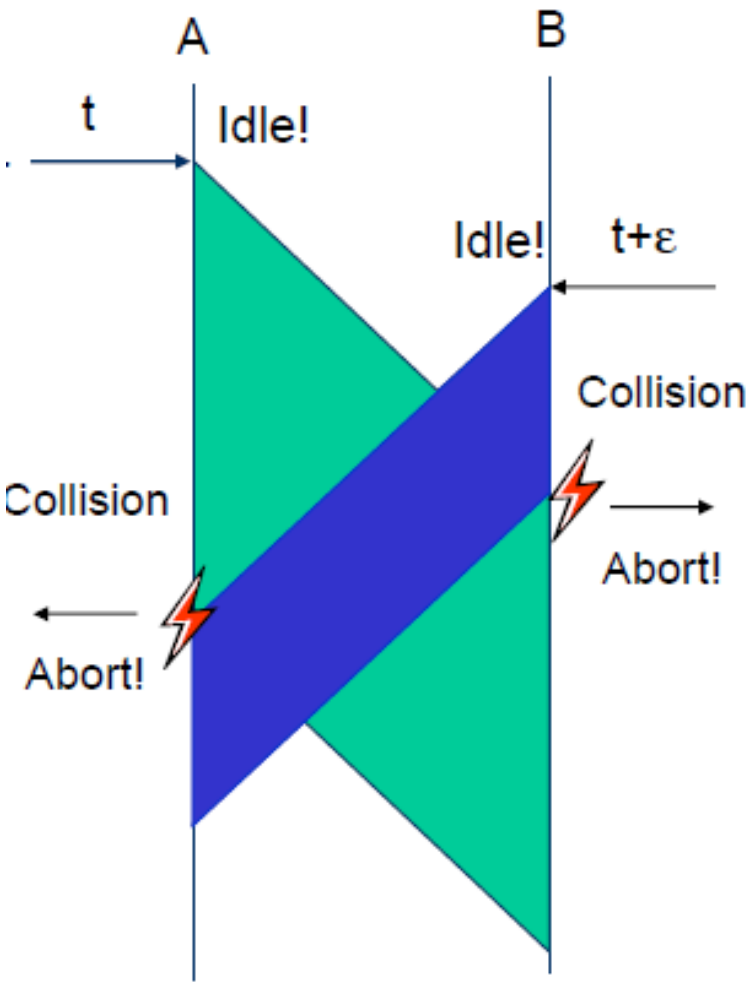
1. Time *t*: Host A starts transmitting
2. Time *t* + *d*: Host B starts transmitting
3. Time *t* + 2**d*: collision detected



Basic idea: Host A must be transmitting at time 2*d!

CSMA/CD

- CSMA/CD can be in one of three states: contention, transmission, or idle.
- To detect all the collisions we need
  - $T_f \geq 2T_{pg}$
  - where T_f is the time needed to send the frame
  - And T_{pg} is the propagation delay between A and B



Minimum Packet Size

- Host A must be transmitting after 2*d time units
  - Min_pkt = rate (b/s) * 2 * d (s)
    - … but wha
    - Propagati
  - This gives:
  - Min_pkt =
- So cable length is equ
  - Dist = min_pkt * light sp /(2 * rate)

- 10 Mbps Ethernet
- Packet and cable lengths change for faster Ethernet standards

(64B*8)*(2.5*10^{8}mps)/(2*10^{7}bps) = 6400 meters

Cable Length Examples

min_frame_size*light_speed/(2*bandwidth) = max_cable_length

$(64B*8)*(2.5*10^8 mps)/(2*10Mbps) = 6400$ meters

- What is the max cable length if min packet size were changed to 1024 bytes?
  - 102.4 kilometers
- What is max cable length if bandwidth were changed to 1 Gbps ?
  - 64 meters
- What if you changed min packet size to 1024 bytes and bandwidth to 1 Gbps?
  - 1024 meters

Maximum Packet Size

- Maximum Transmission Unit (MTU): 1500 bytes
- Pros:
  - Bit errors in long packets incur significant recovery penalty
- Cons:
  - More bytes wasted on header information
  - Higher per packet processing overhead
- Datacenters shifting towards Jumbo Frames
  - 9000 bytes per packet

Long Live Ethernet

- Today's Ethernet is switched
  - More on this later
- 1Gbit and 10Gbit Ethernet now common
  - 100Gbit on the way
  - Uses same old packet header
  - Full duplex (send and receive at the same time)
  - Auto negotiating (backwards compatibility)
  - Can also carry power