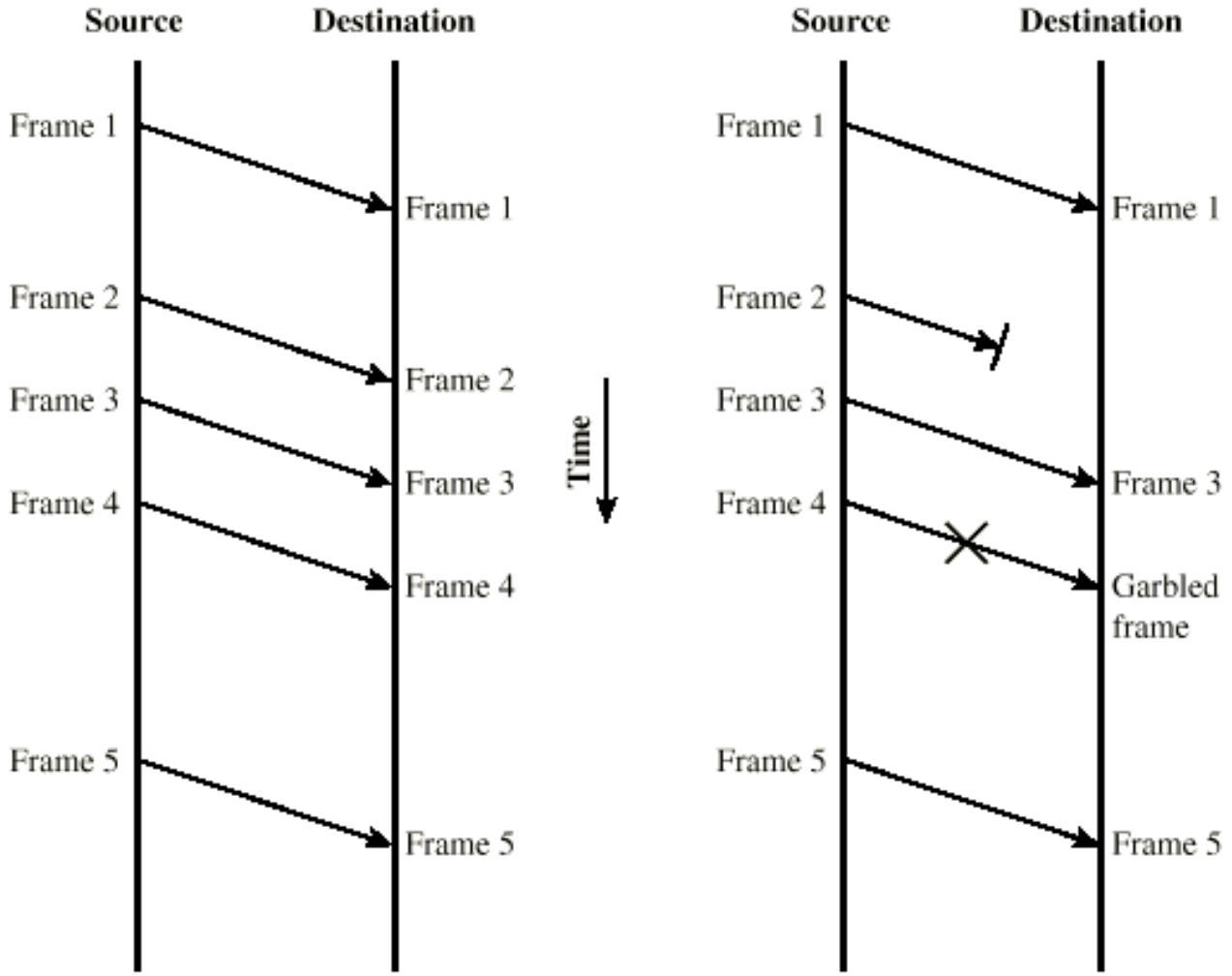
Data Link Layer Protocols

Flow Control

- Ensuring the sending entity does not overwhelm the receiving entity
  - Preventing buffer overflow
- Transmission time
  - Time taken to emit all bits into medium
- Propagation time
  - Time for a bit to traverse the link

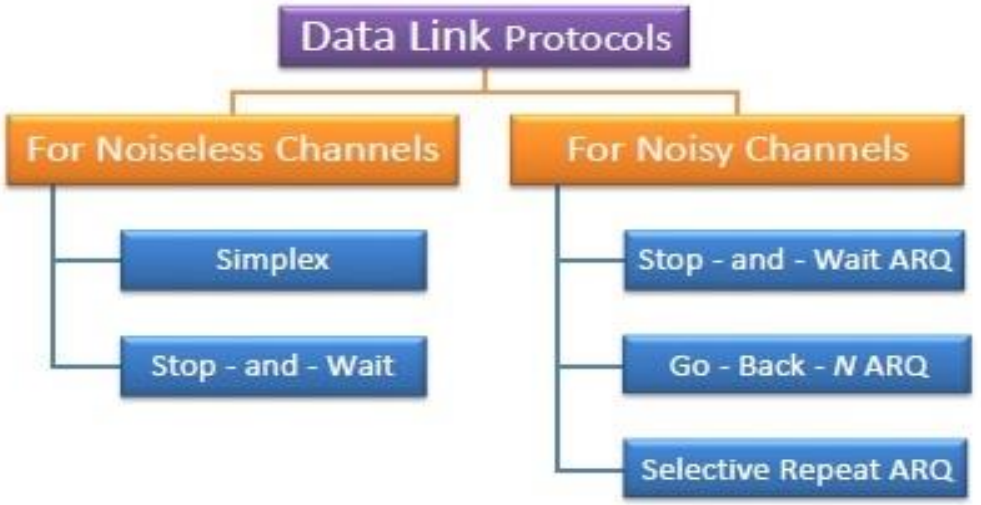
Model of Frame Transmission



(a) Error-free transmission

(b) Transmission with losses and errors

Data link protocols can be broadly divided into two categories, depending on whether the transmission channel is noiseless or noisy.



Initial Assumptions

- Simplex Channel
- Infinite buffer capacity with the receiver
- Error free transmission
- Network layer at the senders end is always ready with data
- No need for flow control

Unrestricted Simplex Protocol – the Utopia

- Both the transmitting and receiving network layers are always ready
- Framing only
- No error or flow control
  - Infinite buffer space is available
  - The communication channel between the data link layers never damages or loses frames
- Data are transmitted in one direction only
- Processing time can be ignored

Unrestricted Simplex Protocol

```
void sender1 (void)
{
    frame s;
    packet buffer;
    while (true) {
    from_network_layer(&buffer);
    s.info = buffer;
    to_physical_layer(&s);
                    }
}
```

```
void receiver1 (void)
{
    frame r;
    event_type event;
    while (true) {
    wait_for_event(&event);
    from_physical_layer(&r);
    to_network_layer(&r.info);
                    }
}
```

Simplex Protocol

- The Simplex protocol is hypothetical protocol designed for unidirectional data transmission over an ideal channel, i.e. a channel through which transmission can never go wrong.
- It has distinct procedures for sender and receiver.
- The sender simply sends all its data available onto the channel as soon as they are available its buffer.
- The receiver is assumed to process all incoming data instantly.
- It is hypothetical since it does not handle flow control or error control.

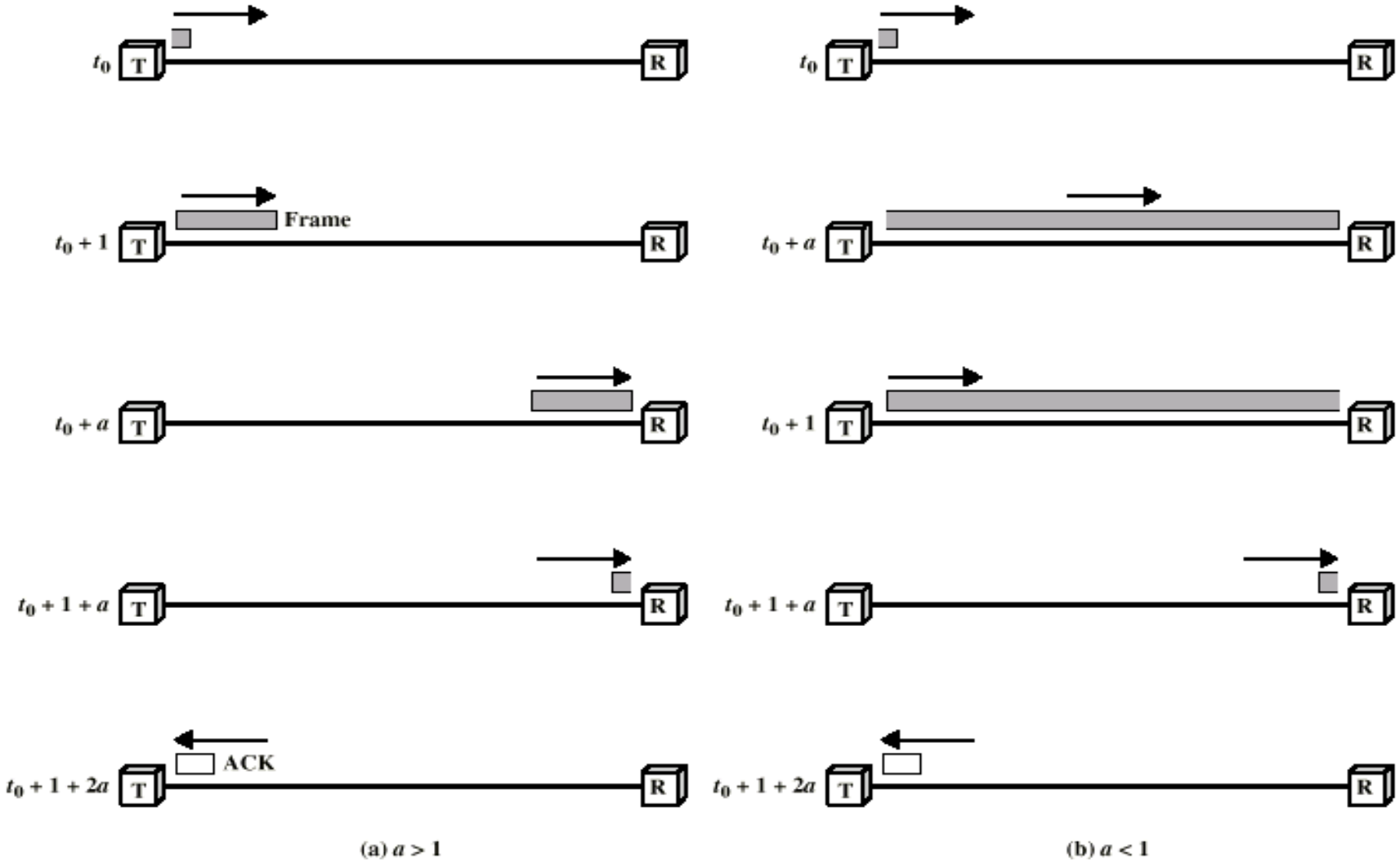
Simplex Stop-and-Wait Protocol

- Assumptions
- Finite buffer
- The communication channel between the data link layers never damages or loses frames
- Data are transmitted in one direction only
- Processing time can be ignored

Stop and Wait

- Send one packet & Wait for Ack before proceeding
  - Source transmits frame
  - Destination receives frame and replies with acknowledgement
  - Source waits for ACK before sending next frame
  - Destination can stop flow by not send ACK
  - Works well for a few large frames

Stop and Wait Link Utilization



Simplex Stop-and-Wait Protocol

```
void sender2 (void)
{
      frame s;
      packet buffer;
      event_type event;
      while (true) {
      from_network_layer(&buffer);
      s.info = buffer;
      to_physical_layer(&s);
      wait_for_event(&event);
                        }
}
```

```
void receiver2 (void)
{
    frame r, s;
    event_type event;
    while (true) {
    wait_for_event(&event);
    from_physical_layer(&r);
    to_network_layer(&r.info);
    to_physical_layer(&s);
                    }
}
```

Stop – and – Wait Protocol

- Stop – and – Wait protocol is for noiseless channel.
- It provides unidirectional data transmission without any error control facilities.
- However, it provides for flow control so that a fast sender does not drown a slow receiver.
- The receiver has a finite buffer size with finite processing speed.
- The sender can send a frame only when it has received indication from the receiver that it is available for further data processing.

A Simplex Protocol for a Noisy channel

Assumptions

- Finite Buffer
- A positive acknowledgement with retransmission protocol.
- Data are transmitted in one direction only
- Processing time can be ignored

Automatic Repeat request (ARQ) protocols

- Positive Ack
- 1-bit sequence number in frames (not in acks)
- Timeout to detect lost frames/acks
- Retransmission
- Can fail under early timeout conditions
- Full Duplex Communication
- Piggybacking of acks
  - Temporarily delaying outgoing acknowledgements so that they can be hooked onto the next outgoing data frame

Stop – and – Wait ARQ

- Stop – and – wait Automatic Repeat Request (Stop – and – Wait ARQ) is a variation of the Stop –and- wait protocol with added error control mechanisms, appropriate for noisy channels.
- The sender keeps a copy of the sent frame.
- It then waits for a finite time to receive a positive acknowledgement from receiver.
- If the timer expires or a negative acknowledgement is received, the frame is retransmitted.
- If a positive acknowledgement is received then the next frame is sent.

Fragmentation

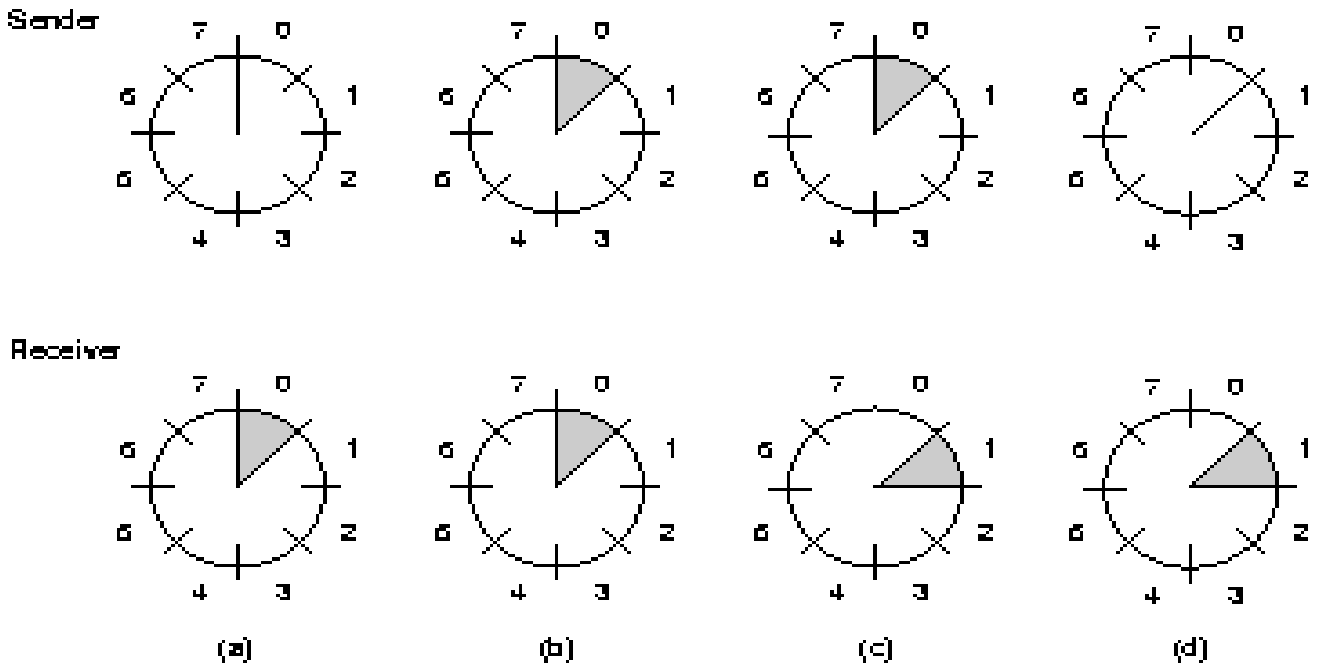
- Large block of data may be split into small frames
  - Limited buffer size
  - Errors detected sooner (when whole frame received)
  - On error, retransmission of smaller frames is needed
  - Prevents one station occupying medium for long periods
- Stop and wait becomes inadequate

Sliding Windows Flow Control

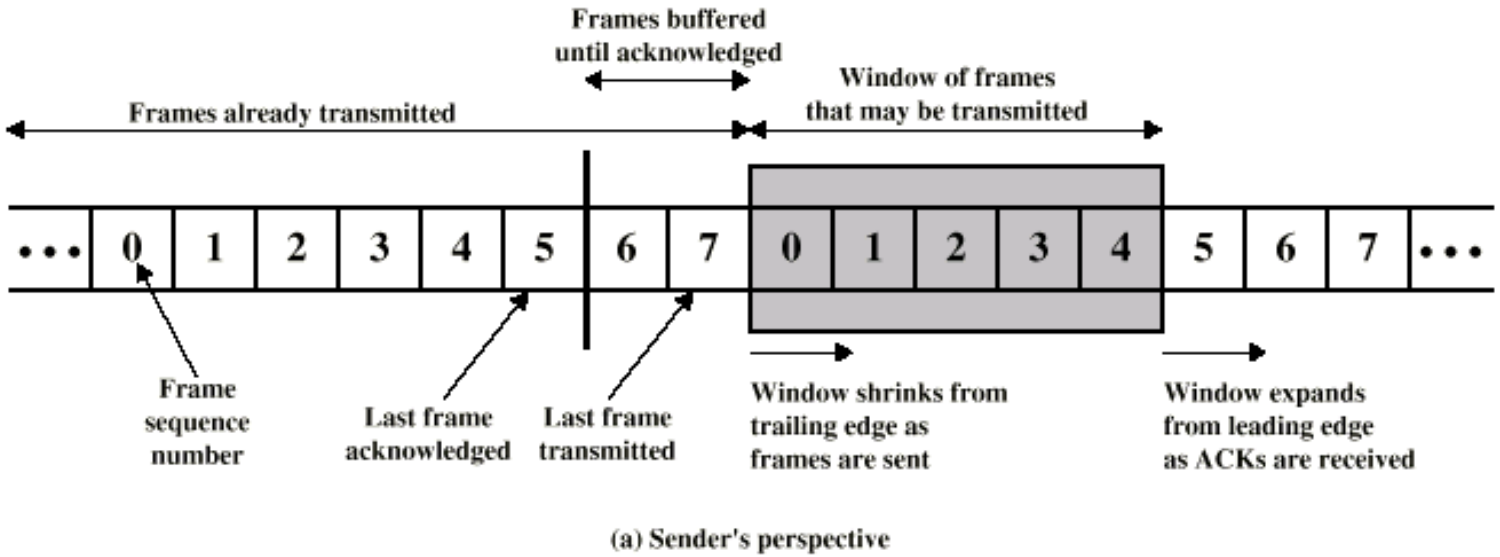
- Allow multiple frames to be in transit
- Receiver has buffer W long
- Transmitter can send up to W frames without ACK
- Each frame is numbered
- ACK includes number of next frame expected
- Sequence number bounded by size of field (k)
  - Frames are numbered modulo 2^k

Sliding Window Protocols

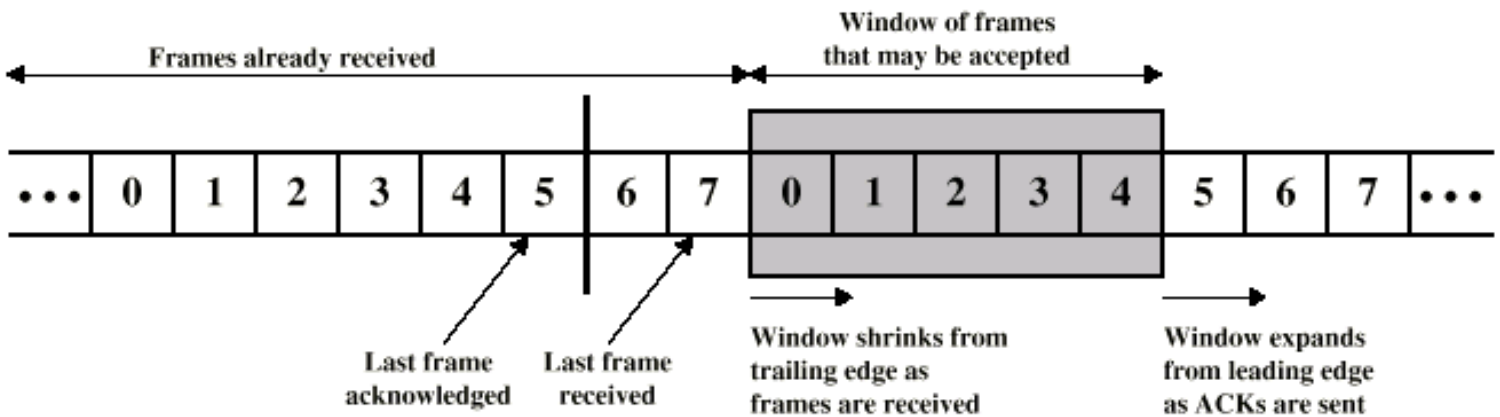
- Window = Set of sequence numbers to send/receive
- Sender window
  - Sender window increases when ack received
  - Packets in sender window must be buffered at source
  - Sender window may grow in some protocols
- Receiver window
  - Packets outside window discarded
  - Window advances when sequence number = low edge of window received
  - Receiver window always constant
  - Sender transmits W frames before blocking (pipelining)


Sender
Receiver
7
0
1
2
3
4
5
6
(a)
(b)
(c)
(d)

Sliding Window Diagram

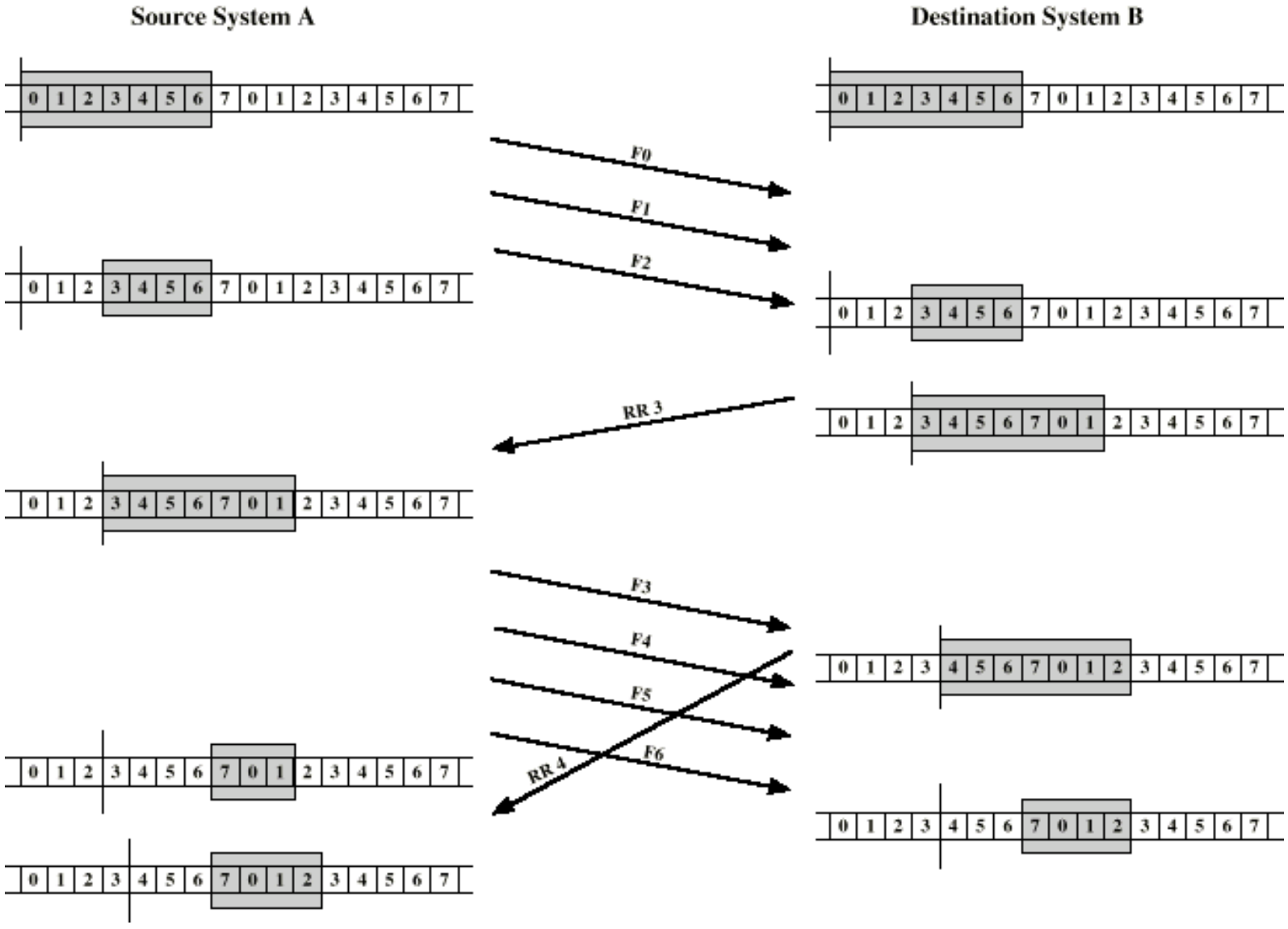


(a) Sender's perspective



(b) Receiver's perspective

Example Sliding Window



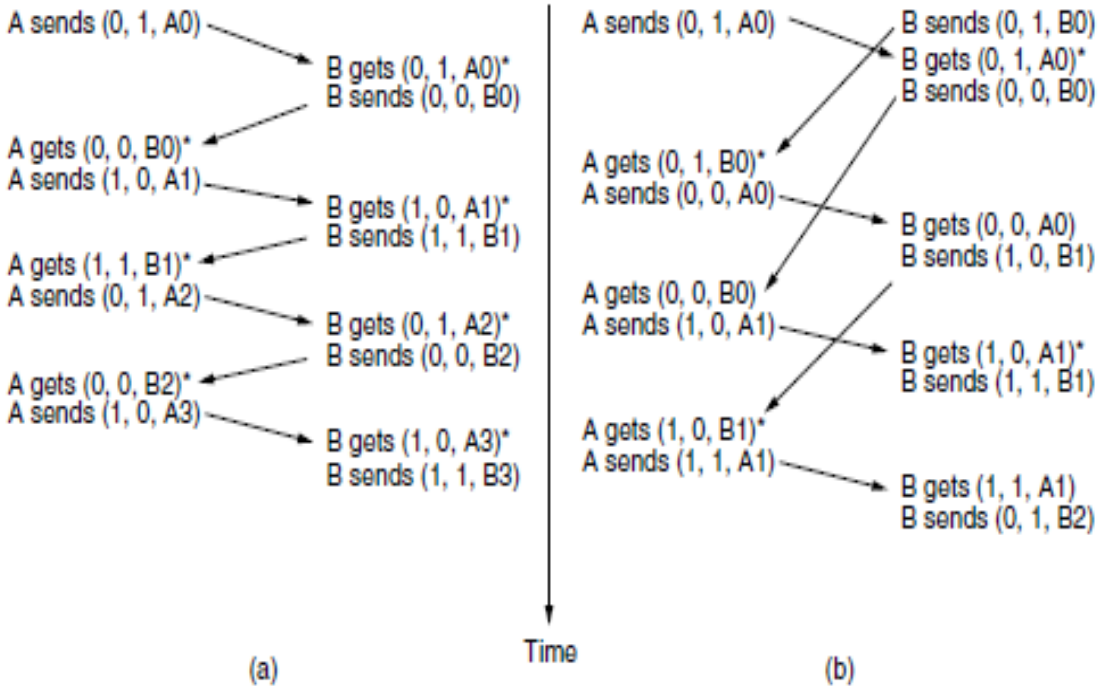
Sliding Window Protocols

- A One-Bit Sliding Window Protocol
- A Protocol Using Go Back N
- A Protocol Using Selective Repeat

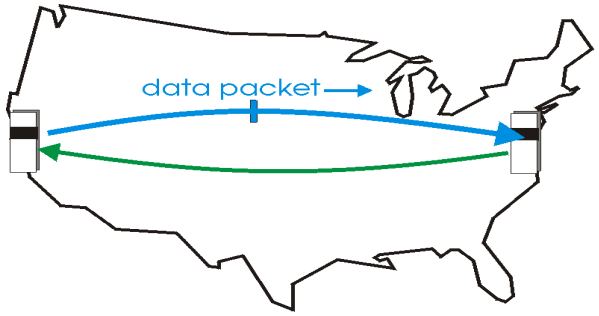
One-Bit Sliding Window

- Stop and Wait + error control
- Piggybacked acks
- Acks sent instantly
- Packets may be resent to accommodate immediate piggybacking
- Acks contain sequence number
- Peculiar case: when both sides start simultaneously

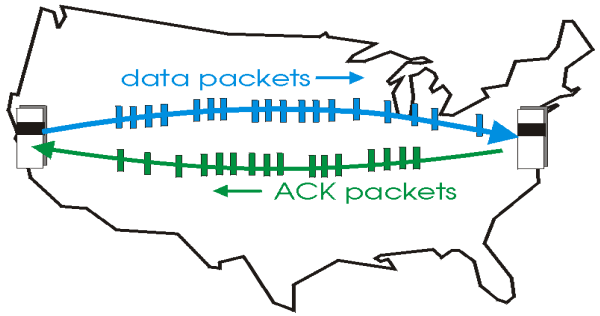
Two scenarios for protocol 4. (a) Normal case.
(b) Abnormal case.
The notation is (seq, ack, packet number). An asterisk indicates where a network layer accepts a packet.



- Sliding Window Protocol(Duplex)
  - At any instant of time, the sender & receiver maintain a set of sequence numbers corresponding to frames it is permitted to send/receive
  - Piggybacking
  - Pipelining

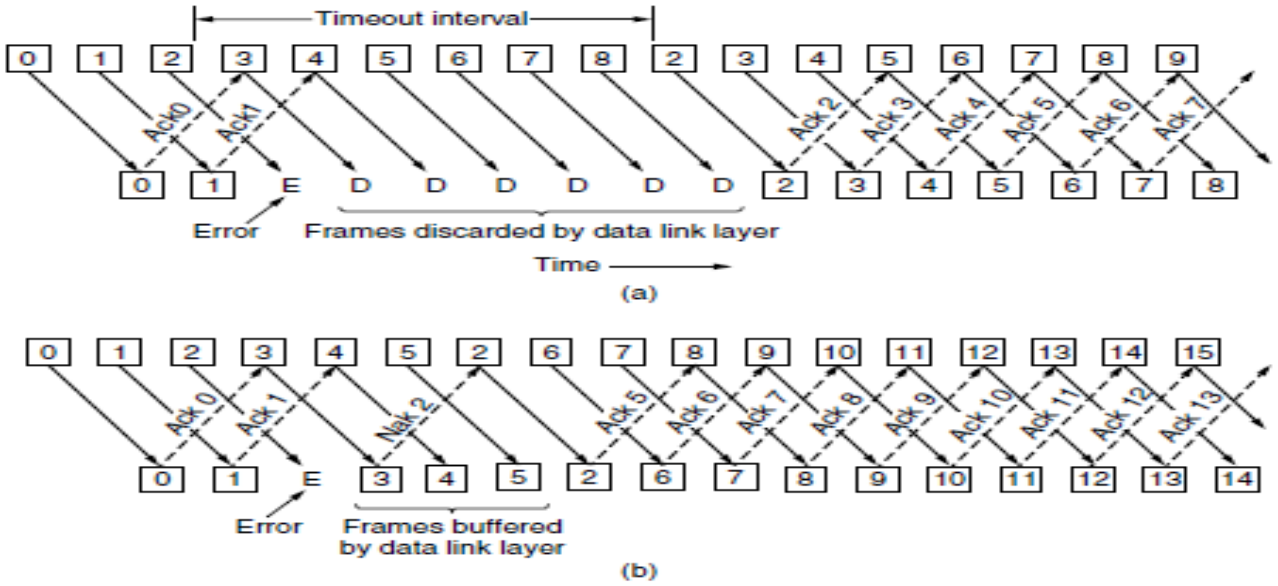


(a) a stop-and-wait protocol in operation



(b) a pipelined protocol in operation

Pipelining and error recovery. Effect of an error when (a) receiver's window size is 1 and (b) receiver's window size is large.



Go – Back – N ARQ

- Go – Back – N ARQ provides for sending multiple frames before receiving the acknowledgement for the first frame.
- It uses the concept of sliding window, and so is also called sliding window protocol.
- The frames are sequentially numbered and a finite number of frames are sent.
- If the acknowledgement of a frame is not received within the time period, all frames starting from that frame are retransmitted.

- Round trip propagation + transmission time not negligible, Stop and Wait is not efficient
- Sender is allowed to transmit multiple packets without waiting for an acknowledgement, but is constrained to have no more than some maximum allowable number (N)
- Use cumulative acknowledgement
- Ack the latest packet (N) if all packets (< N) are received.
- Discard out-of-order packets, no receiver buffering
  - Sliding Window + Retransmit all packets starting from earliest unacked packet
- Can waste a lot of bandwidth if error rate high

Selective Repeat ARQ

- This protocol also provides for sending multiple frames before receiving the acknowledgement for the first frame.
- However, here only the erroneous or lost frames are retransmitted, while the good frames are received and buffered.

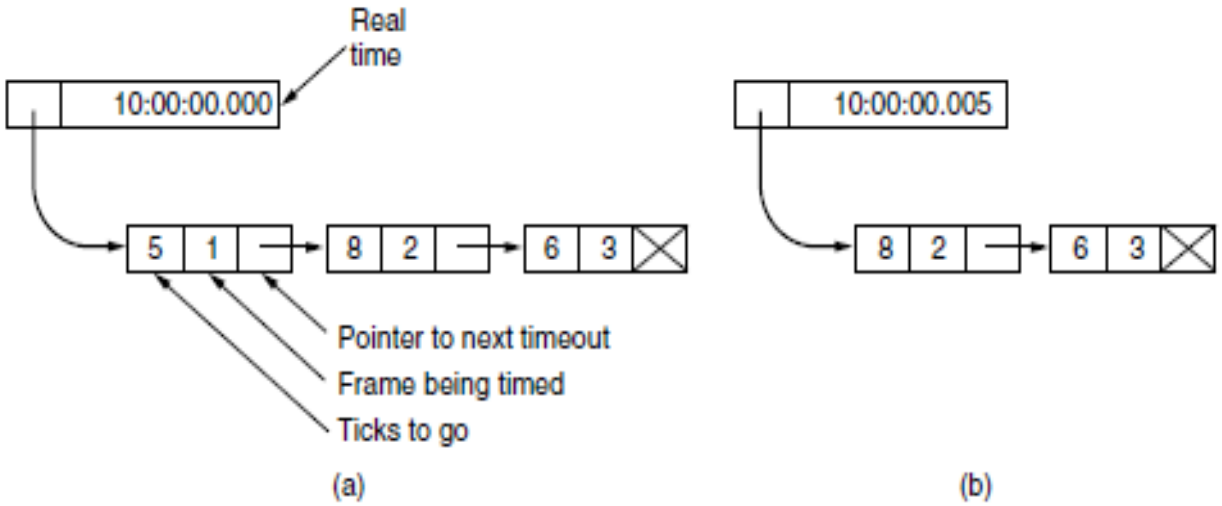
Selective Repeat

- Sliding Window + Retransmit just the bad frame when Nak received
- Out-of-order buffering required at destination
- Loses time (and hence throughput) if bursty loss of frames
- Acks not cumulative.
- Non-sequential arrival of packets
  - ensure no overlap with old window when receiver advances window
  - Maximum Window = (Sequence Number Space)/2

Out-of-order buffering not mandatory at destination

- Maximum Window = Sequence Number Space - 1
- N timers or 1 timer may be used

Simulation of multiple timers in software. (a) The queued timeouts. (b) The situation after the first timeout has expired.



Enhancement to piggybacking

- Auxiliary Ack timer
- Set timer when packet to be acked is received
- If no reverse data, generate Ack when timer goes off
- Send negative ack (Nak) when out-of-order frame received
- Source retransmits Nak’ed frame
- Naks not cumulative
- Do not send duplicate Nak

Sliding Window Enhancements

- Receiver can acknowledge frames without permitting further transmission (Receive Not Ready)
- Must send a normal acknowledge to resume
- If duplex, use piggybacking
  - If no data to send, use acknowledgement frame
  - If data but no acknowledgement to send, send last acknowledgement number again, or have ACK valid flag (TCP)