| CS 4/54201 Computer Communication Network | Kent State University Dept. of Computer Science [www.cs.kent.edu/~javed/class-NET06F/] |
Today’s class will be very useful for doing the simulation assignment. In fact, I will describe all the algorithms that you will need to know to do the first assignment.
A Simple Data Link Layer Protocol

Sender Layer

```c
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender1(void)
{
    frame s;
    packet buffer; /* buffer for an outbound frame */
    /* buffer for an outbound packet */

    while (true) {
        from_network_layer(&buffer); /* go get something to send */
        s.info = buffer; /* copy it into s for transmission */
        to_physical_layer(&s); /* send it on its way */
        Creeps in this petty pace from day to day
        To the last syllable of recorded time
        - Macbeth, V, v */
    }
}
```
Receiver Layer

```c
void receiver1(void)
{
    frame r;
    event_type event; /* filled in by wait, but not used here */
    
    while (true) {
        wait_for_event(&event); /* only possibility is frame_arrival */
        from_physical_layer(&r); /* go get the inbound frame */
        to_network_layer(&r.info); /* pass the data to the network layer */
    }
}
```

Characteristics of Protocol-1

- A very simple data link layer protocol.
- Features:
  - One directional communication only.
  - Assumes that the communication channel is error free.
  - Assumes that receiver can process all the input infinitely fast.
Protocol-2 with Limited Buffer

- Stop-and-go: After receiving each frame the receiver sends a dummy ACK frame. Sender cannot flood the receiver.

```c
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void) {
    frame s; /* buffer for an outbound frame */
    packet buffer; /* buffer for an outbound packet */
    event_type event; /* frame_arrival is the only possibility */

    while (true) {
        from_network_layer(&buffer); /* go get something to send */
        s.info = buffer; /* copy it into s for transmission */
        to_physical_layer(&s); /* bye bye little frame */
        wait_for_event(&event); /* do not proceed until given the go ahead */
    }
}
```

Protocol-2 with Limited Buffer

```c
void receiver2(void) {
    frame r, s; /* buffers for frames */
    event_type event; /* frame_arrival is the only possibility */

    while (true) {
        wait_for_event(&event); /* only possibility is frame_arrival */
        from_physical_layer(&r); /* go get the inbound frame */
        to_network_layer(&r.info); /* pass the data to the network layer */
        to_physical_layer(&s); /* send a dummy frame to awaken sender */
    }
}
```

- But still, the transmission is not reliable.
- What if there is a timer?
- What if the ACK is lost?
Characteristics of Protocol-1

- A very simple data link layer protocol.
- Features:
  - One directional communication only.
  - Assumes that the communication channel is error free.
  - Assumes that receiver can process all the input infinitely fast.

The problem of Missing ACK

- Packet A sent
- Timer runs out
- Copy of A resent
- Packet A received
- ACK of A sent and lost
- Copy of A received
- Is it a copy of A or B?
- Solution?
Protocol-3 with Sequence Number

```c
#define MAX_SEQ 1 /* must be 1 for protocol 3 */

typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"

void sender3(void)
{
    seq_nr next_frame_to_send; /* seq number of next outgoing frame */
    frame s; /* scratch variable */
    packet buffer; /* buffer for an outbound packet */
    event_type event;
    next_frame_to_send = 0; /* initialize outbound sequence number */
    for_network_layer(&buffer); /* fetch first packet */
    while (true) {
        s.info = buffer; /* construct a frame for transmission */
        s.seq = next_frame_to_send; /* insert sequence number in frame */
        to_physical_layer(&s); /* send it on its way */
        start_timer(s.seq); /* if (answer takes too long, time out */
        wait_for_event(&event); /* frame_arrival, cksum_err, timeout */
        if (event == frame_arrival) { /* get the acknowledgement */
            from_physical_layer(&s);
            if (s.ack == next_frame_to_send) { /* get the next one to send */
                inc(next_frame_to_send); /* invert next_frame_to_send */
            }
        }
    }
}
```

Protocol-3 with Sequence Number

```c
void receiver3(void)
{
    seq_nr frame_expected;
    frame r, s;
    event_type event;
    frame_expected = 0;
    while (true) {
        wait_for_event(&event); /* possibilities: frame_arrival, cksum_err */
        if (event == frame_arrival) { /* a valid frame has arrived */
            from_physical_layer(&r); /* get the newly arrived frame */
            if (r.seq == frame_expected) { /* this is what we have been waiting for */
                to_network_layer(&r.info); /* pass the data to the network layer */
                inc(frame_expected); /* next time expect the other sequence nr */
            }
            s.ack = 1 - frame_expected; /* tell which frame is being acked */
            to_physical_layer(&s); /* none of the fields are used */
        }
    }
}
```
Protocol-3: Various Scenarios

- Everything Fine
- ACK lost

- Frame Lost
- Low time out value
Protocol-3: Transmission Sequence

**Quiz:**
- How many bits are needed in the sequence number field?

Protocol-4: Full-duplex Communication

- In full-duplex communication instead of sending separate ACK frames, both sides have a sender and a receiver.
- Each side use the outgoing frames to **piggyback** the ACK of the incoming channel.
- Each frame now has a ACK field. It is only few bits. This overhead is much better than sending a series of extra frames (fewer frame arrived interrupt, fewer checksum check, etc.)
- If there is no, outgoing frame, only then it sends a distinct ACK only frame. Waiting too long can trigger sender to retransmit.
Transmission Efficiency of Stop-and-go

- Consider a 1.5Mbps link with 45-ms round trip time (RTT)
  - If the packet size is 1 KB, it can send only one frame per 45 ms.
  - This is \(\frac{1024 \times 8}{0.045} = 182\) Kbps
  - Only 1/8th of the link capacity is utilized for simplex
  - Only 1/4th for the duplex version.

• Quiz:
  - How to increase channel utilization?

Protocol-5: Sliding Window

• Solution: Do not wait for ACK for each Frame. Allowed to send N frames without ACK

• Sender and receiver both must have buffer of size N.
• Why?
Sliding Window Management

- SWS=Senders Window Size, RWS=Receivers Window Size
- LFS=Last Frame Sent, LAR=Last ACK Received
- LFA=Last Frame Acceptable, NFE=Next Frame Expected

**Quiz:**
- How many bits are needed in the sequence number field?

Sliding Window and Stop-and-go

- Sliding window of size 1, with 3-bit sequence number. (a) initially, (b) after first frame has been sent, (c) after the first frame has been received, (d) after the first ACK has been received.
Efficiency of Sliding Window

• If window size $n \times$ time to send one packet is larger than RTT then it has the potential of fully utilizing the channel capacity.

• Quiz:
  – what if one of the frame in the middle of a long stream is lost?

Error in a Sequence

• If a frame in the middle of a long stream is lost or damages, large number of succeeding frames will arrive at the receiver, before the sender even finds out that anything is wrong!
• When a damaged frame arrives at the receiver. It is discarded.
• But, what to do with the correct frames following it?
  – Strategy-1: Go Back-n (protocol-5)
  – Strategy-2: Selective repeat. (protocol-6)
Go-Back-N
(Protocol-5)

- Large number of frames lost!

Selective Repeat

- Receiver stores all the correct frames following bad one. When the sender finally notices the that something is wrong, it just retransmits the bad one. Not all its successors.

- Receiver, waits until all the preceding ones have been passed to the network layer.
Conclusions

- In a noisy channel Selective Time out is more efficient. But it needs more buffer, and separate clock timer for every frame.
- The sliding-window protocol is a many in one. It performed:
  - Reliability by retransmission
  - Ordering of frames
  - Flow control/ buffer management.
- It is possible to design communication model which can separate this three functions and perform them at separate layers.

Protocol-1

- A very simple data link layer protocol.
- Features:
  - One directional only.
  - Assumes that the communication channel is error free.
  - Assumes that receiver can process all the input infinitely fast.