Announcements

- **Reading**
  - Today: Chapter 5 (5.1-5.2)

- **Program #1 Due at 10 PM not 10AM**

- **Notes on the project have been posted to the web**

- **Note on CRC example from last time:**
  - a generator of degree $r$ has as it’s largest term $x^r$ (or $r+1$ bits)
    so the example in the notes is fine.
Simple Link Protocols

- **Stop-and-wait**
  - Sender
    
    ```
    while (1) {
      get frame from network layer;
      send frame;
      wait for ack;
    }
    ```

  - Receiver:
    
    ```
    while (1) {
      recv frame;
      send frame to network layer;
      send ack;
    }
    ```

- Only one side active (sending) at once
- Ensures rate matching
Sliding Window Protocol

- **Need to**
  - have multiple outstanding packets
  - limit total number of outstanding packets
  - permit re-transmissions to occur

- **Sliding Window**
  - permit at most N outstanding packets
  - when packet is ACK’d advance window to first non-ACK’d pkt

- **Retransmission**
  - Go-back N
    - when a packet is lost, restart from that packet
    - provides in-order delivery, but wastes bandwidth
  - Selective Retransmission
    - use timeout to re-sent lost packet
    - use NACK as a **hint** that something was lost
Sliding Window Example

From: Computer Networks, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.
Network Layer

- **Responsibility**
  - end-to-end delivery of packets to the network
  - selecting routes for the packets to take
    - implies knowledge of the network topology
  - managing utilization of the links
    - provide flow control (across multiple links)
    - spread load among different routes

- **Interface Design**
  - should be independent of subnet technology
  - hide number, type, and topology of network from upper layers
  - export a common number plan for entire network
Connection vs. Connectionless

- **Two possible designs for network layer**
  - connection oriented service (ATM)
    - based on experience of telcos
  - connectionless service (IP)
    - based on packet switching (ARPANET)

- **Connectionless**
  - transport datagrams from source to destination
    - end-point addresses in every datagram
  - less complex network layer, more complex transport

- **Connection oriented**
  - also called virtual circuits
  - establish an end-to-end connection with network state
    - can use VCI (global or next hop) in each packet
Datagram vs. VC Addresses

- **Datagrams**
  - must include full address in each packet
  - addresses must be unique for entire network
    - don’t re-use too often
    - addresses per src/dest pair

- **Virtual Circuit**
  - globally unique
    - requires allocation scheme to ensure its unique
    - consumes many bits per packet
  - per link
    - requires translation at each switch
    - uses fewer bits (important for small packets like ATM)
Link Failure in Virtual Circuits

- **Re-establish virtual circuit**
  - router near failure can patch up link
  - original host/router creates new virtual circuit

- **Virtual circuit is dropped**
  - transport layer can handle recovery
## Virtual Circuit vs. Datagram

<table>
<thead>
<tr>
<th>Issue</th>
<th>Datagram</th>
<th>Virtual Circuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit setup</td>
<td>not needed</td>
<td>necessary</td>
</tr>
<tr>
<td>Addresses</td>
<td>full source/dest per packet</td>
<td>next hop vc sufficient</td>
</tr>
<tr>
<td>state</td>
<td>no state in network</td>
<td>per connection data at each router</td>
</tr>
<tr>
<td>routing</td>
<td>each packet individually</td>
<td>once at VC setup</td>
</tr>
<tr>
<td>router/link failure</td>
<td>a few packets may be lost</td>
<td>all VCs through router are terminated</td>
</tr>
<tr>
<td>congestion control</td>
<td>difficult</td>
<td>many pre-allocation and policing policies permitted</td>
</tr>
</tbody>
</table>
Routing: Goals

- **Correctness**
  - packets get where they are supposed

- **Simplicity**
  - easy to implement correctly
  - possible to make routing choices fast (or updates easy)

- **Robustness**
  - failures in the network still permit communication

- **Stability**
  - small changes in link availability results in a small change in the routing information

- **Fairness**
  - each host, VC, or datagram has the same chance

- **Optimality**
  - best possible route
  - best utilization of bandwidth