Chap 5 Peer-to-Peer Protocols and Data Link Layer

• Peer-to-peer protocol: involves the interaction of two or more processes (at the same layer) through the exchange of messages → layer n peer process.
  – e.g. a web browser (client) ↔ a web server application layer peer processes

5.1.3 End to End versus Hop by Hop

• Two basic settings of peer-to-peer protocols:
  1) across a single hop
  2) end-to-end across an entire network

• Example: to provide reliable communication (Fig 5.8)
  – Two options:
    1. detect/recover from error at every hop
       \[ A \leftrightarrow B \leftrightarrow C \leftrightarrow D \text{ (e.g. HDLC)} \]
    2. only the end systems detect/recover errors
       \[ A \leftrightarrow D \text{ (e.g. TCP)} \]
5.1.3 End to End versus Hop by Hop

- Hop-by-Hop:
  1. Initiate error recovery more quickly
  2. The processing in each node is more complex and every node in the path must operate correctly
  3. If errors are very likely, hop-by-hop is preferred.
- End-to-End:
  1. Consume less processing resources
  2. Bandwidth is wasted if error happens at the first hop
  3. If errors are infrequent, end-to-end is preferred
5.2 ARQ Protocols and Reliable Data Transfer Service

- ARQ: Automatic Repeat Request.
  - Error detection (e.g. by using CRC)
  - Retransmission (if time-out or NAK)
- Objective: to ensure that packets are delivered error free to the dest. Exactly once without duplicates, in the same order in which, they were transmitted.
  - Three basic types:
    - Stop-and-Wait
    - Go-Back-N
    - Selective Repeat.

5.2.1 Stop-and-Wait ARQ

1. The transmitter A sends an information frame to receiver B
2. A stops and waits for an ACK from B
3. If no ACK in some time-out period, A resends the frame, otherwise, A sends the next information frame

The need for sequence numbers.
- Fig 5.10 for an example. (if frames are unnumbered)
  - loss of an ACKs can result in the delivery of a duplicate packet
- Fig 5.11 (if ACKs are unnumbered)
  - premature time-outs (or delayed Acks) combined with loss of I-frame an result in gaps in the delivered packet sequence
- Sequence number cannot be arbitrary large (only a finite number of bits available in the frame headers)
- One-bit sequence number is enough for stop-and-wait
  read pg.295-296 of textbook.
Need for Sequence Numbers

(a) Frame 1 lost
- Time-out

(b) ACK lost
- Time-out

- In cases (a) & (b) the transmitting station A acts the same way
- But in case (b) the receiving station B accepts frame 1 twice
- Question: How is the receiver to know the second frame is also frame 1?
- Answer: Add frame sequence number in header
- $S_{\text{last}}$ is sequence number of most recent transmitted frame

Sequence Numbers

(c) Premature Time-out
- Time-out

- The transmitting station A misinterprets duplicate ACKs
- Incorrectly assumes second ACK acknowledges Frame 1
- Question: How is the receiver to know second ACK is for frame 0?
- Answer: Add frame sequence number in ACK header
- $R_{\text{next}}$ is sequence number of next frame expected by the receiver
- Implicitly acknowledges receipt of all prior frames
5.2.1 Stop-and-Wait ARQ (continue)

• Performance analysis:
  – In the absence of errors, the time to send a frame and receive an ACK is:

\[
t_0 = 2t_{\text{prop}} + 2t_{\text{proc}} + t_f + t_{\text{ack}} = 2t_{\text{prop}} + 2t_{\text{proc}} + \frac{n_f}{R} + \frac{n_a}{R}
\]

where:
- \( t_{\text{prop}} \): propagation delay
- \( t_{\text{proc}} \): processing time of frame (e.g. to check CRC)
- \( n_f \): number of bits in an information frame
- \( n_a \): number of bits in an ACK
- \( R \): bit rate of the transmission channel.

Stop-and-Wait Model
5.2.1 Stop-and-Wait ARQ (continue)

– Throughput in the absence of errors:
  \[ R^0_{\text{eff}} = \frac{n_f - n_0}{t_0} \]
  \(n_0\): number of overhead bits in an I-frame.

– Transmission efficiency:
  \[ \eta_0 = \frac{R^0_{\text{eff}}}{R} = \frac{1 - \frac{n_0}{n_f}}{1 + \frac{n_u}{n_f} + \frac{2(t_{\text{prop}} + t_{\text{proc}})R}{n_f}} \]

5.2.1 Stop-and-Wait ARQ (continue)

– Now consider the effect of errors
  
  \( P_f \): prob. that a frame transmission has errors and needs to be retransmitted

  on average, we need \( \frac{1}{1 - p_f} \) transmissions/frame if errors are independent (Number of transmissions/per frame \( \sim \) Geometric distribution)

  \( \therefore \) To transmit one frame successfully, on average need:

  \[ t_{\text{sw}} = t_0 \cdot \frac{1}{1 - p_f} \Rightarrow R_{\text{eff}} = \frac{n_f - n_0}{t_{\text{sw}}} = R^0_{\text{eff}} (1 - p_f) \]

  So
  \[ \eta_{\text{sw}} = \eta_0 (1 - p_f) = \frac{1 - \frac{n_0}{n_f}}{1 + \frac{n_u}{n_f} + \frac{2(t_{\text{prop}} + t_{\text{proc}})R}{n_f}} \cdot (1 - p_f) \]
5.2.1 Stop-and-Wait ARQ (continue)

- Effect of bit error rate
  - \( p \): bit error rate (for a single bit)
  - If we assume errors of bits are independent, and no error in ACK.
    
    \[
    1 - p = (1-p)^n_f
    \]
  - If we also consider errors in ACK
    
    \[
    1 - p = (1-p)^{n_f+n_a}
    \]
  - Example: \( n_f = 1250 \) bytes with overhead \( n_o = 25 \) bytes, \( n_{ack} = 25 \) bytes. \( R = 1 \) Mbps, Reacting time:
    
    \[
    2(t_{prop} + t_{proc}) = 1 \text{ms.}
    \]
  - Bit error rate \( p = 10^{-6}, 10^{-5}, 10^{-4} \). Find \( \eta_{sw} \)
    
    \[
    \eta_{sw} = 86.6\%, 70.2\%, 32.2\%
    \]

5.2.2 Go-Back-N

- If \( t_{prop} + t_{proc} \gg \frac{n_f}{R} \), then \( \eta_{sw} \ll 1 \). Stop-and-wait very inefficient
  - Stop-and-wait: can only have 1 frame to be outstanding without ACK
  - Go-Back-N: Improve Stop-and-Wait by not waiting! Keep channel busy by continuing to send frames
    - Allow a window of up to \( W_s \) outstanding frames without Ack
    - \( W_s \): window size.
  - Alternative: Use timeout
### 5.2.2 Go-Back-N

1. A sends frame 0, 1, ……, $W_s$-1 to B continuously
2. If B received frames 0, 1, ……, i-1 successfully and frame i with error. B ACKs each of the frame from 0 to i-1 and ignore frames i, i+1, ……
3. When A receives Ack for frame 0 → A sends frame $W_s$
   A receives Ack for frame 1 → A sends frame $W_s+1$
   ……
   A receives Ack for frame i-1 → A sends frame $W_s+i-1$
   A receives no Ack for frame i → A resends i, i+1, ……, i+$W_s$-1.

   See Fig 5.15 for an example $W_s$=4

   When no Ack received, “go back N” frames. N=$W_s$

---

**Go-Back-N ARQ**

- Frame transmission are *pipelined* to keep the channel busy
- Frame with errors and subsequent out-of-sequence frames are ignored
- Transmitter is forced to go back when window of 4 is exhausted
5.2.2 Go-Back-N (continue)

- How to choose $W_s$
  1. too large, inefficient.
  2. too small, no enough time for Ack to arrive.

\[ W_{stf} \geq t_f + t_a + 2(t_{prop} + t_{proc}) \]

\[ W_s = \frac{t_f + t_a + 2(t_{prop} + t_{proc})}{t_f} \]

- Alternative: Use a timeout with each frame
  - When timeout expires, resend all outstanding frames

5.2.2 Go-Back-N (continue)

- Performance analysis
  - $t_{GBN}$: the time that it takes to transmit a frame successfully.
  - $p_f$: prob. That a frame is received with error
    - If transmitted successfully at the first try
      \[ t_{GBN} = t_f \]
    - If transmitted successfully with $i$ retransmissions
      \[ t_{GBN} = t_f + i \cdot W_{stf} \]
    - Let $X$ be the number of retransmissions, $X$: a geometric r.v.
      \[ p_i = P(X=i) = p_i/(1-p_f) \]
    - On average:
      \[ t_{GBN} = \sum_{i=0}^{\infty} p_i (t_f + i \cdot W_{stf}) = t_f + p_f \cdot \frac{W_{stf}}{1-p_f} \]

\[
\eta_{GBN} = \frac{t_{GBN}}{R} = \frac{n_f - n_0}{1 + (w_i - 1)p_f} \cdot (1 - p_f)
\]
Example

$n_f=1250$ bytes = 10000 bits, $n_a=n_d=25$ bytes = 200 bits

Compare S&W with GBN efficiency for random bit errors with $p = 0$, $10^{-6}$, $10^{-5}$, $10^{-4}$ and $R = 1$ Mbps & reaction time=100 ms

Use $W_s = 11$

<table>
<thead>
<tr>
<th>Efficiency</th>
<th>0</th>
<th>$10^{-6}$</th>
<th>$10^{-5}$</th>
<th>$10^{-4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>S&amp;W</td>
<td>8.9%</td>
<td>8.8%</td>
<td>8.0%</td>
<td>3.3%</td>
</tr>
<tr>
<td>GBN</td>
<td>98%</td>
<td>88.2%</td>
<td>45.4%</td>
<td>4.9%</td>
</tr>
</tbody>
</table>

- Go-Back-N significant improvement over Stop-and-Wait
- Go-Back-N becomes inefficient as error rate increases

5.2.3 Selective Repeat

- If an error happens, Go-Back-N retransmit the frame in error and all the subsequent frames
- Selective Repeat:
  1. Only the frame in error is retransmitted
  2. The receiver can accept frames that are out-of-order but error free
- Performance Analysis

\[ t_{sa} = \frac{t_f}{1-p_f} \]

\[ n_f - n_o \]

\[ \eta_{SR} = \frac{t_{sa}}{R} = (1 - \frac{n_o}{n_f})(1 - p_f) \]

- The performance increases at the cost of significantly greater complexity.
Example

\( n_f = 1250 \text{ bytes} = 10000 \text{ bits}, \ n_a = n_0 = 25 \text{ bytes} = 200 \text{ bits} \)

Compare S&W, GBN & SR efficiency for random bit errors with \( p = 0, \ 10^{-6}, \ 10^{-5}, \ 10^{-4} \) and \( R = 1 \text{ Mbps} \) & reaction time=100 ms

<table>
<thead>
<tr>
<th>Efficiency</th>
<th>0</th>
<th>10^{-6}</th>
<th>10^{-5}</th>
<th>10^{-4}</th>
</tr>
</thead>
<tbody>
<tr>
<td>S&amp;W</td>
<td>8.9%</td>
<td>8.8%</td>
<td>8.0%</td>
<td>3.3%</td>
</tr>
<tr>
<td>GBN</td>
<td>98%</td>
<td>88.2%</td>
<td>45.4%</td>
<td>4.9%</td>
</tr>
<tr>
<td>SR</td>
<td>98%</td>
<td>97%</td>
<td>89%</td>
<td>36%</td>
</tr>
</tbody>
</table>

- Selective Repeat outperforms GBN and S&W, but efficiency drops as error rate increases