Multimedia Communication

Multimedia Systems(Module 5 Lesson 2)

Summary:

- Internet Phone Example
  - Making the Best use of Internet's Best-Effort Service.

Sources:

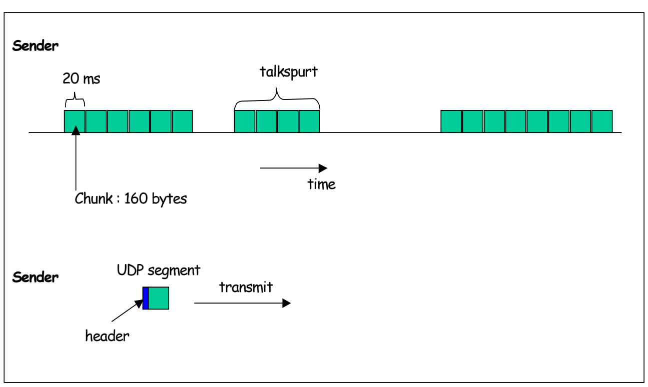
- Chapter 6 from "Computer Networking: A Top-Down Approach Featuring the Internet", by Kurose and Ross

Best of Best-Effort: Internet Phone

The Scenario (Similar to application category 3):

- The speaker generates an audio signal consisting of alternating *talk spurts* and *silent periods.*
- To conserve bandwidth, the application only generates packets during talk spurts. During a talk spurt the sender generates bytes at a rate of 8 Kbytes per second, and every 20 milliseconds the sender gathers bytes into chunks.
  - Number of bytes in a chunk is (20 msecs) · (8 Kbytes/sec) = 160 bytes.
  - A special header is attached to each chunk (!Think RTP!).
- The chunk and its header are encapsulated in a UDP segment, and then the UDP datagram is sent into the socket interface. Thus, during a talk spurt, a UDP segment is sent every 20 msec.

Scenario Figure



Internet Phone Scenario Contd.

- If each packet makes it to the receiver and has a small constant end-to-end delay, then packets arrive at the receiver periodically every 20 msec during a talk spurt. Ideally, the receiver can simply play back each chunk as soon as it arrives.
- Unfortunately, some packets can be lost and most packets will not have the same end-to-end delay, even in a lightly congested Internet. Therefore, the receiver must take more care in
  - determining when to play back a chunk, and
  - determining what to do with a missing chunk.

Limitations of Best-Effort Service

- Packet Loss
  - The UDP datagram traverses intermediate routers' buffers/queues.
  - An datagram may find that the intermediate router has its queue/buffer full: The router drops the datagram
  - Why not use TCP? *Retransmissions!!*
  - Depending on how the audio is coded (maybe with redundancy) we can tolerate losses of 1%-20%.
- End-to-End Delay
  - Transmission processing delay + queuing delay + Propagation delay + end-system processing delay.
  - Acceptable for voice communication: $\leq$ 400 ms.
- Delay Jitter
  - Inter-packet delay at sender is 20 ms. Inter-packet arrival times at receiver may be greater than or less than 20 ms.
  - If the receiver plays as it receives the audio quality suffers.

A. Removing Jitter at Receiver

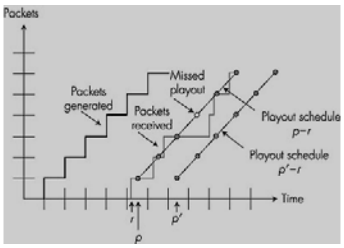
Can be done by *combining* the following three mechanisms:

- **Prefacing each chunk with a sequence number**
  - The sender increments the sequence number by one for each packet it generates.
- **Prefacing each chunk with a timestamp**
  - The sender stamps each chunk with the time at which the chunk was generated.
- **Delaying playout**
  - The playout delay must be long enough so that most of the packets are received before their *scheduled* playout times.
  - Packets that do not arrive before their scheduled playout times are considered lost.
  - The playout delay may be a constant throughout the duration of the conference, or it may vary adaptively during the conference.

Constant Playout Delay

Strategy: The receiver attempts to playout each chunk exactly *q* ms after the chunk is *generated (timestamp).*

- Good choice of *q* : (≤ 400 ms)
  - If there are large variations in end-to-end delay, it is preferable to use a large value of *q*;
  - On the other hand if the variations in delay are small then a small value of q is prefered (perhaps less than 150ms)
- Tradeoff between playout delay and packet loss:



p-r playout schedule:

- The 4th packet misses its scheduled playout time

p'-r playout schedule:

- All packets arrive

Adaptive Playout Delay

Motivation: Minimize the playout delay subject to the constraint that the loss be below a few percent.

Strategy: Estimate the network delay and the variance of the network delay and adjust the playout delay accordingly at the beginning of each talkspurt.

A Generic Algorithm

Let

t_i = timestamp of the i^{th} packet

r_i = the time at which packet *i* is received by the receiver

p_i = the time at which packet *i* is played at the receiver

So, end-to-end delay for packet *i* is $r_i - t_i$

Let

d_i denote an estimate of the *average* network delay upon receipt of the *i'th* packet

Derive,

$d_i = (1-u)\, d_{i-1} + u\,(r_i - t_i)$

where, *u* is a fixed constant (e.g. 0.01)

d_i is a smoothed average of the observed network delays $r_1 - t_1, \ldots, r_i - t_i$

Adaptive Playout Delay

A Generic Algorithm(Contd.)

Let

v_i denote an estimate of the average deviation of the observed network delay from the estimated average delay.

Derive,

$v_i = (1-u)\, v_{i-1} + u\,|r_i - t_i - d_i|$

The estimates d_i and v_i are calculated for every packet received, although they are only used to determine the playout point for the first packet in a talkspurt.

Adaptive Playout Delay

Receiver Playout Algorithm:

- If packet i is the first packet of a talkspurt, its playout time is:

  $p_i = t_i + d_i + Kv_i$

  Where K is a +ve constant (e.g., K=4). The Kv_i term is to set the playout time far enough into the future so that only a small fraction of packets in the talkspurt will be lost due to late arrivals.
- The playout point for any subsequent packet in a talkspurt is given by:

  $p_j = t_j + q_i$

  where,

  $q_i = p_i - t_i$, is the length of time from when the first packet of the talkspurt to which paket j belongs is generated until it is played out.
- How do you find out if a packet i is the first packet of a talkspurt?
  - Compare the timestamps of the i-1^{th} and i^{th} packet:
    if $t_i - t_{i-1}$ > 20 ms then the packet is the first in a talkspurt
  - What if one of the packets within a talkspurt is lost?
    then the timestamp difference between the two packets adjacent to the lost packet may differ by > 20 ms
  - Use the sequence number to detect whether the difference of more than 20 ms is due to a new talkspurt or a lost packet.

B. Recovering from Packet Loss

A packet is lost if either it never arrives at the receiver or it arrives after its scheduled playout time.

Three Approaches:

- **Forward Error Correction (FEC)**
  - **Idea**: Add redundant information to the original packet stream.
  - Used in FreePhone and RAT (a MBONE audio conf. tool)
- **Interleaving**
  - **Idea**: Interleave (reorder the sequence) the media so that originally adjacent units are separated by a certain distance in the transmitted stream.
- **Receiver-Based Repair**
  - **Idea**: Receiver attempts to produce a replacement for a lost packet that is similar to the original.

FEC

Mechanism 1:

- Sends one redundant encoded chunk after every n chunks.
- The redundant chunk is obtained by exclusive OR-ing the n original chunks
  - If any one packet of the group of $n + 1$ packets is lost, the receiver can fully reconstruct the lost packet.
  - If two or more packets in a group are lost, the receiver cannot reconstruct the lost packets.
  - By keeping $n + 1$, the group size, small, a large fraction of the lost packets can be recovered when loss is not excessive.
  - Smaller the group size, greater the relative increase of the transmission rate of the audio stream. In particular, the transmission rate increases by a factor of $1/n$; for example, if n = 3, then the transmission rate increases by 33%.
- This simple scheme increases the playout delay, as the receiver must wait to receive the entire group of packets before it can begin playout.

FEC

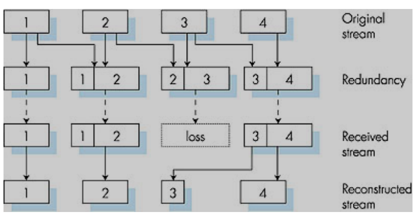
Mechanism 2:

- Send a lower-resolution audio stream as the redundant information.
  - For example, the sender might create a nominal audio stream and a corresponding low-resolution low-bit rate audio stream. (The nominal stream could be a PCM encoding at 64 Kbps and the lower-quality stream could be a GSM encoding at 13 Kbps.)
- The low-bit-rate stream is referred to as the redundant stream.
- The sender constructs the *n*th packet by taking the *n*th chunk from the nominal stream and appending to it the (*n* - 1)st chunk from the redundant stream.

FEC

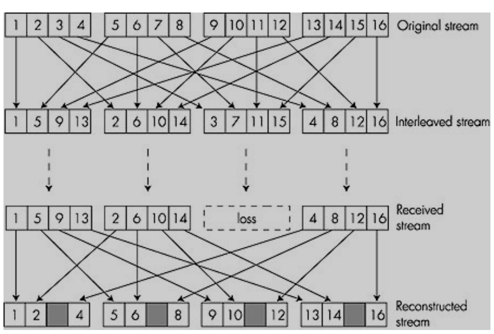
Mechanism 2 (Contd.):

- If there is nonconsecutive packet loss, the receiver can conceal the loss by playing out the low-bit-rate encoded chunk that arrives with the subsequent packet.
- The receiver only has to receive two packets before playback, so that the increased playout delay is small.
- If the low-bit-rate encoding is much less than the nominal encoding, then the marginal increase in the transmission rate will be small.



Interleaving

- Interleaving can mitigate the effect of packet losses. If, for example, units are 5 msec in length and chunks are 20 msec (that is, 4 units per chunk), then the first chunk could contain units 1, 5, 9, 13; the second chunk could contain units 2, 6, 10, 14; and so on.



Interleaving (Contd.)

- The loss of a single packet (see figure) from an interleaved stream results in multiple small gaps in the reconstructed stream, as opposed to the single large gap that would occur in a non-interleaved stream.
- The obvious disadvantage of interleaving is that it increases latency. This limits its use for interactive applications such as Internet phone, although it can perform well for streaming stored audio.
- A major advantage of interleaving is that it does not increase the bandwidth requirements of a stream.