

msecs can be acceptable but are not ideal; and delays exceeding 400 msecs can seriously hinder the interactivity in voice conversations. The receiving side of a VoIP application will typically disregard any packets that are delayed more than a certain threshold, for example, more than 400 msecs. Thus, packets that are delayed by more than the threshold are effectively lost.

Packet Jitter

A crucial component of end-to-end delay is the varying queuing delays that a packet experiences in the network's routers. Because of these varying delays, the time from when a packet is generated at the source until it is received at the receiver can fluctuate from packet to packet, as shown in Figure 7.1. This phenomenon is called **jitter**. As an example, consider two consecutive packets in our VoIP application. The sender sends the second packet 20 msecs after sending the first packet. But at the receiver, the spacing between these packets can become greater than 20 msecs. To see this, suppose the first packet arrives at a nearly empty queue at a router, but just before the second packet arrives at the queue a large number of packets from other sources arrive at the same queue. Because the first packet experiences a small queuing delay and the second packet suffers a large queuing delay at this router, the first and second packets become spaced by more than 20 msecs. The spacing between consecutive packets can also become less than 20 msecs. To see this, again consider two consecutive packets. Suppose the first packet joins the end of a queue with a large number of packets, and the second packet arrives at the queue before this first packet is transmitted and before any packets from other sources arrive at the queue. In this case, our two packets find themselves one right after the other in the queue. If the time it takes to transmit a packet on the router's outbound link is less than 20 msecs, then the spacing between first and second packets becomes less than 20 msecs.

The situation is analogous to driving cars on roads. Suppose you and your friend are each driving in your own cars from San Diego to Phoenix. Suppose you and your friend have similar driving styles, and that you both drive at 100 km/hour, traffic permitting. If your friend starts out one hour before you, depending on intervening traffic, you may arrive at Phoenix more or less than one hour after your friend.

If the receiver ignores the presence of jitter and plays out chunks as soon as they arrive, then the resulting audio quality can easily become unintelligible at the receiver. Fortunately, jitter can often be removed by using **sequence numbers**, **timestamps**, and a **playout delay**, as discussed below.

7.3.2 Removing Jitter at the Receiver for Audio

For our VoIP application, where packets are being generated periodically, the receiver should attempt to provide periodic playout of voice chunks in the presence

of random network jitter. This is typically done by combining the following two mechanisms:

- *Prepending each chunk with a **timestamp**.* The sender stamps each chunk with the time at which the chunk was generated.
- ***Delaying playout** of chunks at the receiver.* As we saw in our earlier discussion of Figure 7.1, the playout delay of the received audio chunks must be long enough so that most of the packets are received before their scheduled playout times. This playout delay can either be fixed throughout the duration of the audio session or vary adaptively during the audio session lifetime.

We now discuss how these three mechanisms, when combined, can alleviate or even eliminate the effects of jitter. We examine two playback strategies: fixed playout delay and adaptive playout delay.

Fixed Playout Delay

With the fixed-delay strategy, the receiver attempts to play out each chunk exactly q msecs after the chunk is generated. So if a chunk is timestamped at the sender at time t , the receiver plays out the chunk at time $t + q$, assuming the chunk has arrived by that time. Packets that arrive after their scheduled playout times are discarded and considered lost.

What is a good choice for q ? VoIP can support delays up to about 400 msecs, although a more satisfying conversational experience is achieved with smaller values of q . On the other hand, if q is made much smaller than 400 msecs, then many packets may miss their scheduled playback times due to the network-induced packet jitter. Roughly speaking, if large variations in end-to-end delay are typical, it is preferable to use a large q ; on the other hand, if delay is small and variations in delay are also small, it is preferable to use a small q , perhaps less than 150 msecs.

The trade-off between the playback delay and packet loss is illustrated in Figure 7.7. The figure shows the times at which packets are generated and played out for a single talk spurt. Two distinct initial playout delays are considered. As shown by the leftmost staircase, the sender generates packets at regular intervals—say, every 20 msecs. The first packet in this talk spurt is received at time r . As shown in the figure, the arrivals of subsequent packets are not evenly spaced due to the network jitter.

For the first playout schedule, the fixed initial playout delay is set to $p - r$. With this schedule, the fourth packet does not arrive by its scheduled playout time, and the receiver considers it lost. For the second playout schedule, the fixed initial playout delay is set to $p' - r$. For this schedule, all packets arrive before their scheduled playout times, and there is therefore no loss.

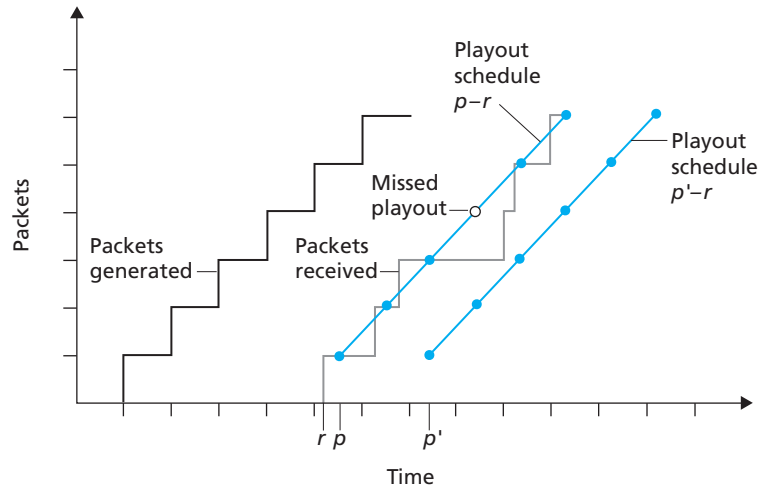


Figure 7.7 ♦ Packet loss for different fixed playout delays

Adaptive Playout Delay

The previous example demonstrates an important delay-loss trade-off that arises when designing a playout strategy with fixed playout delays. By making the initial playout delay large, most packets will make their deadlines and there will therefore be negligible loss; however, for conversational services such as VoIP, long delays can become bothersome if not intolerable. Ideally, we would like the playout delay to be minimized subject to the constraint that the loss be below a few percent.

The natural way to deal with this trade-off is to estimate the network delay and the variance of the network delay, and to adjust the playout delay accordingly at the beginning of each talk spurt. This adaptive adjustment of playout delays at the beginning of the talk spurts will cause the sender's silent periods to be compressed and elongated; however, compression and elongation of silence by a small amount is not noticeable in speech.

Following [Ramjee 1994], we now describe a generic algorithm that the receiver can use to adaptively adjust its playout delays. To this end, let

t_i = the timestamp of the i th packet = the time the packet was generated by the sender

r_i = the time packet i is received by receiver

p_i = the time packet i is played at receiver

The end-to-end network delay of the i th packet is $r_i - t_i$. Due to network jitter, this delay will vary from packet to packet. Let d_i denote an estimate of the *average*