# 6.1 Multimedia Networking Applications

Back in Chapter 2 we examined the Web, file transfer, and electronic mail in some detail. The data carried by these networking applications is, for the most part, static content such as text and images. When static content is sent from one host to another, it is desirable for the content to arrive at the destination as soon as possible. Nevertheless, moderately long end-to-end delays, up to tens of seconds, are often tolerated for static content.

In this chapter we consider networking applications whose data contains audio and video content. We shall refer to networking applications as **multimedia networking applications**. (Some authors refer to these applications *continuous-media applications*.) Multimedia networking applications are typically *highly* sensitive to delay; depending on the particular multimedia networking application, packets that incur more than an x second delay - where x can range from a 100 msecs to five seconds - are useless. On the otherhand, multimedia networking applications are typically **loss tolerant**; occassional loss only causes occassional glitches in the audio/video playback, and often these losses can be partially or fully concealed. Thus, in terms of service requirements, multimedia applications are diametrically opposite of static-content applications: multimedia applications are delay sensitive and loss tolerant whereas the static-content applications are delay tolerant and loss intolerant.

## 6.1.1 Examples of Multimedia Applications

The Internet carries a large variety of exciting multimedia applications. Below we define three classes of multimedia applications.

1. **Streaming stored audio and video:** In this class of applications, clients request on-demand compressed audio or video files, which are stored on servers. For audio, these files can contain a professor's lectures, rock songs, symphonies, archives of famous radio broadcasts, as well as historical archival recordings. For video, these files can contain video of professors' lectures, full-length movies, prerecorded television shows, documentaries, video archives of historical events, video recordings of sporting events, cartoons and music video clips. At any time a client machine can request an audio/video file from a server. In most of the existing stored audio/video applications, after a delay of a few seconds the client begins to playback the audio file *while* it continues to receive the file from the server. The feature of playing back audio or video while the file is being received is called **streaming**. Many of the existing products also provide for **user interactivity**, e.g., pause/resume and temporal jumps to the future and past of the audio file. The delay from when a user makes a request (e.g., request to hear an audio file or skip two-minutes forward) until the action manifests itself at the the user host (e.g., user begins to hear audio file) should be on the order of 1 to 10 seconds for acceptable responsiveness. Requirements for packet delay and jitter are not as stringent as those for real-time applications such as Internet telephony and real-time video conferencing (see below). There are many streaming products for stored audio/video, including RealPlayer from RealNetworks and NetShow from Microsoft.
2. **One to many streaming of real-time audio and video:** This class of applications is similar to ordinary broadcast of radio and television, except the transmission takes place over the Internet. These applications allow a user to receive a radio or television transmission emitted from any corner of the world. (For example, one of the authors of this book often listens to his favorite Philadelphia radio stations from his home in France.) Microsoft provides an Internet radio station

guide. Typically, there are many users who are simultaneously receiving the same real-time audio/video program. This class of applications is non-interactive; a client cannot control a server's transmission schedule. As with streaming of stored multimedia, requirements for packet delay and jitter are not as stringent as those for Internet telephony and real-time video conferencing. Delays up to tens of seconds from when the user clicks on a link until audio/video playback begins can be tolerated. Distribution of the real-time audio/video to many receivers is efficiently done with multicast; however, as of this writing, most of the one-to-many audio/video transmissions in the Internet are done with separate unicast streams to each of the receivers.

3. **Real-time interactive audio and video:** This class of applications allows people to use audio/video to communicate with each other in real-time. Real-time interactive audio is often referred to as **Internet phone**, since, from the user's perspective, it is similar to traditional circuit-switched telephone service. Internet phone can potentially provide PBX, local and long-distance telephone service at very low cost. It can also facilitate computer-telephone integration (so called CTI), group real-time communication, directory services, caller identification, caller filtering, etc. There are many Internet telephone products currently available.With real-time interactive video, also called video conferencing, individuals communicate visually as well as orally. During a group meeting, a user can open a window for each participant the user is interested in seeing. There are also many real-time interactive video products currently available for the Internet, including Microsoft's Netmeeting. Note that in a real-time interactive audio/video application, a user can speak or move at anytime. The delay from when a user speaks or moves until the action is manifested at the receiving hosts should be less than a few hundred milliseconds. For voice, delays smaller than 150 milliseconds are not perceived by a human listener, delays between 150 and 400 milliseconds can be acceptable, and delays exceeding 400 milliseconds result frustrating if not completely unintilligible voice conversations.

One-to-many real-time audio and video is not interactive - a user cannot pause or rewind a transmission that hundreds of others listen to. Although streaming stored audio/video allows for interactive actions such as pause and rewind, it is not real-time, since the content has already been gathered and stored on hard disks. Finally, real-time interactive audio/video is interactive in the sense that participants can orally and visually respond to each other in real time.

# 6.1.2 Hurdles for Multimedia in the Internet

IP, the Internet's network-layer protocol, provides a **best-effort service** to all the datagrams it carries. In other words, the Internet makes its best effort to move each datagram from sender to receiver as quickly as possible. However, the best-effort service does not make any promises whatsoever about the end-to-end delay for an individual packet. Nor does the service make any promises about the variation of pakcet delay within a packet stream. As we learned in Chapter 3, because TCP and UDP run over IP, neither of these protocols can make any delay guarantees to invoking applications. Due to the lack of any special effort to deliver packets in a timely manner, it is extermely challenging problem to develop successful multimedia networking applications for the Internet. To date, multimedia over the Internet has achieved significant but limited success. For example, streaming store audio/video with user-interactivity delays of five-to-ten seconds is now commonplace in the Internet. But during peak traffic periods, performance may be unsatisfactory, particularly when intervening links are congested links (such as congested transoceanic link).

Internet phone and real-time interactive video has, to date, been less successful than streaming stored audio/video. Indeed, real-time interactive voice and video impose rigid constraints on packet delay and packet jitter. **Packet jitter** is the variability of packet delays within the same packet stream. Real-time voice and video can work well in regions where bandwidth is plentiful, and hence delay and jitter are minimal. But quality can deteriorate to unacceptable levels as soon as the real-time voice or video packet stream hits a moderately congested link.

The design of multimedia applications would certainly be more straightforward if their were some sort of first-class and second-class Internet services, whereby first-class packets are limited in number and always get priorities in router queues. Such a first-class service could be satisfactory for delay-sensitive applications. But to date, the Internet has mostly taken an egalitarian approach to packet scheduling in router queues: all packets receive equal service; no packets, including delay-sensitive audio and video packets, get any priorities in the router queues. No matter how much money you have or how important you are, you must join the end of the line and wait your turn!

So for the time being we have to live with the best effort service. No matter how important or how rich we are, our packets have to wait their turn in router queues. But given this constraint, we can make several design decisions and employ a few tricks to improve the user-perceived quality of a multimedia networking application. For example, we can send the audio and video over UDP, and thereby circumvent TCP's low throughput when TCP enters its slow-start phase. We can delay playback at the receiver by 100 msecs or more in order to diminish the effects of network-induced jitter. We can timestamp packets at the sender so that the receiver knows when the packets should be played back. For stored audio/video we can prefetch data during playback when client storage and extra bandwidth is available. We can even send redundant information in order to mitigate the effects of network-induced packet loss. We shall investigate many of these techniques in this chapter.

## 6.1.3 How Should the Internet Evolve to Better Support Multimedia?

Today there is a tremendous -- and sometimes ferocious -- debate about how the Internet should evolve in order to better accommodate multimedia traffic with its rigid timing constraints. At one extreme, some researchers argue that it isn't necessary to make any fundamental changes to the best-effort service and the underlying Internet protocols. Instead, according to these extremists, it is only necessary to add more bandwidth to the links (along with network caching for stored information and multicast support for one-to-many real-time streaming). Opponents to this viewpoint argue that additional bandwidth can be costly, and as soon as it is put in place it will be eaten up by new bandwidth hungry applications (e.g., high-definition video on demand).

At the other extreme, some researchers argue that fundamental changes should be made to the Internet so that applications can explicitly reserve end-to-end bandwidth. These researchers feel, for example, that if a user wants to make an Internet phone call from host A to host B, then the user's Internet phone application should be able to explicitly reserve bandwidth in each link along a route from host A to host B. But allowing applications to make reservations and requiring the network to honor the reservations requires some big changes. First we need a protocol that, on the behalf of applications, reserves bandwidth from the senders to their receivers. Second, we need to modify scheduling policies in the router queues so that bandwidth reservations can be honored. With these new scheduling policies, all packets no longer get equal treatment; instead, those that reserve (and pay) more get more. Third, in order to honor reservations, the applications need to give the network a description of the traffic that they intend to send into the network. The network must then police each application's traffic to make sure that it abides to the description. Finally, the network must have a means of determining whether it has sufficient available bandwidth to support any new reservation request. These mechanisms, when combined, require new and complex software in the hosts and routers as well as new types of services.

There is a camp inbetween the two extremes - the so-called differentiated services camp. This camp wants to make relatively small changes at the network and transport layers, and introduce simple pricing and policing schemes at the edge of the network (i.e., at the interface between the user and the user's ISP). The idea is to introduce a small number of classes (possibly just two classes), assign each datagram to one of the classes, give datagrams different levels of service according to their class in the router queues, and charge users to reflect the class of packets that they are emitting into the network. A simple example of a differentiated-services Internet is as follows. By toggling a single bit in the datagram header, all IP datagrams are labeled as either first-class or second-class datagrams. In each router queue, each arriving first class datagram jumps in front of all the second-class datagrams; in this manner, second-class datagrams do not interfere with first-class datagrams -- it as if the first-class packets have their own network! The network edge counts the number of first-class datagrams each user sends into the network each week. When a user subscribes to an Internet service, it can opt for a "plantinum service" whereby the user is permitted to send a large but limited number of first-class datagrams into the network each week; first-class datagrams in excess of the limit are converted to second-class datagrams at the network edge. A user can also opt for a "low-budget" service, whereby all of his datagrams are second-class datagrams. Of course, the user pays a higher subscription rate for the plantinum service than for the low-budget service. Finally, the network is dimensioned and the first-class service is priced so that "almost always" first-class datagrams experience insignificant delays at all router queues. In this manner, sources of audio/video can subscribe to the first-class service, and thereby receive "almost always" satisfactory service. We will cover differentiated services in Section 6.8.

## 6.1.4 Audio and Video Compression

Before audio and video can be transmitted over a computer network, it has to be digitized and compressed. The need for digitization is obvious: computer networks transmit bits, so all transmitted information must be represented as a sequence of bits. Compression is important because uncompressed audio and video consumes a tremendous amount of storage and bandwidth; removing the inherent redundancies in digitized audio and video signals can reduce by orders of magnitude the amount the data that needs to be stored and transmitted. As an example, a single image consisting of 1024 pixels x 1024 pixels with each pixel encoded into 24 bist requires 3 MB of storage without compression. It would take seven minutes to send this image over a 64 Kbps link. If the image is compressed at a modest 10:1 compression ratio, the storage requirement is reduced to 300 KB and the transmission time drops to under 6 seconds.

The fields of audio and video compression are vast. They have been active areas of research for more than 50 years, and there are now literally hundreds of popular techniques and standards for both audio and video compression. Most universities offer entire courses on audio and video compression, and often offer a separate course on audio compression and a separate course on video compression. Furthermore, electrical engineering and computer science departments often offer independent courses on the subject, with each department approaching the subject from a different angle. We therefore only provide here a brief and high-level introduction to the subject.

## Audio Compression in the Internet

A continuously-varying analog audio signal (which could emanate from speech or music) is normally converted to a digital signal as follows:

- The analog audio signal is first sampled at some fixed rate, e.g., at 8,000 samples per second. The value of each sample is an arbitrary real number.
- Each of the samples is then "rounded" to one of a finite number of values. This operation is referred to as "quantization". The number of finite values - called quantization values - is typically a power of 2, e.g., 256 quantization values.
- Each of the quantization values is represented by a fixed number of bits. For example if there are 256 quantization values, then each value - and hence each sample - is represented by 1 byte. Each of the samples is converted to its bit representation. The bit representations of all the samples are concatenated together to form the digital representation of the signal.

As an example, if an analog audio signal is sampled at 8,000 samples per second , each sample is quantized and represented by 8 bits, then the resulting digital signal will have a rate of 64,000 bits per second. This digital signal can then be converted back - i.e., decoded - to an analog signal for playback. However, the decoded analog signal is typically different from the original audio signal. By increasing the sampling rate and the number of quantization values the decoded signal can approximate (and even be exactly equal to) the original analog signal. Thus, there is a clear tradeoff between the quality of the decoded signal and the storage and bandwidth requirements of the digital signal.

The basic encoding technique that we just described is called **Pulse Code Modulation (PCM)**. Speech encoding often uses PCM, with a sampling rate of 8000 samples per second and 8 bits per sample, giving a rate of 64 kbs. The audio Compact Disk (CD) also uses PCM, without a sampling rate of 44,100 samples per second with 16 bits per sample; this gives a rate of 705.6 Kbps for mono and 1.411 Mbps for stereo.

A bit rate of 1.411 Mbps for stereo music exceeds most access rates, and even 64 kbps for speech exceeds the access rate for a dial-up modem user. For these reasons, PCM encoded speech and music is rarely used in the Internet. Instead compression techniques are used to reduce the bit rates of the stream. Popular compression techniques for speech include GSM (13 Kbps), G.729 (8.5 Kbps) and G.723 (both 6.4 and 5.3 Kbps), and also a large number of proprietary techniques, including those used by RealNetworks. A popular compression technique for near CD-quality stereo music is MPEG layer 3, more commonly known as MP3. MP3 compresses the bit rate for music to 128 or 112 Kbps, and produces very little sound degradation. An MP3 file can be broken up into pieces, and each piece is still playable. This headerless file format allows MP3 music files to be streamed across the Internet (assuming the playback bitrate and speed of the Internet connection are compatible). The MP3 compression standard is complex; it uses psychoacoustic masking, redundancy reduction and bit reservoir buffering.

## Video Compression in the Internet

A video is a sequence images, with each image typically being displayed at a constant rate, for example at 24 or 30 images per second. An uncompressed, digitally encoded image consists of an array of pixels, with each pixel encoded into a number of bits to respresent luminance and color. There are two types of redundancy in video, both of which can be exploited for compression. Spatial redundancy is the redundancy within  a given image. For example, an image that consists of mostly white space can be efficiently compressed. Temporal redundancy reflects repitition from image to subsequent image. If, for example, an image and the subsequent image are exactly the same, there is no reason re-encode the subsequent image; it is more efficient to simply indicate during encoding the subsequent image is exactly the same.

The MPEG compression standards are among the most popular compression techniques. These include MPEG 1 for CD-ROM quality video (1.5 Mbps), MPEG2 for high-quality DVD video (3-6 Mbps) and MPEG 4 for object-oriented video compression. The MPEG standard draws heavily from the JPEG standard for image compression. The H.261 video compression standards are also very popular in the Internet, as well are numerous proprietary standards.

Readers interested in learning more about audio and video encoding are encouraged to see [Rao] and [Solari]. Also, Paul Amer maintains a nice set of links to audio and video compression.

**References**

[Rao] K.R. Rao and J.J. Hwang, Techniques and Standards for Image, Video and Audio Coding, Prentice Hall,  1996

[Solari] S.J. Solari, Digital Video and Audio Compression, McGraw Hill Text, 1997.

Return to Table of Contents

---