Submission instructions: Please type your answers and submit electronic copies using turnin *by 11:59pm on the due date. You may use any number of word processing software (e.g., Framemaker, Word, LATEX), but the final output must be in pdf or ps format that uses standard fonts (a practical test is to check if the pdf/ps file prints on a CS Department printer without problem). For experiments and programming assignments that involve output to terminal, please use* script *to record the output and submit the output file. Use* gnuplot *to plot graphs. Use* ps2gif *to convert a eps/ps plot to gif format (e.g., for inclusion in Word).*

PROBLEM 1

Read Chapters 13 (Sections 13.13–13.18), 17, 25, 26, 27, 31, 34, 35 and 40 from Comer.

PROJECT PROBLEM

Design, implement and benchmark a UDP-based peer-to-peer (P2P) pseudo real-time audio streaming application. Your application can be built on top of the CBR sender/receiver application in Problem 4, Assignment V, with modifications to inject congestion control and audio playback components. The sender, my_audio_send, takes as command-line arguments

%   my_audio_send *dest-IP dest-port audio-file payload-size packet-spacing mode*

where *audio-file* is a stored audio file that will be streamed to the receiver (i.e., client)—unless otherwise indicated, assume the file format is binary—*payload-size* (in bytes) is the size of the UDP payload (excluding a 2-byte sequence number inscribed at the start of the payload) at which unit the audio file will be segmented and transported, *packet-spacing* (msec) is the initial packet spacing used in the CBR transmission of the audio file, and *mode* specifies the congestion control mode: 0 (method A), 1 (method B), 2 (method C), and 3 (method D). The receiver, my_audio_rcv, has command-line arguments

%   my_audio_rcv *port-number log-file payload-size pb-del pb-sp buf-sz target-buf*

where *pb-del* is the initial playback delay (sec)—time delay from the arrival of the first audio packet—*pb-sp* (msec) is the time interval at which buffered audio is written to /dev/audio for playback (triggered by SIGALRM), *buf-sz* is the total allocated buffer space (bytes), and *target-buf* (bytes) is the target buffer level (i.e., Q^*). In the receiver's code structure, attention needs to be paid to the shared audio buffer—the SIGIO handler will write to the buffer when audio packets arrives whereas the SIGALRM handler will read from the buffer for audio playback—so that it does not get corrupted due to concurrent access (e.g., semaphores may be used to achieve orderly access). When audio packets, upon arriving, find the audio buffer full, they will be dropped. The sender logs the current sending rate λ (= 1 / *packet-spacing*), along with the time stamp from gettimeofday(), whenever a packet is transmitted. The receiver, upon receiving an audio packet from the sender (SIGIO handler) or dequeueing an audio packet at playback (SIGALRM handler), logs the current time stamp and buffer occupancy $Q(t)$ for off-line diagnosis. As always, measurement logs are written to main memory and flushed to disk at the end of the run to avoid overhead/slow-down stemming from disk I/O. To affect congestion control, the receiver transmits a 12 byte feedback packet containing Q^* (i.e., *target-buf*), $Q(t)$, and γ (in terms of time interval *pb-sp*, not rate) to the sender. Depending on the *mode* value, the sender will utilize the received information to institute the chosen congestion control.

Benchmark the application between two machines where a .au *audio-file* is provided (see TA notes), payload size is 1 KB, initial *packet-spacing* at the sender side is 100 msec, *pb-del* is 5 seconds, *pb-sp* is 50 msec, *buf-sz* is 75 KB, and *target-buf* is 50 KB. Perform one benchmark run per congestion control method. Plot the time series measurement logs using gnuplot: at the sender, $\lambda(t)$ against time t, and packet rate (pps) as a function of time at granularity 1 sec; at the receiver, the queue length time series that plots $Q(t)$ against t, and the received packet rate (pps) at 1 sec time granularity. Discuss your results and findings. Compare the audio quality perception with the numeric performance findings. Note that there is some degree of freedom in the selection of the congestion control parameters. Determine parameter settings for each method that you find work well (relatively speaking). Before running the receiver, make sure to set the audio controls using audioctl (see TA notes for instructions on how to use the command) so that audio can be output through a headphone jack in the PC (you can use your MP3 player, walkman, or any other compatible head set). Do not use external speakers (at least when streaming Britney *et al.*).