CS 690N (Park) **Project Topics (Part I)** **February, 2000**

1. MPEG Layer 3 Profiling and Benchmarking

MPEG Layer 3 (an audio standard for both MPEG I & II) has become a popular format for storing and streamining CD quality audio over the Internet. MP3 achieves the highest compression ratio (about 12:1 or 128 kbps), and as such is VBR. There are three main steps involved in the project. (1) *Traffic profiling.* Analyze MP3 data with respect to its traffic properties, i.e., burstiness and correlation structure at different time scales. (2) *Network streaming.* Transport MP3 data, in real-time, over the network using the AFEC real-time MPEG audio transport protocol (existing system), and benchmark performance. (3) *Real-time OS scheduling.* Extend QoS benchmarking by running the MPEG decoder under DUNES with real-time scheduling (existing system), and ascertain CPU requirements for VBR decoding.

[Group size: 3–4, duration: 4 weeks $(2 + 1 + 1)$]

2. Traffic Generator Suite

The Purdue Infobahn QoS testbed which is comprised of Cisco 7206 VXR routers is driven by five dedicated traffic generation engines. This project entails writing software tools that allow effective workload generation across a number of relevant dimensions—CBR traffic, on/off traffic with exponential on- and off-times (Markovian sources), on/off with heavy-tailed on-periods, reliable transport of files with heavy-tailed size distribution, and trace-driven traffic generation. The principal items comprise of: (1) *Traffic generators.* Write traffic generators over UDP, TCP, and directly over IP. A key requirement is that the traffic generator software be efficient and accurate. (2) *Performance monitoring.* Implement performance monitoring on the receiver side (with some coordination with the sender—i.e., traffic generator) which measures end-to-end throughput, delay, packet loss, and jitter on-line (so that it can be forwarded to traffic managers, including the sender) and off-line (logging into a file for later analysis).

[Group size: 2–3, duration: 3 weeks $(2 + 1)$]

3. MPEG II Video & Audio Synchronization

The current capture and transmission systems for MPEG video and audio developed in the Network Systems Lab do not explicitly handle synchronization of video and audio streams. Synchronization, as a first step, requires using the decoding and presentation time stamps (DTS and PTS) provided in MPEG II to properly pace the video decoding/presentation operation, which is then, in tandem, synchronized with the decoding/presentation of the audio part with the help of the MPEG II Transport Stream. The MPEG II video and audio players, if given as separate processes (which the lab implementations currently are), then require further scheduling and coordination of the two processes such that they are properly paced with respect to each other when scheduled on an OS. (1) *Implement MPEG video/audio synchronization.* There are two designs possible, one, a "maximally" MPEG II compliant implementation which uses the features available in MPEG II Packet Elementary Stream and Transport Stream, and two, a "user-level" implementation that does not interfer with MPEG II, but provides synchronization through orthogonal user-defined mechanisms. (2) *Gang scheduling.* When implemented as separate processes, the two streams' decoding and presentation needs to be paced with respect to each other, whose proper functioning requires the help of a real-time OS. Use DUNES' real-time scheduling capabilities and interface to facilitate synchronized execution of the two tasks.

[Group size: 3–4, duration: 4 weeks $(2.5 + 1.5)$]

4. MPEG I Port for NT for Real-time Audio Compression/Transport

AFEC's sender-side capabilities have been implemented under NT for real-time MPEG I video compression—by interfacing with the real-time compression engine Optibase which captures frames from a video camera—which is then streamed onto the network using AFEC to its receiver module across an IP network. The AFEC audio transport system is available for UNIX. Port the AFEC system under NT for MPEG I audio by interfacing with Optibase's API, then formatting and streaming the traffic via AFEC across the network to a host with an AFEC receiver (driving an MPEG audio player) on UNIX. The task will involve modifying the MPEG video implementation under NT to the same processing and transport for audio.

[Group size: 2, duration: 2 weeks]

5. QoS-sensitive Playback of Pseudo-real-time MPEG Video/audio using Dynamic Prefetching

Popular streamed video/audio players available on the Internet (e.g., RealAudio, RealVideo), use the technique of prefetching coupled with client-side buffering to affect a smooth presentation of the video or audio stream, at the expense of delaying the initial playback time into the future. Most systems are sensitive and susceptible to "network congestion" which causes noticeable periods of QoS breakdown (no video/audio feed at all, or jitterish output). Use adaptive prefetching/buffer control—as a function of end-to-end network state—to affect QoS-sensitive presentation of video/audio at receiver side. (1) *Adaptive prefetching/buffer control.* Implement adaptive prefetching and buffer control—algorithm will be conveyed/discussed separately—for AFEC system for MPEG audio transport. The same module should be portable to the AFEC MPEG video transport system. Check if generic modular extension is possible for RealAudio and other software systems. (2) *Benchmark performance under varying network conditions.* Transport information over shared router with cross traffic (supplied separately), and measure performance with respect to ability to "hide" network congestion from receiver. Goal: minimize initial playback time delay and client-side buffer capacity.

[Group size: 2–3, duration: 3 weeks $(2+1)$]

6. More Control

Those who cannot wait to jump into full-fledged traffic control, speak with me.