CS 690N (Park) **Project Topics (Part II)** **March, 2000**

1. Guaranteed Service Implementation over RSVP

RSVP is a signalling protocol that allows router resources to be allocated on a per-flow basis using a receiver-oriented reservation and set-up protocol (also called *Int-Serv*). This project will implement the end system functionality necessary to interact with Cisco IOS—Cisco's router operating system—and its implementation of RSVP. The system is to achieve three objectives: (1) implement a simple, peak-rate admission control (most suitable for CBR traffic) through which RSVP is accessed, (2) implement the end system support that initiates and terminates RSVP connections by interacting with Cisco IOS through RSVP signalling, and (3) evaluate performance using multiple CBR and VBR (i.e., peak-rate limited) based on existing traffic generators and QoS measurement tools.

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

2. Implement Scalar QoS Control over Assured Service in IOS

Assured Service is a specific differentiated service scheme defined by IETF where routers implement a certain per-hop behavior: as a function of the label value (called DSCP) contained in the DS field (i.e., precedence or TOS field) forward the packet to one of 12 possible configured RIO schedulers (variant of RED). By itself, this scheme exports one specific, ad hoc way of providing differentiated treatment of packets which is possible in a multitude of other ways without addressing the critical problem: how to effectively fashion end-to-end QoS so that users' QoS requirements can be satisfied (Assured Service assumes a global admission control procedure which is antithesis to achieving scalability).

The project consists of two steps: (1) configure the routers such that 12 classes are implemented by specification of RED threshold values, and (2) implement end-to-end label control which adaptively changes a connection's label value (DSCP in IP header) to achieve a user-specified target end-to-end QoS. Compare this scheme against that of implementing label switching over priority queues (also configurable under IOS).

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

3. MPEG I Port for NT for Real-time Video/Audio Transport over Futuretel Board

This is an extension of the Part I project where two additional tasks are incorporated: (1) port the AFEC system under NT for MPEG I video and audio implemented for Optibase MPEG compression engine to Futuretel's MPEG compression engine and its API, and (2) activate the AFEC feature of the NT implementation of real-time video transport for Optibase and perform benchmark experiments with respect to its end-to-end performance.

[Group size: 2, duration: 4 weeks]

4. Internet Information Map [not really networking]

The aim of the project is to discover and characterize the "Information Topology" of the Internet with respect to the connectivity structure of URLs in the World Wide Web. We can associate a directed graph structure to the World Wide Web in the following way. URLs are the *nodes* in the graph. A URL pointing to a hypertext or HTML document is an *internal* node if it contains hypertext links (or URLs) to other Web objects. Otherwise, the URL is a *leaf* (or terminal) node. There is a directed edge from URL A to URL B if A contains the URL B. The aforementioned definition is but a rough, first cut definition of the graph structure since CGL and Java scripts need to be suitably incorporated. In general, search engines can be viewed as implicitly maintaining hypertext links. For simplicity, we will exclude them from consideration in the graph structure.

What we are interested in is the characterization of the connectivity structure of the resulting information topology. Specific questions include: What is the distribution of out- and in-degrees of the information graph? (We would expect popular sites to have a huge in-degree.) Is the graph structure sparse, dense, in-between? Are there many disconnected components (i.e., islands)? What is their distribution? Are there characteristic differences between `.com`, `.edu`, `.org` and `.gov` sites? And so forth. The Information Topology project is motivated by the fact that recently researchers have discovered that the Internet topology (physical topology) exhibits heavy-tailed properties. In particular, if all nodes are ordered by their in- and out-degree (say in decreasing order), then as a function of rank r the in- and out-degree d obeys polynomial decay of the form $d \approx r^{-\alpha}$ where $\alpha > 0$ is some constant. It is expected that the structural property of graph structure, in particular, of the Information Topology, will yield important information relevant for network performance.

The project consists of two steps: one, write a breath-first search Web graph discovery engine that given a starting node A (i.e., URL) and depth ℓ commences to identify the Web structure rooted at A up to depth ℓ. It is important that the output file created encode the directed graph structure and relevant information in a succinct and compact way (e.g., adjacency list) such that graph traversing tools can be run with the output file as input to yield information about its connectivity structure. Of course, during the search process, care needs to be exercised not to fall into infinite loops, and the search should be fast and efficient (e.g., use of hash tables). Two, write tools that will compute output the various graph properties listed above, and check for trends, patterns, or identifiable laws in the resulting structure.

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

5. Implementation of Multiple Time Scale TCP

In this project, you will engage in limited modification of the TCP congestion control module in the protocol stack to implement the Slope Control feature of TCP-MT. TCP-MT, by monitoring the network state over larger time scales (e.g., 2 seconds) makes predictions about the contention level over the next 2 second time interval, and adjusts the slope of the linear increase phase commensurate with the predicted state. In particular, if the network state is predicted to be good, then the slope is set to a high value so that bandwidth is not unnecessarily wasted by conservative increase. On the other hand, if contention level is predicted to be high, then the slope is decreased to reduce the probability of frequent timeouts.

Most of the implementation code exists as module implementations in LBL's *ns* network simulator. The project consists of porting the *ns* based modification to TCP Reno code into the corresponding TCP congestion control module under Linux. Benchmarking is carried out by comparing the reliable throughput of TCP vs. TCP-MT under bursty traffic conditions.

[Group size: 2–3, duration: 4 weeks]

6. QoS-sensitive Playback of Pseudo-real-time MPEG Video/Audio: Part II

This project builds on the earlier project developed in Part I for QoS-sensitive playback of real-time MPEG video/audio using prefetching and adaptive client-side buffering. The current system implements variable prefetching to achieve a target client-side buffer occupancy which prevents blocked playback due to delayed or missing frames by maintaining sufficient frames in the buffer for continuous playback.

In Part II, the existing system is to be extended to incorporate error control which minimizes the impact of missing frames, or packets containing information belonging to a frame, which degrades QoS. The extension is to proceed on three lines: (1) integrate with adaptive FEC such that probability of missing frames is minimized, (2) integrate with ARQ using selective ACK over UDP to fill in missing packets, and (3) use ARQ—as implemented by TCP—with `write` system call at the sender passing variable payload to achieve reliability (i.e., user-level control using dynamic prefetching and client-side buffer management, currently implemented on top of UDP, is affected on top of TCP). Part (3) is a trivial exercise, and serves as a reference point when carrying out performance evaluations.

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