A Max-Min Fairness Congestion Control for Layered Streaming of Scalable Video

Hsu-Feng Hsiao, *Member, IEEE*, and Jenq-Neng Hwang, *Fellow, IEEE*

***Abstract*—In a best-effort networking environment, efficient and fair congestion control is highly desired for every traffic flow to share the bandwidth appropriately. This paper proposes a congestion control algorithm for user datagram protocol rate-based layered streaming of scalable video, e.g., 3-D wavelet based scalable video streaming, which provides a variety of video bit rates. This proposed congestion control mechanism, as an extension of explicit control protocol that is a newly proposed congestion control protocol believed to be superior to transport control protocol, accommodates both window-based and rate-based flows to the heterogeneous network environment which can include wired and wireless channels. This paper further introduces the notion of reserved packet length so that the traffic of layered video can better share the bandwidth of a network by taking account of the max-min fairness with other traffic.**

***Index Terms*—Explicit control protocol, layered video coding, max-min fairness, rate-based congestion control, reserved packet length (RPL).**

I. Introduction

IN A BEST-EFFORT networking environment, it is highly desired for every traffic flow to be regulated by some efficient and fair congestion control mechanism so that every traffic flow can have its appropriate share of bandwidth. The majority of the Internet traffic is from transport control protocol (TCP) window-based flows whose congestion control mechanism basically follows the additive-increase and multiplicative-decrease (AIMD) algorithm to adjust its window length for traffic regulation.

On the other hand, multimedia streaming, mainly consisting of video and audio data, emerges to be the main sources of traffic. Generally speaking, these kinds of applications have low tolerance of delay and jittering. Further, to overcome the scalability issues of video streaming dissemination over the heterogeneously networked receivers, the video-content providers would either prepare several copies of a video sequence, each at different bit rate, or have the video encoded in (scalable) layered structure so that a receiver can subscribe an appropriate copy based on its network condition and play the incoming video in real time. Therefore, we can conclude that flexibility in terms of video encoding scalability at bit stream level can facilitate the dissemination of video streaming over nowadays networks which has the characteristic of time-dependent available bandwidth. MPEG-4 fine grain scalability (FGS) video coding [1] makes use of bit plane encoding at DCT domain to offer arbitrary truncation ability at video bit stream. Since a MPEG-4 FGS encoder does not possess the knowledge of the data rate of the received enhancement layers, motion estimation/compensation is only implemented at base layer to avoid error drifting problems. To eliminate error drifting as well as to increase coding efficiency, 3-D wavelet motion-compensated temporal filtering (MCTF) [2] replaces the prediction along the time axis by a wavelet filter, which can nevertheless be operated in combination with motion compensation. Three-dimensional embedded subband coding with optimized truncation (3-D ESCOT) [3] emphasizes the frame-rate scalability and resolution scalability by coding the subband coefficients with fractional bit plane coding approach. This fine grain scalable coding scheme can be easily adapted to layered fashion by appropriately dividing the bit plane packets into layers. After the scalable video layers are formed, one common transport protocol to deliver the contents is real-time transport protocol (RTP) on top of user datagram protocol (UDP) [4], which basically provides neither guarantee on the quality of service (QoS) nor friendliness to the competing traffic. A popular companion to RTP is real-time transport control protocol (RTCP), which provides the feedback related to QoS index for a sender to regulate the data rate [5].

There is an extensive literature on the subject of rate-based congestion control for the applications of video/audio streaming [6]–[8]. Most of these algorithms follow the congestion control concept of TCP [9], which observes the round-trip time (RTT) and packet loss rate to implement AIMD mechanism, so as to achieve TCP-friendly rate control.

However, several studies in literature [10]–[12] have reported that TCP-based congestion control has fairness problems and has difficulty in utilizing the network resource efficiently over the high per-flow bandwidth-delay product networks. For fairness issues, one example is that even with two receivers sharing the same bottleneck of a network, the steady throughputs are different as long as their RTTs are different. In the TCP or TCP-friendly congestion control algorithms, packet drop is an important index to indicate network congestion. However, packet loss due to network congestion induces the price of longer queue length in the routers and thus it will lead to inevitable longer delay and jittering effect.

Many researches [12]–[15] have worked on the improvements of TCP congestion control. In the explicit congestion notification (ECN) protocol [13], the routers pass forward the

Manuscript received November 8, 2004; revised July 14, 2005. This paper was recommended by Associate Editor H. Sun.

H.-F. Hsiao is with the Department of Computer Science, National Chiao Tung University, Hsinchu 30010, Taiwan. R.O.C. (e-mail: hillhsiao@cs.nctu.edu.tw).

J.-N. Hwang is with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: hwang@ee.washington.edu).

Digital Object Identifier 10.1109/TCSVT.2006.881865

network condition using a single bit in the IP header to notify the presence of congestion. Explicit control protocol (XCP) [12], which can be regarded as an extension of ECN, explicitly fills in a few bits of the packet header with some details of the network condition to enable the window-based congestion control senders, in response to the feedback from the receivers, to quickly and accurately adapt to the assigned fair bandwidth. With the verification of our own ns2 simulations, as also reported in the work of [12], XCP proves to be much superior to TCP in terms of inter-session fairness and the convergence time to bandwidth variation, both in typical or high-speed (giga range) networks.

The traffic of layered video streaming, on the other hand, behaves quite differently from the window-based TCP (or XCP) flows. Layered video enables a receiver to decode and display a subset (usually the cumulative collection) of video layers it can receive. The resolution of the changes of data-rate is determined by the bandwidth of a video layer. A larger number of video layers generally ensure greater bandwidth resolution. Generally, depending on the coding method, there is a tradeoff between the number of layers and the video coding efficiency. Therefore, a sender of layered video cannot transmit an arbitrary data rate that is reported by congestion control algorithms, unless the assigned data rate equals the accumulated bandwidth of some video layers.

The objective of this paper is to define a congestion control mechanism for layered video streaming to approach inter-session fairness to the background traffic, which can include window-based flows, rate-based flows, and other layered multimedia traffic. The fairness criterion adopted here is the max-min fairness [16], which assigns the largest possible resource to the flow with lowest throughput. We examine and extend the aggregated behavior of available bandwidth of a node in window-based XCP to both window-based and rate-based congestion control and further we propose the reserved bandwidth strategy through reserved packet length (RPL) for layered video to fairly compete the traffic.

The remaining paper is organized as follows. In Section II, we give a quick review of related works. In Section III, we discuss the congestion control algorithm for both window-based and rate-based traffic as an extension of XCP. We propose reserved-packet-length embedded explicit control protocol (RPL-XCP) for layered video streaming in Section IV. In Section V we present simulation results, followed by the concluding remarks in Section VI.

II. Related Work

Many of the current congestion control algorithms work on the end-to-end points to adapt the sending rate of a controlled flow based on the observed network conditions, such as round-trip delay and packet loss rate. In the works of [8] and [17], the congestion control algorithms are TCP throughout equation-based, while in [6] and [7], the algorithms are based on the AIMD to approach inter-session fairness. Binominal congestion control algorithms in [41] present a generalized AIMD to enlarge the control scope. Comparison and scalability studies of equation-based congestion control and AIMD can be found in [18]–[20]. Congestion control algorithms that only rely on end-to-end points generally react to packet loss that serves as an indication of network congestion. However, packet loss usually leads to the degradation of application performances and also increases the packet transmission time due to the longer queuing delay. For the congestion control algorithms with retransmission mechanism, packet loss increases the chances of the retransmission which can further complicate the congested network. On the other hand, for networks that include wireless links, packet loss is no longer only due to network congestion but also can be from wireless errors, which can result from multipath fading, shadowing, or attenuation. Thus, the wireless packet loss can mistakenly lead to dramatic performance degradation for these kinds of congestion control algorithms.

Congestion control of packet-pair layered multicast (PLM) in [25] acquires estimated available bandwidth using packet pair technique [26] so that the optimal number of video layer to transmit can be determined. There are many other tools to estimate available bandwidth. Pathload [27] estimates the available bandwidth based on the trend of one-way delays of probing packets. The initial gap increasing (IGI) [28] method investigates the gap distances of probing packets to infer available bandwidth. Spread pair unused capacity estimate (Spruce) [29] also works on the distance of probing packets to estimate the bandwidth, while Pathchirp [30] uses a self-induced congestion concept. Such tools can be adopted to assist the rate control decision in a congestion control algorithm. Besides the accuracy concerns of estimated bandwidth, however, those tools in general rely on additional probing, which can be intrusive to the background traffic and they also require better synchronization of bandwidth adaptation among inter-sessions since each flow targets to consume the available bandwidth it estimates which shall have been shared by all the competing flows, instead.

Due to the limitations of end-to-end congestion control algorithms, congestion control algorithm involving information feedback from routers becomes more important and popular. A random early detection (RED) [21] router gives an early warning about network congestion by forced packet discarding. RED monitors the queue length to detect the potential congestion and probabilistically discards the packets before the router queue is full. Instead of dropping packets, ECN [13] enables routers to inform the end-to-end nodes explicitly about the network congestion by updating at some specified location of the packet header so that a sender can adapt to the changing network condition earlier. Based on the infrastructure of active networks [23], [24], the adaptive rate control algorithm for layered MPEG4-FGS video in [22] utilizes the explicit router information to facilitate congestion control mechanism, with the cost of extra memory usage in the routers.

XCP [12] suggests a stateless congestion control model for window-based rate control, which decouples congestion control from bandwidth allocation decision making. There are two modules involved in an XCP: the efficiency controller and the fairness controller. An XCP sender fills in some fields of the congestion header, such as window size and RTT, while the routers between the sender and its receiver update a congestion feedback field in the header by means of the efficiency controller and the fairness controller, without the need of per-flow

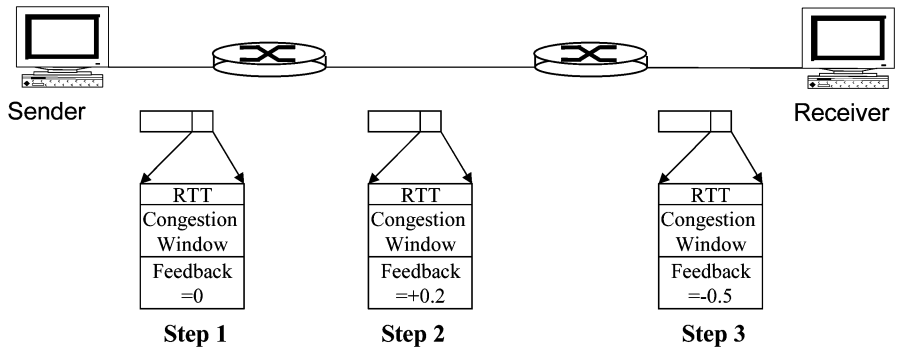


Fig. 1. Illustration of the updating of the feedback field of packet header by routers. At step 1, the feedback field is zero. At step 2 and 3, a router can annotate the field according to the feedback calculation locally. The fields of RTT and congestion window are maintained by the sender.

operation. An XCP receiver functions similarly to a TCP receiver with the addition that it uses those ack packets to transmit the congestion feedback for the sender to adapt the congestion window to the fair throughput. Based on this concept of explicit congestion information, we have proposed a congestion control mechanism [31], RPL-XCP, which works on both window-based and rate-based flows as well as the flows which can only have quantum changes of sending rate such as the layered streaming of scalable video.

III. Congestion Control Model

This section provides an overview of the congestion control model [12], [31] for both window-based and rate-based flows. The metric of this extended congestion control includes five factors: stability, scalability, efficiency, fairness, and suitability.

For stability, the throughput of a flow regulated by the congestion control algorithm shall converge to its equilibrium point in a reasonable time. For scalability, a congestion control mechanism shall work well even if the number of inter-sessions grows greatly. For efficiency, the total effective throughput of a link shall be close to the link capacity. For fairness, network resource shall be shared fairly between different flows. The definition of fairness considered here is max-min fairness [16]. For suitability, some application-related requirements need to be fulfilled. For the layered streaming of scalable video, the applications are sensitive to the packet delay and the delay jittering. The goal of the sensitivity metric here is to minimize packet delay and jittering which is measured by the first-order of delay deviation.

A. Window-Based XCP

The window-based XCP [12] requires the cooperation among a sender, its receiver, and the routers between the sender and receiver. A sender maintains its congestion window and RTT so that the routers between the end-to-end points can acquire those parameters via the congestion header in every packet. The routers monitor the difference between the link capacity and the input rate to each of the outgoing queues, and then ask the flows to increase or decrease their congestion windows by annotating the congestion header of the packets. The aggregated feedback is divided among the flows through their congestion window and RTT values by the AIMD principle. The feedback field of the congestion header only carries the feedback from the most congested router along the path, as illustrated in Fig. 1. The receiver passes back the congestion header to the sender via ack packet so that the sender can change its congestion window accordingly. For more details, please refer to the description in [12] and [31].

B. Modified Aggregate Traffic and the Efficiency Controller

Similar to XCP, a router first determines the *aggregate feedback* ϕ for all the flows to adapt, so that the link utilization can be maximized and the probability of network congestion can be minimized.

The simplest way for a router to infer spare bandwidth of a given network node would be $(C_{\text{out}} - R_{\text{in}})$, where C_{out} is the outgoing-link capacity and R_{in} is the incoming rate to that queue. Since the queuing delay is the primary source of the one-way packet delay and the delay jittering, we take account of the queue length Q_{end} at the end of each control period d when calculating the aggregate feedback so that the persistent queue length and the queue introduced by busty traffic can be consumed in the next control period, and thus one-way packet delay and jittering can be minimized. The final aggregate feedback ϕ is

$$\phi = \alpha(C_{\text{out}} - R_{\text{in}}) - \beta\frac{Q_{\text{end}}}{d} \tag{1}$$

where α and β are constant parameters for the stability consideration and d is measured by the average RTT. From the analysis of Nyquist plot in [12], it concludes that for system to be stable, α and β shall satisfy the following conditions:

$$0 < \alpha < \frac{\pi}{4\sqrt{2}}, \beta = \alpha^2\sqrt{2}. \tag{2}$$

However, in [12], only the persistent queue $Q_{\min}$, measured as the minimum queue size during the last propagation delay, is considered. Since $Q_{\text{end}} \geq Q_{\min}, Q_{\text{end}}$ in (1) not only contributes absorbing the packets in the queue but also increases system stability due to the more conservative calculation about the aggregate feedback. Note that the aggregate feedback in (1) holds for both window-based and rate-based network protocols since the calculation of R_{in} and Q_{end} only concerns packet length.

C. Fairness Controller for Rate/Window-Based Flows

The fairness controller is based on the AIMD principle. Specifically, when $\phi > 0, \phi$ is divided equally to all the flows. Otherwise, it is allocated to the flows proportionally according to their current throughputs when $\phi < 0$. The aggregate feedback information is an aggregated calculation of the packets passing through the node. To avoid keeping track of the per-flow status in a router, which can only annotate the traffic information in each packet's feedback field, the feedback information for each packet needs to be derived from the aggregate feedback ϕ without the knowledge of the flow status. The *per-packet feedback* is expressed as the difference of the positive feedback p_K from the negative feedback n_k [32]

$$p_k = \varepsilon_p \cdot \frac{s_k}{r_k}, \quad \text{where } \varepsilon_p = \frac{h + \max(\phi, 0)}{\sum \frac{s_i}{r_i}}$$
$$n_k = \varepsilon_n \cdot s_k, \quad \text{where } \varepsilon_{\mathrm{n}} = \frac{h + \max(-\phi, 0)}{\sum s_i} \tag{3}$$

where s_k is the packet length (available in the packet header) of packet k; r_k is the flow throughput specified either implicitly in the window-based XCP header or explicitly in the header of our rate-based flows and layered video streaming flows. To avoid convergence stalling at $\phi = 0$, *bandwidth shuffling* is introduced in [12], where h is the shuffled data rate which is about 10% of the traffic. Equation (3) shows that actually this fairness controller works most naturally for rate-based flows.

Simple translation between the congestion window for window-based flows and the data rate for rate-based flows that needs to be indicated in the congestion header can be derived from the following equation:

$$r = \mathrm{CWND} \cdot s/\mathrm{RTT} \tag{4}$$

where r is the throughput of the rate-based flow, CWND is the corresponding congestion window measured in number of packets, s is the packet size, and RTT is the round-trip time. The concept behind (4) is that in the time period RTT, for a window-based flow, there are CWND packets being transmitted. Therefore, the equivalent transmission rate can be expressed as in (4).

Once all the per-packet feedbacks are collected by the sender of rate-based applications, the total change in the throughput of a flow is equal to the sum of the per-packet feedbacks it receives for that flow. In the next section, we will further consolidate the per-packet feedbacks for the layered streaming of scalable video.

IV. Congestion Control Strategy for Layered Streaming of Scalable Video

In the XCP protocol, when a sender receives the per-packet feedback from the returning ack packets, it is supposed to update the congestion window right away to reflect its share of spare bandwidth along the end-to-end path. For the case of rate-based congestion control protocol, it can be done by explicitly updating the sending rate. However, for layered-structure data, the possible transmission rate can only be an element of the set which is composed of some pre-defined discrete values. A sender cannot increase the sending rate before its accumulated per-packet feedback plus the current sending rate is greater than the data rate of next level. When the feedback is negative, it needs to reduce the sending rate at the resolution of layer bandwidth which is no less than the amount suggested by the feedback information.

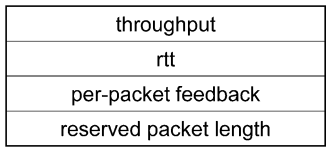


Fig. 2. Congestion header with RPL.

Therefore, for the XCP window-based protocol, it is difficult to adapt to the staircase-like bandwidth distribution of layered video. If the assigned spare bandwidth of a layered video flow can not be reserved somehow, this spare bandwidth will be re-assigned to all the flows in the next control iteration. The layered video might not have chance to actually increase the number of video layers being transmitted.

There are several applications that require layered-structure data transmission, such as *simulcast streaming* which uses various streams at different bit rates and *layered video streaming* which delivers the videos encoded in a layered (scalable) structure so that the video receivers can acquire an appropriate cumulative subset of generated video layers based on the network condition and play the incoming video in real time. Video codecs for layered video streaming or simulcast streaming include MPEG4-FGS [1], [33], H.264 SP/SI coding [34], and 3-D wavelet video coding [2], [3], [35]. In this section, we are interested in the applications of layered streaming of 3-D wavelet scalable video. For the cases of streaming simulcast and layered streaming of other scalable video codecs, the concepts of congestion control algorithm discussed in this section can also be easily applied.

To preserve the spare bandwidth before the sender can actually transmit a new layer to its receiver, we introduce the concept of RPL Δs_k as the additional information placed in the packet congestion header, as shown in Fig. 2, so that when a router receives packets and calculates the bandwidth, it will take RPL into account. Field "*throughput*" is the current transmission rate of that flow. To be consistent with the window-based XCP, the measure of the throughput is in window size; as a result, for rate-based flows, we use (4) to translate the explicit throughput to the corresponding window size. Field "RTT" is the current RTT estimated by the moving average at the sender. Field "*per-packet feedback*" is updated by the routers using (3), where proper translation by (4) is also required for the coexistence of the window-based and the rate-based flows. When the routers calculate the per-packet feedback for each incoming packet, as suggested in (1) and (3), the packet length s_k is replaced by $(s_k + \Delta s_k)$. The information Δs_k written in the field RPL of the congestion header shall be always greater than or equal to zero. When the calculated Δs_k is less than zero, the sender needs to withdraw a layer (or a number of layers)

and readjust the RPL Δs_k, which will be further discussed in Section IV-B.

A receiver's duty in the RPL-XCP algorithm is to send back the per-packet feedback about the network condition. To alleviate the feedback implosion situation, the RPL-XCP receivers can consolidate the feedback packets by sending back only an ack packet over time duration t_d. This ack packet carries the information of the *accumulated per-packet feedback* $P_{\rm sum}$ as expressed in the following equation and the sender can also update the RTT value upon receiving this ack packet

$$p_{\rm sum} = \sum_{\rm t_d} (p_k - n_k). \tag{5}$$

Longer duration t_d reduces the amount of feedback packets and increases the responsive time to react to the assigned bandwidth. Before the sender acquires the first consolidated *ack* packet so as to determine the RTT value, a pre-defined initial value is used. Alternatively, the RTT value from the initial control signal (TCP packets) to set up this video streaming application can be used as the initial RTT value for the UDP session.

When the sender receives a new accumulated feed-back $P_{\rm sum}$ that is annotated in the congestion header of this *ack* packet, the recursive equation to update the RPL Δs_k is:

$$\Delta s_k \Leftarrow \Delta s_k + \delta s_k. \tag{6}$$

From the routers' point of view, the difference of the receiving rate introduced by δs_k should be equal to the accumulated feedback. Specifically

$$p_{\rm sum} = N_k \cdot \delta s_k \Rightarrow \delta s_k = \frac{p_{\rm sum}}{N_k} \tag{7}$$

where N_k is the *packet rate* (number of packets per second) transmitted to the receiver, which can be derived as follows:

$$N_k = \frac{r_k}{s_k}. \tag{8}$$

The resulting receiving rate $\tilde{r}_k$ of flow k observed by the routers between the end-to-end points, taking account of the RPL Δs_k, becomes

$$\tilde{r}_k = r_k + N_k \cdot \Delta s_k = r_k \cdot \left(1 + \frac{\Delta s_k}{s_k}\right). \tag{9}$$

With the RPL in the loop, the shared spare bandwidth can be actually preserved by recording Δs_k in the congestion header instead of being taken away by other flows in the next control iteration.

For clearer discussion's sake in the later subsections without the loss of generality, we assume every 3-D wavelet video layer has the same bandwidth r and packet size s. This assumption is not necessary in our model but just for easier discussion purpose. We also assume that the receiver currently has m video layers. It is important to determine when to add a new layer (or layers) and when to discard a layer (or layers) with respect to the changes of RPL Δs_k.

$$\delta s_k = \frac{p_{\rm sum}}{N_k} = p_{\rm sum} \cdot \frac{s}{mr} \tag{10}$$

$$\tilde{r}_k = \frac{mr}{s} \cdot (s + \Delta s_k). \tag{11}$$

In a sender's point of view, the additional RPL δs_k upon the receiving of the accumulated per-packet feedback as shown in (7) can be determined by (10). The transmission rate plus the reserved bandwidth can be rewritten as in (11).

A. Increase of Video Layers

Assuming the reserved bandwidth by the RPL is enough for at least one extra layer; the current transmission rate of m layers plus the reserved bandwidth shall be greater than or equal to the data rate of $m+1$ layers as expressed in (12) and thus Δs shall satisfy (13)

$$\frac{mr}{s}(s + \Delta s) \geq (m+1) \cdot r. \tag{12}$$

$$\Delta s \geq \frac{s}{m} \tag{13}$$

Therefore, the criteria for the sender to start to transmit $i(\geq 0)$ additional layers to its receiver is to examine if (13) holds. Once the number of transmitted layers m is indeed increased, the new RPL $\hat{\Delta}s$ should also be updated accordingly to reflect this increase of video layers. The observed data rates taking account of the RPL by a RPL-XCP router shall be equal, before and after the layer increase, as

$$\frac{mr}{s}(s + \Delta s) = \frac{(m+i)r}{s}(s + \hat{\Delta}s) \tag{14}$$

$$\hat{\Delta}s = \frac{m \cdot \Delta s - i \cdot s}{m+i}. \tag{15}$$

Due to the fact that the updated $\hat{\Delta}s$ needs to be greater than or equal to zero as expressed in (16), we can derive number i as the largest integer satisfying (17). This largest integer stands for the largest number of additional video layers to transmit at the sender side

$$\hat{\Delta}s = \frac{m \cdot \Delta s - i \cdot s}{m+i} \geq 0 \tag{16}$$

$$i \leq \frac{m\Delta s}{s}. \tag{17}$$

B. Decrease of Video Layers

On the other hand, when $\Delta s < 0$, which indicates that the sender needs to withdraw $i(\leq m-1)$ layers from being sent to its receiver. As a result, the bandwidth would be reduced by $(r \cdot i)$. The reduction of transmission bandwidth $(r \cdot i)$ could be more than the indicated feedback reported by the routers. To compensate for this possible over-reduction, a new RPL $\hat{\Delta}s$ can be derived from (18) and (19)

$$\frac{mr}{s}(s + \Delta s) = \frac{(m-i)r}{s}(s + \hat{\Delta}s). \tag{18}$$

$$\hat{\Delta}s = \frac{m \cdot \Delta s + i \cdot s}{m-i}. \tag{19}$$

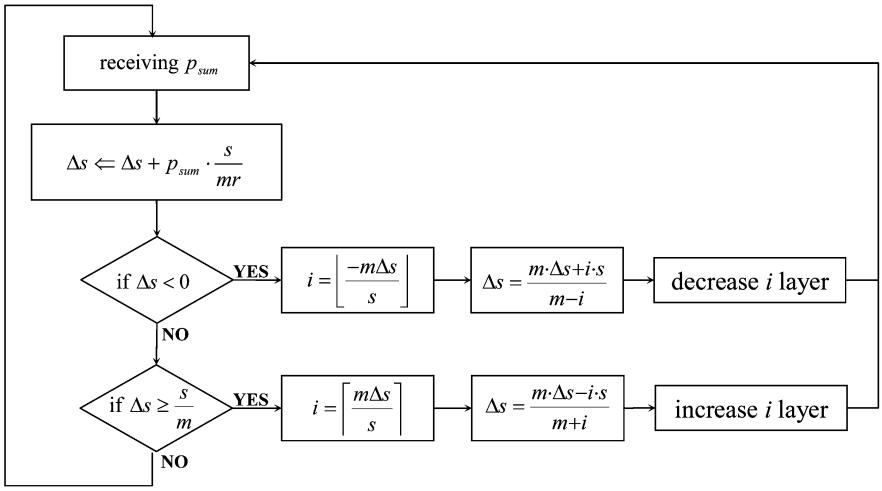


Fig. 3. RPL-XCP control loop for the senders of scalable layered video.

Similar to last subsection, because the new $\hat{\Delta}s$ needs to be greater than or equal to zero as expressed in (20), hence, i would be the smallest integer satisfying (21) and it stands for the smallest number of video layers that the sender needs to withdraw in order to conform the regulation of RPL-XCP congestion control

$$\hat{\Delta}s = \frac{m \cdot \Delta s + i \cdot s}{m - i} \geq 0. \tag{20}$$

$$i \geq \frac{-m\Delta s}{s}. \tag{21}$$

Therefore, the congestion control strategy of RPL-XCP on the sender's side can be summarized as the following block diagram (Fig. 3). The RPL-XCP algorithm for the sender of 3-D wavelet layered streaming is actually quite light-weighted in terms of calculation effort.

C. Impact on the Delay-Sensitive Applications

For delay-sensitive applications such as streaming video/audio, it is desirable to receive such multimedia packets at timely manner, i.e., one of the congestion control objectives is to reduce packet delay time and delay jittering as much as possible. First, we examine the source of the end-to-end packet delay and discuss how the algorithm of RPL-XCP can reduce the packet delay as well as the delay jittering.

The end-to-end one-way packet delay can be modeled as the summation of the following quantities: propagation delay for the electromagnetic waves to traverse all the link media along the end-to-end path, queuing delay which is the main cause of congestion that leads to the packet loss, and router processing delay which is required for the routers to multiplex, reassemble, and forward packets. Propagation delay and router processing delay are usually constant for a given end-to-end path and a given packet length. The model of packet delay can be summarized by the following equation:

$$T_d = \sum_i T_{q,i} + \sum_i \frac{P_s}{C_i} + \sum_i T_{p,i},$$
$$T_{p,i} \propto P_s, \tag{22}$$

where T_d is the one-way packet delay; $T_{q,i}$ and C_i are the queuing delay and capacity of link $i,$; $T_{p,i}$ is the router processing delay, which is proportional to the packet size as expressed in (22); and P_s is the packet size. We can further simplify (22) to (23) and then (24), where ε_i and ε are the time-invariant proportion coefficients as long as the routing path is not changed

$$T_d = \sum_i T_{q,i} + \sum_i \frac{P_s}{C_i} + \sum_i \varepsilon_i \cdot P_s \tag{23}$$

$$T_d = \sum_i T_{q,i} + P_s \cdot \varepsilon \tag{24}$$

where

$$\varepsilon = \sum_i (\varepsilon_i + \frac{1}{C_i}).$$

The deviation of T_d can be considered as the first-order absolute difference between T_d and the mean value of T_d as expressed in (25)

$$|T_d - \overline{T}_d| = \left| \left(\sum_i T_{q,i} - \overline{\sum_i T_{q,i}} \right) + (P_s - \overline{P_s}) \cdot \varepsilon \right|. \tag{25}$$

Equation (24) shows that for a fixed routing path, the packet one-way delay comes from the contribution of queuing delay at each link plus a fixed bias. For delay deviation, the relationship in (25) shows that only the deviation of the queuing delay will contribute to the variation of delay deviation, given the time-invariant packet length.

For a given packet length, what a congestion control algorithm can do to reduce the packet one-way delay and the delay jittering is to reduce queuing delay $T_{q,i}$. In the following section, we will show that RPL-XCP exhibits short queuing delay and negligible deviation of packet delay.

V. Network Simulations

We conduct simulations in this section to investigate the proposed RPL-XCP algorithm at various aspects. The re-

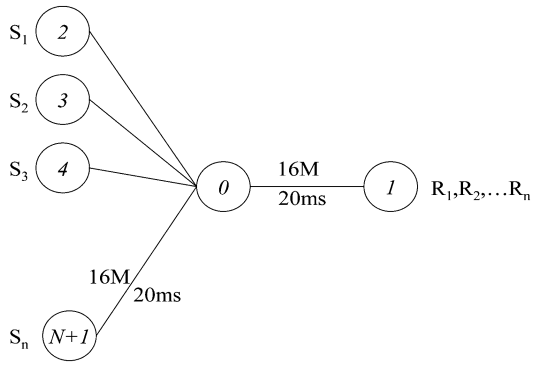


Fig. 4. Basic topology.

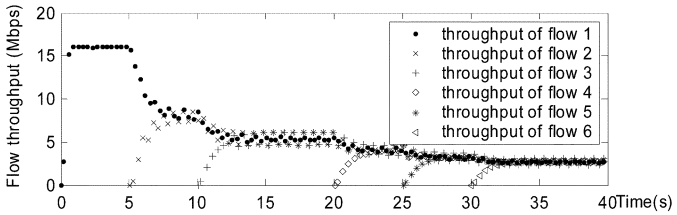


Fig. 5. Six window-based XCP flows start their transfers at times 0, 5, 10, 20, 25, and 30 s.

maining section is organized as follows. We first evaluate the performance of modified rate-based XCP flows as presented in Section III. We then consider the performance of the rate-based RPL-XCP flows of layered streaming of 3-D wavelet scalable video with the techniques of RPL described in Section IV.

A. Rate-Based Explicit Control Protocol

A basic network protocol we use is shown in Fig. 4. For the bottleneck between node 0 and node 1, the link capacity is 16 Mbps and the link delay is 20 ms. The N senders located (from S_1 to S_n) at node 2 to node $N+1$ and their receivers located at the other side of the bottleneck link. The link capacity and link delay of all other links are the same as of the bottleneck. The available queue length is 150 packets and random early detection (RED) [21] is employed with the minimum and maximum thresholds set to one-third and two-thirds the queue size, respectively. The packet length is 1000 bytes for both window-based and rate-based XCP simulations.

In Fig. 5, we show the convergence property of window-based XCP. In Fig. 6, we have 6 rate-based XCP flows $(N = 6)$ which initiate their transmission at times 0, 5, 10, 20, 25, and 30 s. The mechanism of suggested rate-based XCP is addressed in Section III. The simulation results in Fig. 6 show that rate-based XCP exhibits high link utilization and small queue size. The queue size from the beginning of transmission to the end of simulation is much smaller than the minimum threshold of RED and there is no packet drop for any of the flows. The throughputs of those rate-based XCP flows show that each flow converges to its equilibrium point rather fast and the max-min fairness is reached. The convergence of the rate-based XCP flows is even better than the window-based XCP flows as shown in the flow throughputs of Figs. 5 and 6.

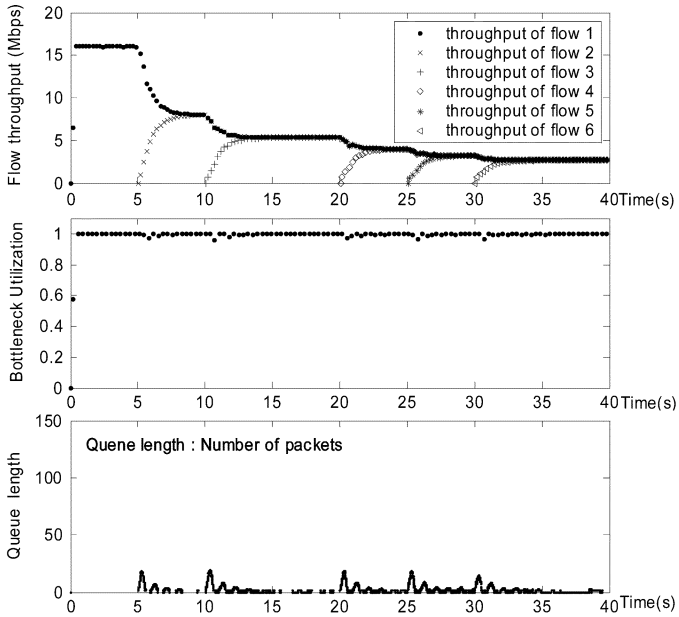


Fig. 6. Six rate-based XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s, achieve good fairness, high utilization, and relatively small queue size.

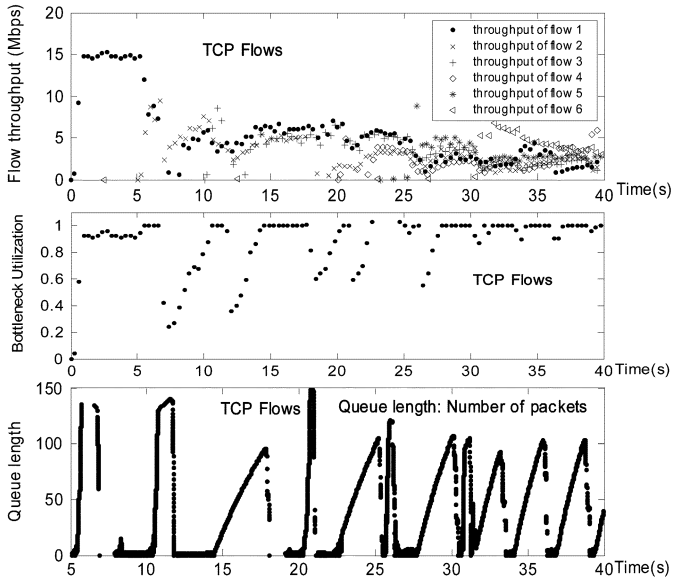


Fig. 7. Six TCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s, show poor fairness, frustrating bottleneck utilization, and large queue size.

We use the same network topology (Fig. 4) to show the performance of TCP flows. The TCP packet length is 1000 bytes. From the simulation results in Fig. 7, those flows can not converge quickly enough. The bottleneck utilization is not stable. Numerous packets are discarded and the queue length remains pretty large. All the receivers in this network topology have the same round-trip link delay. In the case of different round-trip link delay, the TCP fairness will be worse.

In Fig. 8, there are three window-based XCP and three rate-based XCP flows starting their transmission alternately. Three window-based XCP flows begin their transfers at times 0, 10, and 25 s while the rate-based XCP flows start their transfers at times 5, 20, and 30 s, respectively. It shows that the window-

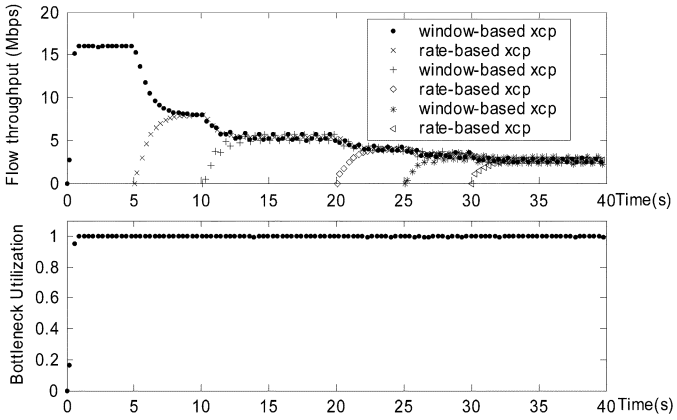


Fig. 8. Six rate-based and window-based XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s, achieve good fairness and high utilization.

based and rate-based XCP flows can coexist perfectly with high utilization and good inter-session fairness.

B. Layered Streaming Congestion Control

In this section, we examine the performance of RPL-XCP for data that is prepared in layered structure, such as the data of layered streaming of 3-D wavelet subband video coding. We use the network topology, which is the same as of the one of last section in Fig. 4, to evaluate the convergence speed, stability, efficiency, and fairness properties of RPL-XCP.

The layered data has the base layer at 500 Kbps and each additional layer at 1000 Kbps. Packet length is 1000 bytes. In Fig. 9, we have 6 rate-based RPL-XCP flows $(N = 6)$ which initiate their transmission at times 0, 5, 10, 20, 25, and 30 s. The mechanism of RPL-XCP is addressed in Section IV. It shows that the rate-based RPL-XCP algorithm exhibits good inter-session fairness and extremely short queue length. There is not any packet discarded. The throughput of rate-based RPL-XCP shows that each flow converges to its equilibrium layers fast and the max-min fairness is reached. The graph of layer subscription in Fig. 9 shows the instances where the number of subscribed layers is being updated along the time axis and it reveals quite a stable RPL-XCP layer subscription with only a few layer oscillations.

The queue length is even smaller than that of the rate-based XCP flows in Fig. 6. This is because the reserved bandwidth from RPL can help to consume the packets in the queue. For the packet-delay deviation of the RPL-XCP flows, the distribution is quite similar for those six flows. Since we do not assume globally synchronized clock between a sender and its receivers, we use the relative one-way trip time (ROTT), which is measured by a receiver as the time difference between the receiving time and the packet sending time stamp plus a fixed bias. The deviation of packet delay will be equal to the deviation of ROTT, given that the bias is a fixed value. The deviation of ROTT is obtained by calculating the following equation [36] with $\alpha = 1/32$

$$\begin{aligned} \mathrm{ROTT}_{\mathrm{mean}} &= (1-\alpha) * \mathrm{ROTT}_{\mathrm{mean}} + \alpha * \mathrm{ROTT}, \\ \mathrm{ROTTdev} &= (1-2\alpha)\mathrm{ROTTdev} \\ &\quad + 2\alpha * |\mathrm{ROTT} - \mathrm{ROTT}_{\mathrm{mean}}|. \end{aligned} \tag{26}$$

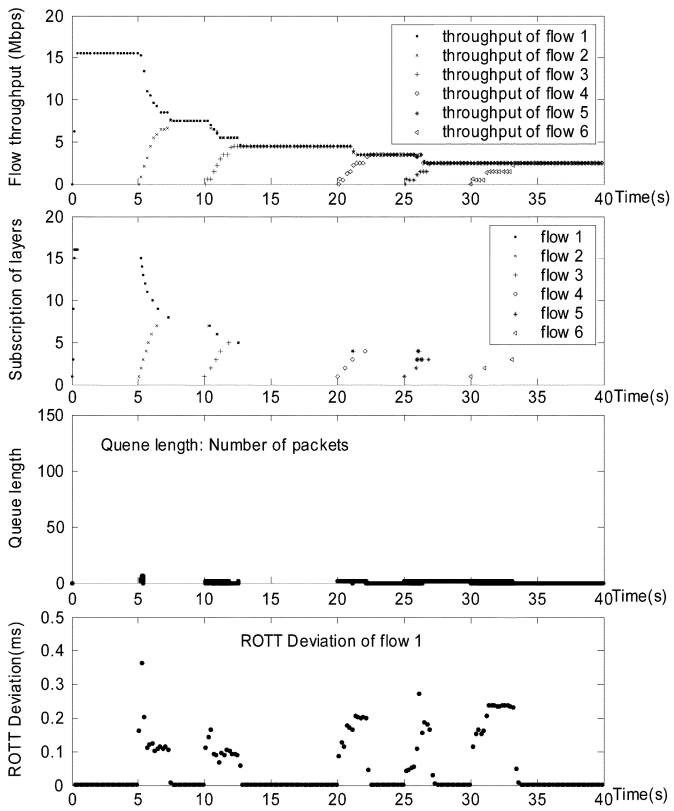


Fig. 9. Six rate-based RPL-XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s, achieve good fairness. The queue length at bottleneck is extremely short and the deviation of ROTT can be ignored for video applications.

Only the ROTT deviation of flow 1 is shown in Fig. 9 for clear illustration. The maximum value of the deviation of ROTT is less than half a millisecond, which is quite ignorable for the layered video streaming applications.

C. Robustness of RPL-XCP

Simulations presented above show the efficiency and convergence properties of the rate-based XCP and RPL-XCP flows. In this section we will examine the robustness of RPL-XCP in terms of compressed feedback dynamics, stability to the number of RPL-XCP flows, and the robustness for different round-trip delays. The metrics we evaluate here are the bottleneck link utilization, the inter-session fairness, and the packet loss counts.

We adopt the fairness index f_i [37] to evaluate the inter-session fairness between multiple sessions of the same bottleneck. f_i is defined by (27)

$$f_i = \frac{\left(\sum_{k=1}^{N} R(k)\right)^2}{N \cdot \sum_{k=1}^{N} R^2(k)} \tag{27}$$

where $R(k)$ is the throughput of kth session and N is the number of different sessions. Assume that the demand of the flow throughput is infinite, we say max-min fairness is achieved when f_i approaches unity, i.e., the throughputs of all sessions are equal.

The bottleneck utilization in this section is measured as the average of bandwidth utilization (the summation of throughputs

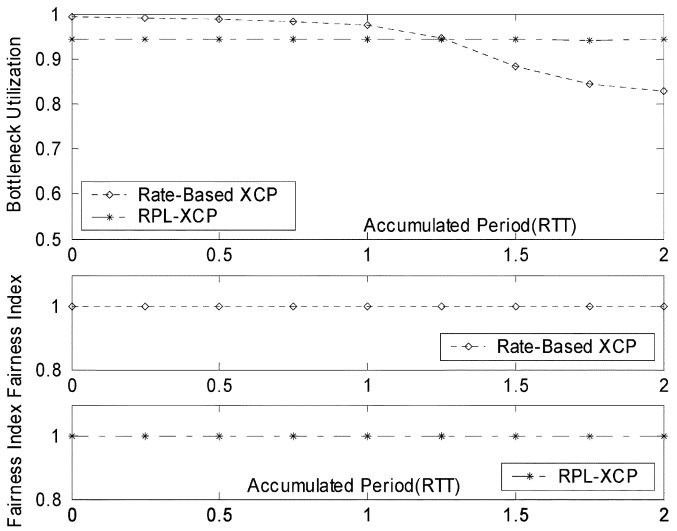


Fig. 10. Bottleneck utilization and fairness index of six rate-based XCP/RPL-XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s, versus various accumulated periods.

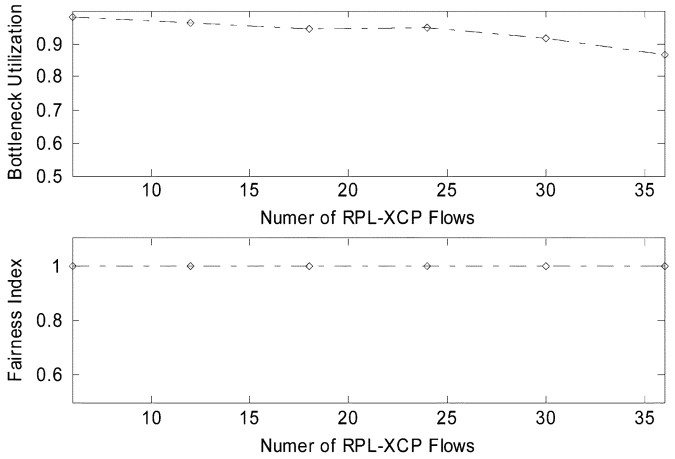


Fig. 11. Bottleneck utilization and fairness index of six rate-based XCP/RPL-XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s.

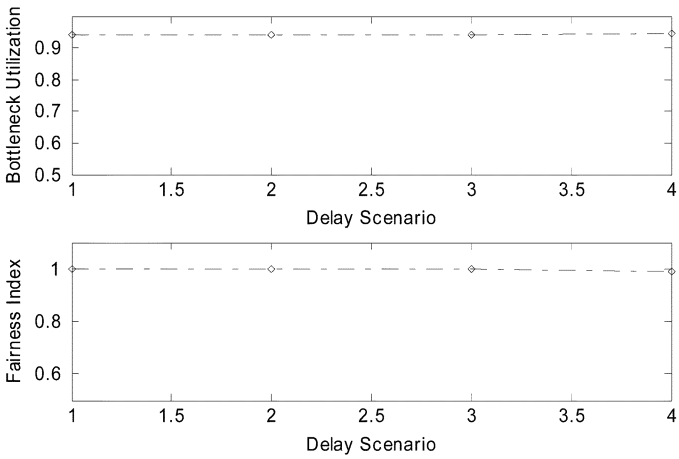


Fig. 12. Bottleneck utilization and fairness index of six rate-based RPL-XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s at different RTT scenarios.

of all flows over link capacity) of the bottleneck router. The packet drop counts are the number of total packet drops of the target flows in that event.

1) Compressed Per-Packet Feedback: Time duration t_d which per-packet feedbacks are accumulated controls the compression ratio of ack traffic and also the responsive time to the assigned bandwidth. We use the same topology as shown in Fig. 4 for more simulations to evaluate the performance at different compression ratios in terms of the bottleneck utilization and the inter-session fairness for the six rate-based XCP flows and six rate-based RPL-XCP flows (layer bandwidth is 500 Kbps), respectively. The bottleneck utilization and fairness index are measured in the state of equilibrium. From Fig. 10, both scenarios show promising results with high utilization and perfect fairness measured by the fairness index indicated in (27). Especially for the RPL-XCP flows, performance remains constant with regard to a wide range of accumulated period (measured from 0.0 to 2.0 times of the RTT). It reveals that the reserved bandwidth plays a key role to stabilize the throughput variation. Since the bandwidth of an RPL-XCP flow is discontinuously distributed, it prevents the bottleneck utilization from fully utilized (1.00).

Also, an important advantage of the rate-based XCP/RPL-XCP algorithm is that there is not a packet discarded from the start to the end of the flow transmission.

2) Impact of the Number of RPL-XCP Flows: We again use the topology shown in Fig. 4, but the link capacity is 50 Mbps to simulate the behavior in terms of the flow number. The flows, which are RPL-XCP at layer-bandwidth resolution 500 Kbps, start at one of the following times 0, 5, 10, 20, 25, and 30 s. In Fig. 11, it shows that the RPL-XCP algorithm performs quite well in terms of the bottleneck utilization and it also exhibits excellent fairness among all these flows. Note that since these flows are layered 3-D scalable wavelet video data, it is rare for the utilization to reach unity due to the discontinuous bandwidth allocation. Some of the bandwidth is reserved until the sender can transmit one more layer while we only consider real bandwidth in the calculation of bottleneck utilization. The simulations in this environment also show no packet loss and reasonably small queue size. Hence, the deviation of ROTT is quite similar to the one in Fig. 9.

3) Impact of Various RTT Scenarios: It is well-known that the throughput of a TCP follow is closely related to its RTT; as a result, flows with different RTTs will possess different throughputs. We use the topology as shown in Fig. 4 with the link capacity 16 Mbps to simulate the RPL-XCP behavior at various RTTs. There are six RPL-XCP flows, at layer-bandwidth resolution of 500 Kbps, which start at 0, 5, 10, 20, 25, and 30 s, respectively. For the first delay scenario, the total link delays for these flows are {80,80,80,80,80,80} ms. The second scenario is {80,160,240,320,400,480} ms. The third one is {80,280,480,680,880,1080} and the fourth one is {80,480,880,1280,1680,2080} ms. In Fig. 12, it shows that RPL-XCP performs quite well in terms of the bottleneck utilization and fairness index, irrelevant to RTT values.

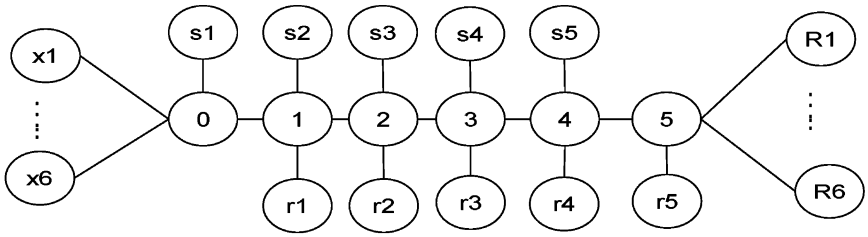


Fig. 13. Complex network topology with multiple congested links.

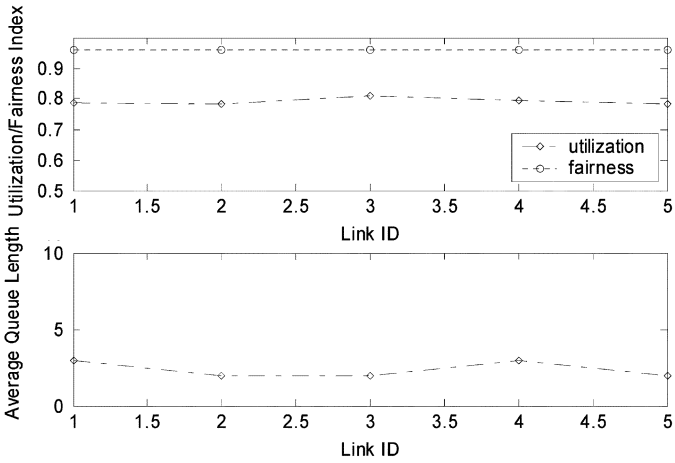


Fig. 14. Bottleneck utilization and fairness index of six rate-based RPL-XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s at different bottleneck links.

4) Impact of the Multiple Congested Queues to RPL-XCP: We examine the behaviors of RPL-XCP over complex network topology with multiple congested links as shown in Fig. 13. The capacity of each link is 30 Mbps with link delay 20 ms.

There are six RPL-XCP flows from node $(x1$–$x6)$ to node $(R1$–$R6)$. Also, we have cross traffic of five RPL-XCP flows at link1 (from node $s1$ to $r1$), link 2 (from node $s2$ to $r2$), link 3 (from node $s3$ to $r3$), link 4 (from node $s4$ to $r4$), and link 5 (from node $s5$ to $r5$). As a result, besides the target flows, there are totally 25 RPL-XCP flows competing different bottlenecks.

In Fig. 14, the link utilization at different bottlenecks and the fairness index remain quite high even with multiple congested links. Again, for RPL-XPL flows, due to the layered data structure of 3-D scalable wavelet video, the utilization is limited by the layer bandwidth resolution, the number of competing flows, and the link bandwidth, as also shown in previous sections. The average queue length of each bottleneck is really small (not more than five packets.)

D. RPL-XCP Over Hybrid Networks With Wireless Loss

In this section we evaluate the RPL-XCP performance in the network topology shown in Fig. 4, with the last links being wireless channels. In a wireless network environment, packet loss is inevitable regardless the designs of congestion control algorithms. Common wireless channel errors can be from multiplath fading, shadowing, or attenuation.

We use Gilbert/Elliot's two-state Markov chain model [38] to simulate the fading phenomena of wireless channels, which usually exhibits bursty errors. The model assumes two states, good and bad channels, with the following transition matrix:

$$P = \begin{pmatrix} P_{gg} & P_{gb} \\ P_{bg} & P_{bb} \end{pmatrix}. \tag{28}$$

The transition probability P_{xy} is the probability of transition to state y given that the current state is x. The average bit-error rate (BER) P_b can be expressed as

$$P_b = \frac{P_{gb}}{P_{gb} + P_{bg}}. \tag{29}$$

It has been reported that Gilbert/Elliot model is suitable for short-term instead of long-term error correlation [39]. For long-term bursty errors, block interleaving is usually adopted to remove the long-term bursty phenomenon. Besides the interleaving technique, channel coding, such as block coding and convolutional coding, is also an important component to reduce errors induced by impaired channels. In our simulations, (7, 4) Hamming code is applied to correct possible single error for each code word. The packet-error rate of this model will increase if the packet size increases.

For $P_{bb} = 0.2$ and $P_{gb} = 0.00005$, packet size $= 500$ bytes, and the number of packets is 5000, our simulations show that the average packet-error rate is about 4.36%. Without the (7, 4) Hamming code, the packet-error rate is 21.42%. The bit error rate is 7.18e-5 while the theoretical BER from (29) is about 6.25e-5, close to the simulation results.

Many of the end-to-end congestion control algorithms suffer from the presence of wireless errors as discussed in [40], since more or less they need to use packet loss as a congestion index. We also have simulation results listed in [40] to show the devastating effects of wireless errors. Without the knowledge of the packet loss classification, such congestion control algorithms usually fail. RPL-XCP, on the other hand, doesn't use packet loss as a measure to control flow traffic; therefore it is more robust to wireless errors; although the wireless packet loss will lead to the loss of per-packet feedback.

We study the performance of RPL-XCP with three different kinds of packet lengths. They are 500, 1000, and 1500 bytes. $P_{\rm bb} = 0.2$ and $P_{gb} = 0.00005$ are used for the parameters of the Gilbert/Elliot's two-state Markov chain model which serves as our wireless channel model. In Table I, we list the fairness index of RPL-XCP flows which compete the same bottleneck and its link utilization at different packet sizes. The results show that with the scenarios of different packet sizes, the fairness index

TABLE I
PACKET ERROR RATES FOR EACH FLOW; PERFORMANCE OF FAIRNESS INDEX AND LINK UTILIZATION AT DIFFERENT PACKET SIZES

	Packet Error Rate							
Packet Size	flow1	flow2	flow3	flow4	flow5	flow6	Fairness Index	Utilization
500 bytes	3.33%	3.30%	3.44%	3.40%	3.43%	3.18%	1.0000	94.32%
1000 bytes	6.38%	6.45%	6.03%	6.47%	6.08%	6.45%	1.0000	94.26%
1500 bytes	9.00%	8.81%	9.15%	9.30%	9.96%	9.18%	0.9999	94.21%

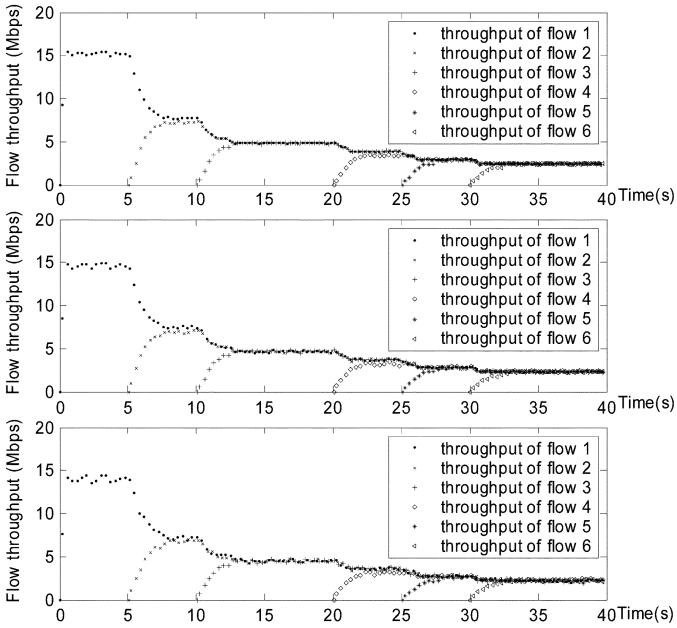


Fig. 15. Six rate-based RPL-XCP flows, starting their transfers at times 0, 5, 10, 20, 25, and 30 s. The packet sizes from top to bottom are 500, 1000, and 1500 bytes, respectively.

and link utilization are quite good, and we can conclude that the proposed RPL-XCP has good resistance to the wireless packet loss for layered video streaming.

Note that the link utilization is measured at the bottleneck (before the last wireless link) to see the effect of the loss of accumulated per-packet feedback. In Fig. 15, we present their throughputs at different packet lengths. Through a closer look, we may see some small disturbance of throughput when the wireless error is severe; however, it's not really noticeable. The convergence speed is still rather fast and the fairness is achieved.

VI. CONCLUSION

Motivated by the advantages offered by XCP, which enables packets to convey control information between end-to-end points, we generalize the window-based XCP to include rate-based flows and rate-based layered data transmission, such as layered streaming of 3-D wavelet scalable videos. To counter the abrupt changes of staircase-like bandwidth in layered video streams, we introduce RPL to reserve the assigned spare bandwidth for the scenarios with not yet enough available bandwidth for one additional layer increase/decrease.

We show that RPL-XCP exhibits important properties, such as fast convergence speed, excellent inter-session fairness, high bottleneck utilization, good robustness to various environment, small queue size, and nearly packet-loss free. The last two properties are especially essential for video streaming applications since the queuing delay is the main source of packet delay and delay jittering; moreover, packet loss can cause dramatic quality degradation. Fast convergence and nominal rate oscillation also make RPL-XCP an attractive protocol for streaming applications when stable human perspective quality is an important consideration. Good resistance to wireless channel errors of the proposed RPL-XCP algorithm further assures the stability of layered streaming of 3-D wavelet scalable videos.

REFERENCES

[1] Text of ISO/IEC 14496-2, "MPEG-4 Video FGS v.40", Mar. 2000, N3317, Proposed Draft Amendment (PDAM), Noordwijkerhout, The Netherlands, Mar. 2000.

[2] J.-R. Ohm, M. van der Schaar, and J. W. Woods, "Interframe wavelet coding-motion picture representation for universal scalability," *Image Commun., Special Issue on Digital Cinema*, vol. 19, no. 9, pp. 877–908, 2004.

[3] J. Xu, S. Li, and Y.-Q. Zhang, "A wavelet codec using 3-D ESCOT," presented at the PCM2000, Dec. 2000.

[4] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," presented at the Internet Eng. Task Force, Jan. 1996, RFC 1889.

[5] D. Wu, Y. T. Hou, W. Zhu, Y. Q. Zhang, and J. M. Peha, "Streaming video over the internet: Approaches and directions," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 282–300, Mar. 2001.

[6] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet," *Proc. IEEE INFOCOM*, pp. 1337–1345, 1999.

[7] D. Sisalem and H. Schulzrinne, "The loss-delay based adjustment algorithm: A TCP-friendly adaptation scheme," in *Proc. NOSSDAV*, Cambridge, U.K., Jul. 1998, pp. 215–226.

[8] S. Floyd, M. Handley, J. Padhye, and J. Widrner, "Equation-based congestion control for unicast applications," in *Proc. SIGCOMM*, Aug. 2000, pp. 43–54.

[9] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," presented at the Internet Eng. Task Force, Jan. 1997, RFC 2001.

[10] G. Hasegawa and M. Murata, "Survey on fairness issues in TCP congestion control mechanisms," *IEICE Trans. Commun.*, vol. E84–B, no. 6, pp. 1461–1472, Jun. 2001.

[11] S. Shalunov, "Giga TCP-TCP on gigabit ethernet," presented at the Proc. Internet2/NLANER Joint Tech. Workshop, Boulder, CO, Jul. 2002.

[12] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proc. SIGCOMM*, 2002, pp. 89–102.

[13] S. Floyd, "TCP and explicit congestion notification," *ACM Comput. Commun. Rev.*, vol. 24, no. 5, pp. 8–23, 1994.

[14] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion notification (ECN) to IP," presented at the Internet Eng. Task Force, Sep. 2001, RFC 3168.

[15] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[16] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[17] J. Widmer, C. Boutremans, and J.-Y. Le Boudec, "End-to-end congestion control for TCP-friendly flows with variable packet size," in *Proc. SIGCOMM*, 2004, vol. 34, no. 2, pp. 137–151.

[18] M. Vojnovie and J.-Y. Le Boudec, "On the long-run behavior of equation-based rate control," in *Proc. SIGCOMM*, 2002, pp. 103–106.

[19] S. Floyd, M. Handley, and J. Padhye, A Comparison of Equation-Based and AIMD Congestion Control [Online]. Available: http://www.aciri.org/tfrc/ May 2000

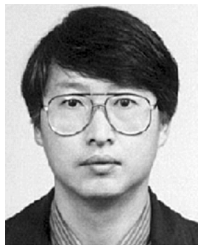
[20] D. Loguinov and H. Radha, "Increase-decrease congestion control for real-time streaming: Scalability," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 525–534.

[21] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[22] H.-F. Hsiao and J.-N. Hwang, "Layered FGS video over active network with selective drop and adaptive rate control," in *Proc. IEEE ICASSP*, 2003, pp. 752–755.
[23] B. Schwartz, A. W. Jackson, W. T. Strayer, W. Zhou, R. D. Rockwell, and C. Partridge, "Smart packets: Applying active networks to network management," *ACM Trans. Comput. Syst.*, vol. 10, no. 1, pp. 67–88, 2000.
[24] D. J. Wetherall, J. Guttag, and D. L. Tennenhouse, "ANTS: A toolkit for building and dynamically deploying network protocols," in *Proc. IEEE OPENARCH*, San Francisco, CA, Apr. 1998, pp. 117–129.
[25] A. Legout and E. W. Biersack, "PLM: Fast convergence for cumulative layered multicast transmission schemes," in *Proc. ACM SIGMETRICS'2000*, Jun. 2000, pp. 13–22.
[26] S. Keshav, The packet pair flow control protocol Int. Comp. Sci. Inst., Berkeley, CA, May 1991, Tech. Rep. 91-028.
[27] M. Jain and C. Dovrolis, "Pathload: A measurement tool for end-to-end available bandwidth," in *Proc. Passive Active Measurements*, Fort Collins, CO, Mar. 2002, pp. 14–25.
[28] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 6, pp. 879–894, Aug. 2003.
[29] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *Proc. IMC2003*, Miami Beach, FL, 2003, p. .
[30] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "PathChirp: Efficient available bandwidth estimation for network paths," in *Proc. Passive Active Measurement Workshop*, 2003, pp. 13–24.
[31] H.-F. Hsiao and J.-N. Hwang, "A max-min fairness congestion control for streaming layered video," *Proc. ICASSP*, vol. 5, pp. 981–984, May 2004.
[32] D. Katabi, "Decoupling congestion control and bandwidth allocation policy with application to high bandwidth-delay product networks," Ph.D. dissertation, MIT, Cambridge, Mar. 2003.
[33] H.-F. Hsiao, Q. Liu, and J.-N. Hwang, "Layered video over IP networks by using selective drop routers," *Proc. ISCAS*, vol. 1, pp. 441–444, May 2002.
[34] M. Karczewicz and R. Kurçeren, "The SP and SI frames design for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 637–644, Jul. 2003.
[35] B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Low bit rate scalable video coding with 3-D set partitioning in hierarchical trees (3–D SPIHT)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 8, pp. 1374–1387, Dec. 2000.
[36] S. Cen, P. C. Cosman, and G. M. Voelker, "End-to-end differentiation of congestion and wireless losses," in *Proc. ACM Multimedia Comput. Netw.*, Jan. 2002, vol. 4673, pp. 1–15.
[37] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithm for congestion avoidance in computer networks," *Comput. Netw. ISDN Syst.*, vol. 17, pp. 1–14, 1989.
[38] E. Elliot, "Estimates of error rates for codes on burst-noise channels," *Bell Syst. Tech. J.*, vol. 42, pp. 1977–1997, 1963.
[39] A. Willi, "A new class of packet- and bit level models for wireless channels," in *IEEE Symp. Personal, Indoor Mobile Radio Commun.*, Sep. 2002, vol. 5, pp. 2434–2440.
[40] H.-F. Hsiao, J. Ritcey, Y.-C. Chen, and J.-N. Hwang, "A new multimedia packet loss classification algorithm for congestion control over wired/wireless channels," *Proc. IEEE ICASSP*, pp. 1105–1108, Mar. 2005.
[41] D. Bansal and H. Balakrishnan, "Binominal congestion control algorithms," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 631–640.

Hsu-Feng Hsiao (M'05) received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, R.O.C. in 1995, the M.S. degree in electrical engineering from the National Chiao Tung University, Hsinchu, Taiwan, R.O.C. in 1997, and the Ph.D. degree in electrical engineering from the University of Washington, Seattle, in 2005.

He was an Engineering Officer in the Communication Research Laboratory, Ministry of National Defense, Taiwan, R.O.C., from 1997 to 1999. From 2000 to 2001, he was a Software Engineer at HomeMeeting, Redmond, WA. He then became a Research Assistant in the Department of Electrical Engineering, University of Washington until 2005. He is currently an Assistant Professor in the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., since August 2005. His research interests include multimedia signal processing and wired/wireless communications.



Jenq-Neng Hwang (S'82–M'84–SM'96–F'01) received the B.S. and M.S. degrees, from the National Taiwan University, Taipei, Taiwan, R.O.C., in 1981 and 1983, respectively, both in electrical engineering, and the Ph.D. degree from the University of Southern California, Los Angeles, in December 1988.

He spent 1983–1985 in obligatory military services. He was then a Research Assistant in the Signal and Image Processing Institute, Department of Electrical Engineering, University of Southern California. He was also a Visiting Student at Princeton University, Princeton, NJ, from 1987 to 1989. In the summer of 1989, he joined the Department of Electrical Engineering, University of Washington, Seattle, where he is currently a Professor. He has published more than 180 journal and conference papers, and book chapters in the areas of image/video signal processing, computational neural networks, multimedia system integration, and networking. He is the coauthor of the *Handbook of Neural Networks for Signal Processing* (CRC Press, 2001).

Dr. Hwang served as the Secretary of the Neural Systems and Applications Committee of the IEEE Circuits and Systems Society from 1989 to 1991, and was a member of Design and Implementation of the Signal Processing (SP) Systems Technical Committee of the IEEE SP Society. He is also a Founding Member of the Multimedia SP Technical Committee of IEEE SP Society. He served as the Chairman of the Neural Networks SP Technical Committee of the IEEE SP Society from 1996 to 1998, and the Society's representative to the IEEE Neural Network Council from 1997 to 2000. He served as Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING and IEEE TRANSACTIONS ON NEURAL NETWORKS, and is currently an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. He is also on the Editorial Board of the *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*. He was a Guest Editor for the IEEE TRANSACTIONS ON MULTIMEDIA, Special Issue on Multimedia over IP in March/June 2001, the Conference Program Chair for the 1994 IEEE Workshop on Neural Networks for Signal Processing held in Ermioni, Greece, in September 1994, the General Co-Chair of the International Symposium on Artificial Neural Networks held in Hsinchu, Taiwan, R.O.C., in December 1995, the Chair of the Tutorial Committee for the IEEE International Conference on Neural Networks (ICNN'96) held in Washington, DC, in June 1996, and the Program Co-Chair of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP) held in Seattle, WA, in 1998. He received the 1995 IEEE Signal Processing (SP) Society's Annual Best Paper Award (with S.-R. Lay and A. Lippman) in the area of Neural Networks for Signal Processing.