Explicit Rate Adjustment (ERA): Responsiveness, Network Utilization Efficiency and Fairness for Layered Multicast

Somnuk PUANGPRONPITAG
E-mail: somnuk.p@msu.ac.th
Faculty of Informatics, Mahasarakham University
Mahasarakham, 44150, Thailand.

Roger BOYLE
E-mail: roger@leeds.ac.uk
School of Computing, University of Leeds
Leeds, LS2 9JT, UK.

and

Surasak SANGUANPONG
E-mail: surasak.s@ku.ac.th
Department of Computer Engineer, Faculty of Engineer, Kasetsart University
Bangkok, 10900, Thailand.

ABSTRACT

To provide layered multicast with responsiveness, efficiency in network utilization, scalability and fairness (including inter-protocol fairness, intra-protocol fairness, intra-session fairness and TCP-friendliness) for layered multicast, we propose in this paper a new multicast congestion control, called Explicit Rate Adjustment (ERA). Our protocol uses an algorithm relying on TCP throughput equation and Packet-bunch Probe techniques to detect optimal bandwidth utilization; then adjusts the reception rate accordingly. We have built ERA into a network simulator (ns2) and demonstrate via simulations that the goals are reached.

Keywords: Layered Multicast Congestion Control, Packet-bunch Probe, TCP-friendliness

1. INTRODUCTION

In recent years, several studies (such as [21], [31], [6], and [13]) have focused on the design of layered multicast congestion control. In the work ([27] and [26]), we have investigated some recently proposals and evaluated their advantages and disadvantages. We have found that the previous proposals have some major drawbacks. Some designs cause over-subscription and high packet losses. Some are slow to converge and unresponsive. Some are TCP-unfriendly. Some designs are too complex or even arguable in terms of feasibility. Others are not scalable.

Hence, in this paper, we propose a new design of layered multicast congestion control, which has the following properties: scalability, responsiveness, fast convergence, fairness (including intra-session fairness, intra-protocol fairness, inter-protocol fairness and TCP friendliness), efficiency in network utilization, and simplicity to implement. Our design is based on an estimation of an explicit target rate using a Packet-bunch Probe (PP) and a TCP throughput equation. By combining this target rate estimation, the receiver-driven layered multicast approach and our new rate adjustment algorithms as well as our framework for the cooperation between the sender and the receivers, we contribute an innovative layered multicast congestion control protocol, called Explicit Rate Adjustment (ERA).

Our early draft of ERA has been published in [28]. We extend further the design in this paper. Furthermore, we undertake performance evaluation through a network simulation (ns2 [22]). The results show that ERA holds all the goal properties.

The remainder of this paper is organized as follows. We begin in Section 2 by discussion of rationale for research and related work. In Section 3, we describe our design goals. Section 4 explains the protocol basics. The framework and algorithms of ERA are proposed in Section 5. Section 6 shows the performance evaluation using network simulation technique. In Section 7, we conclude. Finally, Section 8 provides acknowledgement.

2. RATIONALE OF RESEARCH AND RELATED WORK

2.1 Related Work

So far, there have been two directions of multicast congestion control schemes proposed, namely *Single-rate Multicast Congestion Control (SR-MCC)* and *Multi-rate Multicast Congestion Control (MR-MCC)*. SR-MCC has limitations in terms of scalability to a certain number of receivers only. So, it aims at networks other than the public Internet.

The public Internet is a gigantic decentralized heterogeneous network interconnecting millions of all types of computing devices and networks throughout the world. The computing devices connected to the Internet can vary from desktop PCs, UNIX-based workstations, Personal Digital Assistants (PDA), TVs, mobile computers, automobiles, to even toasters, as well as other everyday domestic devices being connected. The networks connected to the Internet can range from dial-up modems, wireless, satellite, Digital Service Line (DSL) to high speed dedicated optical lines. In addition, the Internet keeps growing exponentially. In order to provide multicast congestion control over the huge heterogeneous network environment, several MR-MCC proposals (also known as layered multicast protocols) have been made.

Layered multicast was first proposed in [21]. It is based on the ability of a sender to generate the same data at different rates over multiple multicast streams. The sender organizes multiple multicast groups into logical layers.

TCP-friendliness (also known as TCP compatibility) is defined in [18]. It is actually the inter-protocol fairness towards TCP. Since more than 90% of today's Internet traffic is TCP-based [7], [10], the Internet community [19] has suggested that new congestion control mechanisms for traffic likely to compete with best-effort TCP traffic should be TCP-friendly.

RLM is the first proposal of receiver-driven layered multicast congestion control. It introduces the technique called *Join Experiment* to adjust the receivers' reception rate to the network condition. However, there are several fundamental problems of RLM reported in the literature such as unfairness, TCP-unfriendliness, high packet losses due to the join experiment technique and slow convergence.

RLC has been proposed to improve RLM in terms of TCP-friendliness. Yet, it was designed to be fair towards TCP whose RTT is one second only. RLC also introduces the concept of *Burst Test* to avoid over-subscription. However, even with the burst test, its join experiment is still prone to over-subscription and packet losses.

FLID-DL introduces a *Dynamic Layering (DL)* technique to mitigate the IGMP (Internet Group Management Protocol) leave latency problem. Its rate adaptation mechanism uses a probabilistic synchronized join experiment to detect congestion and relies on the distribution of layers to enforce TCP friendliness. However, our experiments in [27] have revealed that FLID-DL is not TCP-friendly, and by relying on a join experiment, it can oversubscribe and cause high packet losses

Instead of relying on a join experiment technique, PLM has been proposed using Packet-pair approach to infer the available bandwidth. However, PLM relies on FQ at routers to enforce fairness, which is infeasible. Our experiments in [27] have revealed that PLM without FQ can cause the starvation of competing TCP connections.

Wave and Equation Based Rate control (WEBRC) [14] has been designed using the TCP throughput equation and the *wave-like scheme*. However, from our experimental results in [26], it trends to cause higher PLR than our protocol. Furthermore, its algorithms are far more complicated compared with our ERA.

2.2 Rationale of Research

Since 1996, a few studies have been conducted and aimed at providing good layered multicast congestion control. However, in [27], we have examined some of the previous proposals (Receiver-driven Layered Multicast (RLM) [21], Receiver-driven Layered Congestion Control (RLC) [31], Fair Layered Increase Decrease with Dynamic Layering (FLID-DL) [6] and Packet-pair Layered Multicast (PLM) [13]), and found out several deficiencies that need to be improved. Some major deficiencies can be described as follows:

High Packet Loss Rate (PLR): For some proposals (such as RLM, RLC, and FLID-DL), the congestion detection relies on the detection of packet loss (such as by using the variant versions of *join experiment* [21]). So, they fundamentally cause high PLR.

Slow convergence and inefficiency in utilizing network: From our performance studies in [27], we have found that RLM, RLC and FLID-DL slowly converge to the optimal layer; thus leave network bandwidth under-utilized.

Slow responsiveness: We have also found that rate adaptation mechanisms of RLM, RLC and FLID-DL are too slow, and cause the protocols to respond to the network congestion slowly. This slow response results in congestion persistency and high packet loss rate.

TCP-unfriendliness: Our investigation in [27] has also revealed TCP-unfriendliness of RLM, RLC, and FLID-DL. In competing with TCP connections, it can cause a starvation of those TCP connections.

Infeasibility of implementation: Unlike RLM, RLC and FLID-DL, PLM have not suffered from the above deficiencies (High PLR, slow convergence, slow responsiveness, TCP-unfriendliness). However, the implementation scheme of PLM is infeasible due to the use of Fair Queuing (FQ) at every router to enforce the fairness. Without FQ, our experiments in [27] have revealed that PLM cannot maintain its fairness (including TCP-friendliness) property.

As a result, in this paper, we propose an innovative design of ERA to conquer these deficiencies.

3. DESIGN GOALS

We aim at designing a layered multicast congestion control protocol that has the following properties:

Responsiveness: In general, the first goal of congestion control protocols is to be responsive to congestion. A good protocol should be able to detect congestion signals quickly and reduce its transmission rate before causing congestion collapse. For layered multicast, responsiveness depends mainly on how receivers detect congestion and how quickly they react to it by dropping layers. Our layered multicast congestion control should dynamically match the bandwidth demand to the available bandwidth. Hence, it should allow users to increase the demand when additional bandwidth is available, and decrease it when the available bandwidth drops. In particular, the responsiveness to detect and fix congestion is generally the first goal of congestion control protocols. Our design goal is also detecting congestion at the incipient state and quickly reacting by unsubscribing from layers.

High network utilization: A good layered multicast protocol should be able to achieve high network utilization. When the network bandwidth becomes available, a good protocol should not leave it under-utilized. This depends mainly on how quickly receivers detect the available bandwidth and join more layers. Being responsive would also provide high network utilization.

Fast Convergence: A good layered multicast protocol should be able to allow receivers to converge rapidly from any starting state to the stable state with an optimal rate of bandwidth consumption. It is highly important to gain high network utilization. Hence, our protocol is designed to allow receivers to converge rapidly from any starting state to the stable state with an optimal rate of bandwidth consumption.

Scalability: According to [17], scalability refers to the behavior of the protocol in relation to the number of receivers and network paths, their heterogeneity, and the ability to accommodate dynamically variable sets of receivers. The IP Multicasting model provided by [9] is largely scalable, as a sender can send data to a nearly unlimited number of receivers. Therefore, good multicast congestion control mechanisms should be designed carefully to avoid severe scalability degradation. We want a protocol that is scalable to a nearly unlimited number of receivers, network paths and receivers' heterogeneity. Our protocol is therefore designed carefully to avoid techniques that may cause severe scalability degradation. In particular, we avoid any messages from receivers back to the sender or any messages among receivers that cause a scalability problem in RLM and other layered multicast protocols.

Fairness: Fairness may not be a big problem during low traffic load when demand of all competing connections can be satisfied. However, when the network becomes congested (i.e., the available bandwidth is less than the demand), it is crucial for a network to offer its resources to the competing connections as fairly as possible. With a poor bandwidth distribution scheme, even for high total network utilization, some connections may enjoy a greater share of the resources at the expense of other connections. Hence, fairness is one of the most significant goals of designing congestion control protocols. In our design, we consider the following senses of fairness:

Inter-protocol fairness (particularly TCP-friendliness): When several connections of different protocols compete for bandwidth, they should be able to share it fairly. In particular, the TCP-friendliness paradigm enforces new congestion control mechanisms should maintain fairness towards TCP.

Intra-protocol fairness: Fairness among transmission sessions of the same protocol

Intra-session fairness: Fairness among receivers of the same multicast session

Low Packet Loss Rate (PLR): In the Internet, packet loss may occur from transmission errors, or more commonly from network congestion. Since advances in networking technologies during the last 10 years have improved the network physical layer enormously, packet loss or corruption due to physical error is now only likely every 10^{-6} packets [30]. Some companies (such as Actelis Network [1]) even claim from their experiments that the Bit Error Rate (BER) of a physical network can be as small as every 10^{-9}, or even 10^{-15}. Hence, the vast majority of packet loss is caused by network congestion and overflowing queues at routers or switches.

Packet loss is a waste of bandwidth and an origin of Quality of Service (QoS) degradation. A good congestion control protocol would act before the network becomes severely congested and drop packets. So, having low packet loss rate is one of our goals

Feasibility: The mechanisms used in our design must be feasible to implement. We also try to keep the algorithms as simple as possible.

4. PROTOCOL BASICS

Best-effort Service: ERA is designed for the *Best-effort Service* networks, which have no quality of service guarantee. So, the multimedia applications supported by ERA will be limited to BE service only. We note that some multimedia applications may need QoS support. They therefore may need layered multicast designs based on the assumption that QoS will be deployed in the future Internet. Network-driven Layered Multicast (NLM) [11], and Differentiated Services Layered Multicast (DSLM) [29] are examples of such designs.

Multicast Support at the Network Layer: ERA assumes multicast support at the network layer. One of two current models of multicast delivery at network layer (the Any-Source Multicast (ASM) [9] or the Source-Specific Multicast (SSM) [3] may be used.

Single Data Source: ERA is a one-to-many congestion control protocol. All data are sent from a single source. ERA's congestion control is done per source. Multiple data sources can be supported by running multiple instances of ERA.

Layered Coding and Receiver-driven Approaches: ERA is designed by using the receiver-driven layered multicast approach (like other layered multicast protocols) to provide scalability for a very large heterogeneous group of receivers. We choose design options carefully to be compatible with the Layer Coding Transport (LCT) [17].

Error Control: To support reliable multicast application, we expect an error control used together with our congestion control. A complete protocol instantiation may include a scalable error control that is compatible with the layered encoding concept [17]. Such possible error control would be the Forward Error Correction (FEC) approach. Its standard is defined in [16]. An effective FEC algorithm (such as DF [5] or tornado [15]) may be used together with our ERA.

Explicit rate adjustment: We believe that finding a simple mechanism for explicit rate adjustment would simplify the congestion control problem. According to our algorithms, the receiver adjusts its reception rate to the target rate, which is explicitly calculated as the minimum of the estimated available bandwidth and the estimated TCP-friendly rate. The reason behind this explicit rate is that: (1) to avoid causing network congestion, we should not abuse the bandwidth by using more than an available bandwidth; (2) to be TCP-friendly, we also should not utilize more bandwidth than TCP traffic in the same condition. The details of estimating the available bandwidth and the TCP friendly rate will be described later on in this paper.

5. FRAMEWORK AND ALGORITHMS

5.1 Sender Operation

The sender has the responsibility to encode the data into multiple layers. Then, the encoded data packets of each layer are sent as a bunch to the receivers. This packet-bunch will be used in order to estimate the available bandwidth at the receiver side.

The header format of each packet is shown in Table 1. *OID* identifies which object the packet contains data for. *LID* identifies which layer the packet is a part of. *PSN* is used in order to detect packet losses. *SCT* indicates the time when the packet is sent from the sender. *FPF* indicates the first packet of the packet-bunch.

Table 1: Packet header format

Name	Description
OID	Object Identifier
LID	Layer Identifier
PSN	Packet Sequence Number
SCT	Sender Current Time
FPF	First Packet bunch Flag

For every predefined *Announcing Time* ($t_{announce}$), the sender advertises a *Session Announcement Message (SAM)* to the receivers. SAM provides a session description with the following information: data rate of each layer, number of layers, IP address of the sender, IP address and port number of each layer, packet size, object length and *Rate Adaptation Interval (RAI),* which is the predefined interval for the receivers to adapt their reception rate. This RAI can be tuned according to the application nature. Some applications may prefer a short RAI to gain more responsiveness, while some others may prefer a long RAI to gain more smoothness of reception rate.

5.2 Receiver Operation

The receiver has to receive a SAM and interpret the session description before joining a session. After that, the receiver has a role to decode, and obtain the necessary data packets to reproduce the object. Congestion control is done at the receiver side using the algorithms in the next section.

5.3 Rate Adaptation Algorithms

Rate Adaptation Algorithms of ERA can be summarized as follows:

(1) For every arrival of a packet-bunch, the receiver estimates the available bandwidth (R'_{pp}) using the technique mentioned in Section 5.5. If the subscribed rate is higher than R'_{pp}, the receiver will immediately reduce its reception rate to avoid overloading the network.

(2) For every RAI, the receiver calculates an estimated bandwidth R_{pp} as the minimum R'_{pp} during the last RAI. There may be a pathological case, when packet bunches are lost during severe congestion. Then, we may not have enough R'_{pp} to make a good estimation of available bandwidth (R_{pp}). In this case, we set R_{pp} to –1 to indicate severe congestion.

(3) The receiver also calculates PLR, RTT, and a TCP-friendly Rate using the techniques in Sections 5.6 – 5.8. Let PLR = l, RTT =t_{RTT} , and the TCP-friendly rate = R_{TCP}

(4) The receiver calculates its current subscribed rate (R_i) using Eq. (1) with respect to the number of subscribed layers (i) maintained at the receiver, and a data rate of each layer obtained from the session description.

(5) The receiver estimates the target reception rate (R_{TARGET}) as follows:

If ($l > 0$) **Then**
  If ($R_{pp} \geq 0$) **Then**
    Set $R_{TARGET} = Min(R_{TCP}, R_{PP})$
  Else If ($R_{pp} = -1$) **Then**
    Set $R_{TARGET} = R_{TCP}$
  End If
Else
  Set $R_{TARGET} = R_{PP}$
End if

(6) The receiver subscribes to or unsubscribes from layers according to the R_{TARGET} as follows:

If ($R_i > R_{TARGET}$) **Then**
  Repeat Until ($R_i \leq R_{TARGET}$)
    If $i > 0$ **Then**
      Unsubscribe from a layer
      $i = i - 1$
    Else
      EXIT the session
    End If
  Loop
Else If ($R_i < R_{TARGET}$)
  Do While ($R_{i+1} < R_{TARGET}$)
    Subscribe to a layer
    $i = i + 1$
  Loop
Else If ($R_i = R_{TARGET}$)
  Maintain the current subscription level
End If

5.4 Layering

Layered encoding was proposed in [21]. It is based on the ability of a sender to generate the same data at different rates over multiple multicast streams. The sender organizes multiple multicast groups into logical layers. There are still several open questions of t layering design as follows:

(1) **Cumulative or non-cumulative organization of layers:** Cumulative layering means each layer provides refined information to the previous layers, and the receiver must subscribe to all layers up to and including the highest layer. For non-cumulative, each layer is independent. Receivers can choose to subscribe to any layer or only one layer. The non-cumulative scheme is also called *Simulcast* as the source transmits multiple copies of the same data simultaneously at different rates. In general, cumulative layering is used due to the complexity of framing application-level data to be compatible with non-cumulative layers and performance penalty of providing non-cumulative layering. However, the recent development of fast FEC encoding for reliable multicast for reliable multicast and fine-grained rate video coding have mitigated the problems. In addition, Byers et al. [4] suggests that a careful design of non-cumulative layering and corresponding congestion control mechanisms would allow receivers to perform fine-grained congestion control (that cumulative layering cannot do). However, there is only little initial work on non-cumulative layering. Whether cumulative or non-cumulative layering would be a better choice for layered multicast protocols is still an open question.

(2) **Layer granularity:** the layer granularity refers to the rate of each layer. Some open questions related to layer granularity are: "how many layers would be used?", "how big would each layer be?"; "fine-grained or coarse-grained?" This is actually an argument of a trade-off between the number of layers, the extra complexity introduced and the bandwidth utilization achieved. A small number of layers would lead to a coarse-grained rate adaptation, while a large number of layers would lead to extra complexity in multicast group management, but fine-grained rate adaptation. Nevertheless, for multimedia applications, layer granularity may not have much choice because it depends highly on the CODEC used to encode audio/video. In particular, the perceived quality and the requirement of bandwidth are the key to layer organization. Too fine-grained adjustment may be useless if that fine granularity cannot improve the user's satisfaction.

Layering scheme: The layering scheme can be specified as equal, double, or multiplicative [6]. Some layered multicast proposes to use doubling scheme (such as RLM and RLC; some use multiplicative scheme (such as FLID-DL). It is still an open research issue what the best layering scheme would be.

For the design of ERA, we choose the layer organization to be cumulative due to the complexity and performance cost of non-cumulative layering. All receivers must subscribe to or unsubscribe from layers in a consecutive order. If L_j denotes the data rate of layer j, the cumulative rate (R_i) of a receiver, which subscribes to layer i, can be calculated as:

$$R_i = \sum_{j=0}^{i} L_j \qquad (1)$$

For the layer granularity and layering scheme, we argue that none of the layering schemes is the best in every situation. In our opinions, the layering scheme and layer granularity should be chosen according to the application's requirement. We therefore leave the layer granularity and layering scheme of ERA unspecific. In the real-life implementation, ERA can be implemented using any suitable layering scheme and layer granularity up to its applications. The analysis of layer granularity and layering scheme towards different kinds of applications is beyond the scope of this work, and is left for future work.

5.5 Available Bandwidth Estimation

To estimate the available bandwidth, we use the receiver-side Packet-pair bunches Probe of Paxson [25], which is improved from the original Packet-pair Probe of Keshav [12].

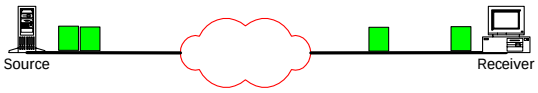

Figure 1: Packet-bunch Probe

With the PP technique (illustrated in Figure 1), the sender in our protocol periodically sends a pair of its data packets as a burst to infer the bandwidth share of the flow. For each arrived packet, the receiver checks *FPF* to determine the first packet of the packet-pairs. Then, the receiver can estimate the available bandwidth (R'_{pp}) as:

$$R'_{PP} = \frac{8M}{t_{gap}} \qquad (2)$$

where M is the packet size (in bytes), and t_{gap} is the inter-arrival time (in seconds) of packets.

5.6 Packet Loss Rate Estimation

The PLR is calculated from the number of packets lost at the receiver divided by the number of packets sent by the sender during a certain observation period. For our protocol, the number of lost packets can be detected by checking the gap in the PSN field of the packet header. The number of packets sent can be estimated as the difference between the highest and lowest PSN during the observation period.

5.7 RTT Estimation

RTT is required for the TCP throughput equation. There are several alternatives proposed to estimate RTT multicast. Some possible alternatives are described as follows:

- ***Use RTT-request packet***

A receiver sends an RTT-request packet to the sender. Then, the sender replies immediately with an RTT-reply packet. Finally, the RTT can be estimated as the time difference between sending the request and receiving the reply. This alternative works well for unicast but faces a scalability problem in multicast. In case of multicast with a large number of receivers, the RTT-request packets sent by the receivers can overload the sender and cause *network implosion*. So, some kind of suppression technique must be used to apply this technique to multicast.

- ***Estimate RTT as twice one-way delay***

The sender transmits a control message every a predefined period with a timestamp (i.e., an *SCT* field in our packet header) to the receivers. When the control message arrives, the receiver estimates half of RTT as the time difference between *SCT* and the message arrival time. However, one-way delay is not a good estimation of half RTT as revealed in [8]. In particular, this method does not work for asymmetrical paths, and requires some kind of synchronization of the clocks between receivers and senders.

- ***Estimate RTT in layered Multicast***

Luby et. al [14] have recently proposed to estimate RTT as the difference between the time of issuance of join request and the arrival time of the first packet of the layer. However, this is not exactly the RTT as the join-request messages only propagate back to the router closest to the sender only (not the sender). So, the latency between the sender and the closest router has not been counted.

We leave the efficient RTT estimation for future work, and assume that the receiver has an efficient estimated RTT. For the purpose of implementation in ns-2, we simply calculate RTT as:

$$t_{RTT} = (2 * one\ \ way\ latency) + \varepsilon \qquad (3)$$

where ε is an estimation of queuing delay. ε is arbitrary specified just for simulation purpose. Also, we use only symmetrical paths in our simulation, and assume synchronized clock between receivers and senders.

5.8 TCP-friendly Rate Estimation

There have been several analytical and empirical studies to estimate the throughput of TCP in steady state. The first model for TCP throughput has been presented in [18]. From this

model, the steady throughput (in bps) of a TCP connection (R_{TCP}) is given as:

$$R_{TCP} = \frac{8cM}{t_{RTT}\sqrt{l}} \quad (4)$$

where c is a constant (varying from 0.87 to 1.31, depending on the assumption of periodic or random loss event), M is the packet size (in bytes), t_{RTT} is the RTT (in seconds), and l is the PLR (between 0.0 and 1.0).

The model makes an assumption that TCP experiences windows reduction events only because of triple duplicate ACKs, not because of timeouts. As revealed in [23], this assumption is reasonable only for low loss rate (below 0.16). However, in a higher loss rate situation, the TCP congestion control becomes more dominated by timeout events and the model can overestimate the TCP throughput.

Hence, Padhye et. al [23] have proposed a better model for a broader range of network conditions. The model is given as:

$$R_{TCP} = \frac{8M}{t_{RTT}\sqrt{\frac{2bl}{3}} + t_{RTO}\,Min\,(1,3\sqrt{\frac{3bl}{8}})l(1+32l^2)} \quad (5)$$

where b is the number of packets acknowledged by each ACK, t_{RTO} is the TCP retransmission timeout (in seconds). This model is for TCP Reno. It is the most widely accepted TCP throughput model by the Internet research community.

To calculate the TCP-friendly rate in our algorithms, we simplify Eq. (5) as recommended in [23] by assuming: $t_{RTO} = 4 * t_{RTT}$, $b = 1$, and $Min\,(1,\ 3\sqrt{\frac{3l}{8}}) = 3\sqrt{\frac{3l}{8}}$. Then, we get:

$$R_{TCP} = \frac{8M}{T_{RTT}\sqrt{\frac{2l}{3}} + 12\sqrt{\frac{3l}{8}}l(1+32l^2)} \quad (6)$$

So, our TCP-friendly rate is relying on the Reno flavour, which is the most commonly used flavour of TCP [24]. Actually, TCP with *Selective Acknowledgement (SACK)*, proposed in [20], has improved a loss recovery scheme and its deployment is now increasing [2]. However, to the best of our knowledge, there is no well-known and widely accepted model for it. Nonetheless, any improved TCP throughput model in the future can replace Eq. (6) in our implementation.

5.9 Receiver Coordination

The coordination of the receivers under the same bottleneck link is necessary to obtain intra-session fairness. In particular, as revealed in [21], this co-ordination is significant for layered multicast protocols to utilized bandwidth efficiently and handle congestion properly.

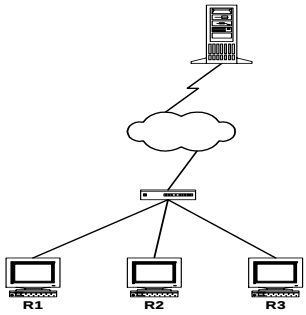

Figure 2: Receivers Coordination

As illustrated in Figure 2, if a receiver (for example, R1) unsubscribes from a layer to tackle congestion, the other receivers under the same bottleneck link (R2 and R3 in this case) should also unsubscribe from that layer. Otherwise, the multicast tree of that layer will not be pruned. The bandwidth consumption for that layer therefore does not cut out. This finally results in congestion persistency.

Furthermore, the subscription coordination is also important to an efficiency of bandwidth utilization. If a receiver (for example, R1 in Figure 2) subscribes for a layer, a multicast tree is grafted for this layer and consumes a certain amount of bandwidth. The other receivers under the same bottleneck link (R2 and R3 in this case) should also subscribe to that layer to utilize this tree as well. Otherwise, the bandwidth used for the multicast tree is not efficiently utilized.

ERA provides coordination by relying on *Session Announcement Message (SAM)* and *Rate Adaptation Intervals (RAI)* as follows. The source sends a *SAM* at every predefined time interval ($t_{announce}$) to provide information about the transmission session. RAI is a part of the information provided in SAM. In our design, we enforce that $t_{announce} = n * RAI$, where n is a positive integer. Furthermore, the receiver can join a transmission session only after it receives a *SAM*, and will adapt its reception rate every RAI. This helps coordinate the subscriptions and unsubscriptions among receivers since they adapt their rate at the same time.

6. PERFORMANCE EVALUATION

We have integrated ERA algorithms into ns-2 and run simulation experiments to evaluate ERA as follows:

6.1 Simulation Parameters

In this section, we define the default parameters used in our experiments. The packet size of all flows (ERA and TCP) is chosen to be 512 bytes. The router's queuing scheme is drop-tail, with a queue size of twice delay-bandwidth product.

Table 2: Default Parameters

Parameters	Default Values
Layering Scheme	Equal
Layer granularity	20 Kbps
Number of layers	100
Length of PP bunch	2
Min PP required	3
Rate Adaptation Interval	1 second

ERA's default parameters are summarized in Table 2. The layer organization for ERA is equal scheme. Each layer has the same size of granularity of 20 Kbps, and the number of layers is set to 100. The length of Packet-pair bunch is set to two for our simulation. In a real implementation, a bigger length could be used to gain a more accurate estimation of available bandwidth. The minimum number of PPs required to estimate available bandwidth is set to three. If there are fewer than three packet-pairs received, ERA will assume that the packet pairs are lost during severe congestion. In this case, it cannot make a good estimation of available bandwidth using PP. According to its rate adaptation algorithms, ERA uses the TCP-throughput equation to calculate the target rate instead. The full details of ERA's rate adaptation algorithms have been discussed in the previous Section.

In our experiments, the Rate Adaptation Interval (RAI) is set to one second. This means ERA will adjust its target rate every second. This value is arbitrarily set only for our simulation purpose. In a real implementation, RAI would be set according to the requirement of applications running on top ERA. Some applications may prefer a short RAI to be very responsive to the network conditions, while some applications (such as multimedia applications) may prefer a longer RAI to maintain smoothness of reception quality. It is not the purpose of this thesis to discover the best RAI value of different multicast applications; optimal RAI setting is left as future work.

For the experiments that also simulate TCP connections, we use the ns-2 implementation of TCP Reno. The maximum size of TCP congestion window is set to 2000 packets to remove the effect of the maximum window size. The applications on top of TCP sources are infinite FTP sessions that have unlimited data to send.

For some experiments, we use only one source and one receiver in multicast session. These experiments may be misunderstood by several people that it would not represent the multicast mode. However, we actually use several receivers for these experiments as well and we have the same results. This is because there is only one multicast flow in our scenario no matter what the number of receivers is. In fact, multicast is designed to use only one multicast flow instead of several unicast flows. The multicast routers should copy and forward the flow to different path themselves. So, in our scenarios, using with only one receiver or several receivers will not make different. There will be only multicast flow and the congestion control results would still be the same.

Each simulation is run 20 times using different Random Number Generator (RNG) seeds. Results are averaged and quoted with respect to confidence intervals of 95%. The error bar is shown where it is appropriate. We do not show error bars when intervals are very small or negligible.

6.2 Performance Metrics

Throughput is defined as the number of data packets (in bits) received at the receiver in a unit of time. For our experiments, the throughputs are reported in Kbps unless noted otherwise. The throughput gained by each flow indicates the rate gained and the bandwidth used by that flow. In general, congestion control is to reduce the throughput in the presence of congestion and to increase it in the absence of congestion. Also, the smoothness or oscillation of throughput with time can show the stability of rate adaptation mechanisms.

Efficiency of Network Utilization (E) is defined as the ratio of throughput gained over the maximum possible throughput. E is actually a normalized throughput, which is bounded from zero to one. If E is one, then the protocol has fully utilized the available bandwidth.

Packet Loss Ratio (PLR) is defined as the ratio of the number of packets lost over the total number of packets transmitted during the simulation. Packet loss causes a waste of bandwidth. The higher PLR, the lower E.

6.3 Experiment I: Convergence to Target Rate

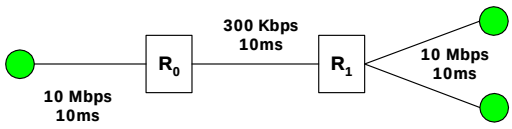

Figure 3 : Simulation Topology of Experiment I

The objective of this experiment is to test how ERA converges to the target rate, and to verify that ERA receivers can estimate the available bandwidth properly and adjust the subscription level to the optimal level quickly. We use the topology depicted in of a single multicast source with two receivers. The input bandwidth is 10 Mbps, while the bottleneck link is 300 Kbps. We start the multicast source at time zero and its sinks randomly three seconds later. The simulation is run for 80 seconds.

Figure 4 shows the results of this experiment. From the figure, we can see that ERA is very fast to converge. The convergence time is only 2 seconds for ERA to converge to subscribe 15 layers without causing packet loss. It is also highly efficient in utilizing the available bandwidth. The average throughput gained is approximately 271.93 ± 1.42 Kbps. The efficiency of network utilization (E) is 0.9 ± 0.03.

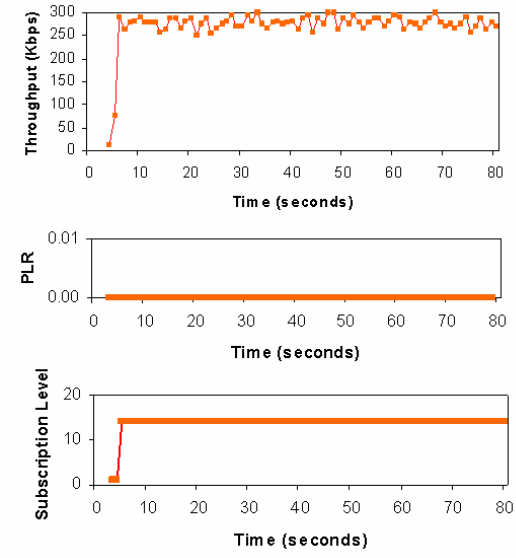

Figure 4 : Convergence to target rate

6.4 Experiment II: Response to Network Conditions

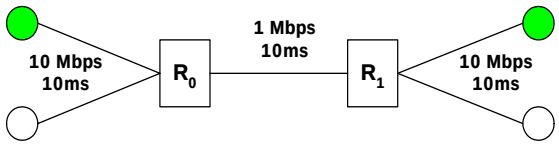

Figure 5 : Simulation Topology of Experiment II

The objective of this experiment set is to test our protocol in terms of responsiveness, packet loss ratio, efficiency of network utilization and smoothness when the available bandwidth changes during a transmission. We use the topology depicted in Figure 5 of a single multicast session sharing with a Constant Bit Rate (CBR) session. The input bandwidth is 10 Mbps, while the bottleneck link is 1 Mbps. We start the multicast source at time 0 and its sink 3 seconds later. At time 20 seconds, we start a CBR source sharing over the bottleneck link at rate 500 Kbps to use half of the bottleneck bandwidth and see how ERA adjusts to the change of available bandwidth.

From the Figure 6, when ERA starts after 3 seconds, it takes roughly 3 seconds converging to 50 layers to have the optimal rate around 1 Mbps. It also shows responsiveness to the changes of network condition when the available bandwidth is halved after 20 seconds. ERA takes only 1 second to adjust its reception rate to suit the available bandwidth. The figure also shows very low packet loss rate of the whole simulation (0-0.05 only).

The low packet loss rate and efficient bandwidth utilization result from that by using its available bandwidth estimation, ERA can sense the changes of network conditions. The rate adaptation of ERA then tries to adjust the reception rate not only to avoid over-using available bandwidth but also to efficiently utilize it.

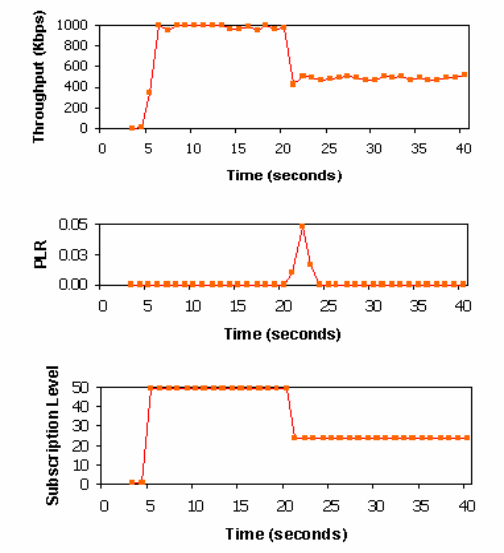

Figure 6 : Response to changes in available bandwidth

6.5 Experiment III: Bandwidth Share with TCP

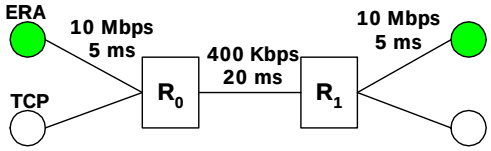

Figure 7 : Simulation Topology of Experiment III

In the previous experiment set, we have seen that ERA congestion control mechanisms can respond to congestion very well. However, just being responsive to congestion is not enough. It also needs to be responsive at the same level as TCP in order to maintain TCP friendliness. So, this further experiment set aims at investigating the behavior of ERA when sharing bandwidth with TCP.

We deploy the dumbbell topology depicted in Figure 7, using a single multicast session sharing with a TCP session. We start one session at the beginning of the simulation and another at time 20 seconds. This is to inspect two cases: (1) when TCP starts first, and (2) when ERA starts first. Each exterior link's bandwidth is 10 Mbps, while the bottleneck link is 400 Kbps. The delay of each exterior link and the bottleneck link is set to 5 and 20 milliseconds, consecutively. We start the multicast source at time zero and its sink randomly three seconds later. The simulation is run for 200 seconds.

The results show good TCP friendliness of ERA regardless of whether TCP or ERA started first. The figure shows very clearly that even when ERA starts first, it does not starve the competing TCP connection. Furthermore, it does not let the TCP connection starve itself, when TCP starts first.

From the figure, after 20 seconds when competing with TCP on the same bottleneck link, ERA and TCP can fairly share bandwidth (around 200 Kbps each). The average throughput during the last 100 seconds of ERA and TCP is approximately 196.4 ± 1.7 Kbps and 176 ± 0.7 Kbps, consecutively, with F approximately equal to 0.9 whether TCP or ERA starts first.

This is because ERA's rate adaptation algorithms have included the TCP-throughput equation to calculate the rate gained by TCP on the same network conditions as its target rate.

We note that being fair to TCP does not mean always getting exactly the same throughput with TCP. Even TCP has not been fair to itself because the throughput is inversely proportional to RTT, varying due to the queuing delay.

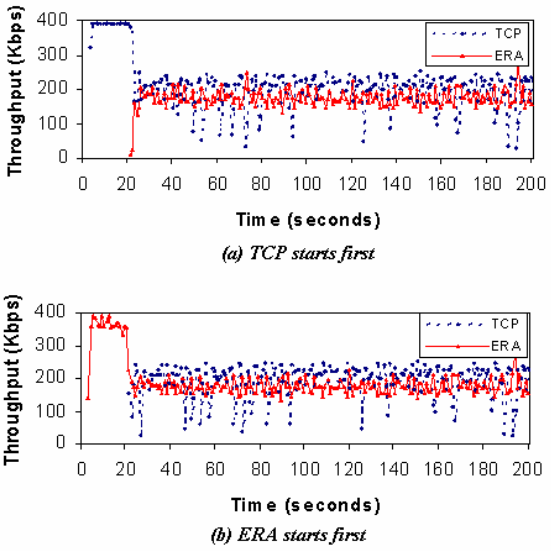

Figure 8 : Bandwidth share with TCP

6.6 Experiment IV: Co-ordination of Receivers

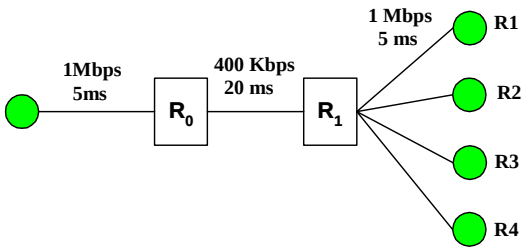

Figure 9 : Simulation Topology of Experiment IV

We use the topology depicted in Figure 9, using a single multicast source with 4 receivers. The bottleneck bandwidth is 400 Kbps, while each exterior link bandwidth is 1 Mbps. The delay of each exterior link and the bottleneck bandwidth is 5 milliseconds and 20 milliseconds, consecutively. We start receiver R1 and R2 at time 3 seconds, and receiver R3 & R4 (as late joiners) at time 20 and 40 seconds, consecutively.

Figure 10 shows a very good intra-session fairness and coordination among the downstream receivers of ERA. Furthermore, the late joining of extra receivers is not a problem for ERA. All late joiners (R3 and R4) can synchronise very well with the previous receivers under the same bottleneck link. All receivers can converge very fast and gain optimal bandwidth consumption.

When R1 and R3 start after 3 seconds, both adjust the subscription level to 20 layers and utilise bandwidth around 400 Kbps. After 20 and 40 seconds, the late joiners R3 and R4 can also adjust their subscription level and reception rate to be the same as R1 and R2. In summary, all downstream receivers, including two later joiners of the same transmission session, demonstrate a very good co-ordination and intra-session fairness to each other.

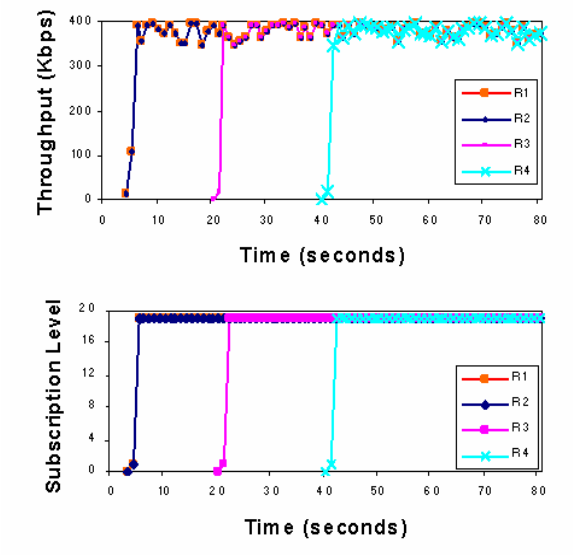

Figure 10 : Co-ordination of receivers

7. CONCLUSIONS

We have proposed a new design of layered multicast protocols using explicit rate notification based on the Packet-pair probe and TCP equation, which provides: scalability, responsiveness, fast convergence, low packet loss rate, fairness including TCP-friendliness. We have also implemented the design in ns2 and undertaken performance evaluation to show that those desirable properties are held.

8. ACKNOWLEDGEMENT

Valuable ideas and feedback on the early draft of this paper were received from Karim Djemame (Leeds University, UK), Mourad Kara (Energis, UK) and Suhaidi Hassan (Utara University, Malaysia). We are also grateful to Arnaud Legout (Castify Networks, France), Mike Luby (Digital Fountain, US) for fruitful discussion on ns-2 and Layered Multicast. This research is funded by Mahasarakham University and Thailand Research Funding.

9. REFERENCES

[1] "Actelis MetaLOOP Technology", http://www.actelis.com/solutions/technology/, Last accessed: August 2005.

[2] M. Allman, "A Web Server's View of the Transport Layer", Computer Communication Review, vol. 30, No. 5, pp. 133-142, June 2000.

[3] S. Bhattacharyya, "An Overview of Source-Specific Multicast (SSM)", IETF, RFC 3569, July 2003.

[4] J. W. Byers, M. Luby, and M. Mitzenmacher, "Fine-grained Layered Multicast", In Proceedings of IEEE INFOCOM, pp. 275-283, Anchorage, Alaska, USA, April 2001.

[5] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data", In Proceedings of ACM SIGCOMM, Vancouver, Canada,, September 1998.

[6] J. W. Byers, M. Frumin, G. Horn, M. Luby, M.. Mitzenmacher, A. Roetter, and W. Shaver, "FLID-DL: Congestion Control for Layered Multicast", In Proceedings of ACM NGC, pp. 71-82, Palo Alto, USA, November 2000.

[7] K. Claffy and G. J. Miller, "The Nature of the Beast: Recent Traffic Measurements from an Internet Backbone", In Proceedings of INET, July 1998.

[8] K. Claffy, H. W. Braun, and G. Polyzos, "Measurement considerations for assessing unidirectional latencies", Journal of Internetworking, vol. 4, No. 3, September 1993.

[9] S. Deering, "Host Extensions for IP Multicasting", IETF, RFC 1112, August 1989.

[10] G. Huston, "TCP Performance", The Internet Protocol Journal, vol. 3, No. 2, pp. 2-24, June 2000, http://www.cisco.com/ipj.

[11] K. Kang, D. Lee, H. Y. Youn, and K. Chon, "NLM: Network-based Layered Multicast for traffic control of heterogeneous network", Computer Communications March 2001.

[12] S. Keshav, "A Control-Theoretic Approach to Flow Control", In Proceedings of ACM SIGCOMM, pp. 3-15, Zurich, Switzerland, September 1991.

[13] A. Legout and E. W. Biersack, "PLM: Fast Convergence for Cumulative Layered Multicast Transmission", In Proceedings of ACM SIGMETRICS, pp. 13-22, Santa Clara, California, USA, June 2000.

[14] M. Luby, V. K. Goyal, and S. Skaria, "Wave and Equation-based Rate Control: A Massively Scalable Receiver Driven Congestion Control Protocol", Computer Communications, vol. 32, No. 4, pp. 191-214, October 2002.

[15] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Effective Erasure Correcting Codes", IEEE Transactions on Information Theory, Special Issues: Codes on Graphs and Iterative Algorithms, vol. 47, No. 2, pp. 569-584, 2001 2001.

[16] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, "Forward Error Correction Building Block", IETF, RFC 3452, December 2002.

[17] M. Luby, J. Gemmell, L. Vicisano, L. Rizzo, M. Handley, and J. Crowcroft, "Layered Coding Transport (LCT) Building Block", IETF, RFC 3451, December 2002.

[18] J. Mahdavi and S. Floyd, "TCP-friendly Unicast Rate-based Flow Control", Technical note sent to the end2end-interest mailing list, January 1997, http://www.psc.edu/networking/papers/tcpfriendly.html.

[19] A. Mankin, A. Romanow, S. Bradner, and V. Paxson, "IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols", IETF, RFC 2357, June 1998.

[20] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgement Options", IETF, RFC 2018, October 1996.

[21] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast", In Proceedings of ACM SIGCOMM, pp. 117-130, New York, USA, August 1996.

[22] ns2, "Network Simulator -- ns version 2", online software: http://www.isi.edu/nsnam/ns.

[23] J. D. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modelling TCP throughput: A Simple Model and its Empirical Validation", In Proceedings of ACM SIGCOMM, pp. 303-314, Vancouver, Canada, September 1998.

[24] V. Paxson, "Automated Packet Trace Analysis of TCP Implementation", In Proceedings of ACM SIGCOMM, pp. 167-179, September 1997.

[25] V. Paxson, "End-to-end Internet Packet Dynamics", Computer Communications, vol. 27, No. 4, pp. 139-152, October 1997.

[26] S. Puangpronpitag and R. D. Boyle, "Performance Comparison of Explicit Rate Adjustment with other Multi-rate Multicast Congestion Control Protocols", In Proceedings of the 19th UK Performance Engineering Workshop (UKPEW 03), pp. 142-153, Warwick, UK, July 2003.

[27] S. Puangpronpitag, R. D. Boyle, and K. Djemame, "Performance Evaluation of Layered Multicast Congestion Control Protocols: FLID-DL vs. PLM", In Proceedings of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 03), Montreal, Canada, July 2003.

[28] S. Puangpronpitag, R. D. Boyle, and K. Djemame, "Explicit Rate Adjustment: an Efficient Congestion Control Protocol for Layered Multicast", In Proceedings of The 11th IEEE International Conference on Networks (ICON 03), Sydney, Australia, September 2003.

[29] P. Stathopoulos, V. Zoi, D. Loukatos, L. Sarakis, and N. Mitrou, "A Network-Driven Architecture for the Multicast Delivery of Layered Video and A Comparative Study", In Proceedings of IFIP Workshop on Internet Technologies, Applications and Social Impact (WITASI), Wroclaw, Poland, October 2002.

[30] A. Tanenbaum, Computer Networks, Fourth ed: Prentice Hall, 2003

[31] L. Vicisano, L. Rizzo, and J. Crowcroft, "TCP-like Congestion Control for Layered Multicast Data Transfer", In Proceedings of IEEE INFOCOM, pp. 996-1003, San Francisco, USA, April 1998.