Scaleable Round Trip Time Estimation for Layered Multicast Protocol

Osman Ghazali and Suhaidi Hassan
Department of Computer Sciences, Faculty of Information Technology
Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia
E-mail: {osman, suhaidi}@uum.edu.my

***Abstract*—Layered multicast protocol (LMP) is designed for simultaneous and real-time content distribution to a large number of disparate receivers across a heterogeneous internet. Most LMPs use TCP-equation model to control their rate, which is usually performed at the receivers. The equation models steady-state TCP behaviour with a function of loss rate, round trip time (RTT), timeout, and packet size. Loss rate can be easily estimated at the receivers, however RTT estimation pose implosion problem at the sender in particular when the number of receivers is very large. In this paper, we proposed a new technique for scalable RTT estimation for layered multicast protocol. The technique enables layered multicast receivers to estimate RTT without causing implosion problem to the sender.**

Index Terms— *Congestion Control, Loss Rate, Layered Multicast, Transport Protocol*

I. INTRODUCTION

Layered Multicast Protocol (LMP) is designed for data distribution to large number of receivers. It is regarded as one of the solutions for data transmission of continuous multimedia applications over the best-effort Internet services. LMP allows users with different network capacities to achieve different reception rates and therefore users of different network bandwidth perceive different multimedia qualities. With this feature, layered multicast protocol can be regarded as the suitable protocol for radio and television broadcast over the Internet.

The Internet is a large and highly heterogeneous network shared by millions of hosts with different network capacities that run various applications. In this kind of environment the behaviour of the protocols particularly their responsiveness toward congestion is critical for the stability of the Internet. Since TCP is the most dominant protocol in the Internet, some researchers [1, 2] suggest that for the sake of the Internet stability and operability non-TCP protocols should be friendly towards TCP flows.

TCP-equation model is the mechanism commonly used by non-TCP protocols to control congestion. It has been adapted in many non-TCP based protocols as it enables the protocols to control congestion and at the same time to be friendly towards TCP. The most popular TCP-equation model is TCP Reno equation model that models the steady state of TCP throughput with the functions of packet size, RTT, and retransmission timeout. Of the three TCP-equation model parameters, this work focuses on RTT estimation for LMP.

RTT is one of the important parameter of TCP-equation model. In TCP, RTT is the time taken by a packet to be sent to a receiver until the reception of its acknowledgement by the sender. However, in a layered multicast session with very large number of receivers, estimating RTT is problematic. The feedback from the receivers will cause feedback implosion at the sender.

In this paper a new technique for scalable RTT estimation for layered multicast is proposed. The technique enables RTT estimation without causing feedback implosion, even when the number of receivers is very large.

The remainder of this paper is organized as follows. The next section gives an overview of TCP-equation model, Section III gives an overview of RTT estimation, Section IV describes the experimental settings, Section V presents the result, Section VI discusses the result, and Section VII concludes this paper.

II. TCP-EQUATION MODEL

A TCP-friendly layered multicast protocol employs a TCP-equation model as the mechanism to control congestion and to be friendly towards TCP data flows. The most popular model is the TCP Reno equation model proposed in [2] by Padhye et al., please refer to (1) for the outline of the model. The protocol used in this work employs the TCP-equation model to estimate TCP-compatible rates, and adjust the sending or reception rate correspond to the estimated target rate.

$$R_{TCP} = \frac{s}{RTT\sqrt{\frac{2l}{3}} + 3RTO\sqrt{\frac{3l}{8}}l(1+32l^2)} \quad (1)$$

where R_{TCP} is the throughput of TCP connection, s is the segment size (in bytes), RTT represents the round trip time, RTO is the retransmission timeout, and l denotes the loss rate (between 0.0 and 1.0).

III. ROUND TRIP TIME ESTIMATION

Currently, there is no accepted Round Trip Time (RTT) estimation standard in LMPs. RTT estimation pose a major problem in layered multicast communication due to the data packet implosion at the sender. Data implosion at the sender

1-4244-0220-4/06/$20.00©2006 IEEE

is the result of too many data packets sent to the sender by a large number of receivers.

To mitigate data implosion problem at the sender, different RTT measurement techniques have been proposed. Among the proposed techniques are fixed RTT value, one way RTT, fixed RTT value with one way queuing delay, RTT estimation with feedback suppression technique, measurement of time between a joint request until reception of the first packet, and RTT estimation with hierarchical technique.

The simplest way to assign RTT value is to use a fixed value, e.g. 1 second. This technique has been used in [3, 4]. Assigning a fixed value to RTT has the advantage that all receivers of a multicast session use the same RTT for target reception rate estimation. This enables receivers behind the same bottleneck to estimate the same rate, and consequently achieve the same layer subscription level, which is desirable for layered multicast communication. However, a protocol that employs fixed RTT value ignores the link delay which is one of the main indicators of congestion build-up in the network.

Another RTT estimation technique is to estimate one-way transmission time or double one-way transmission time [5, 6]. This technique requires the use of the source timestamp on each packet sent to the multicast channel. At the other ends, receivers will compute the time difference between the sending time (the source's time) and the receiving time (the receivers' time). The advantage of this technique is its simplicity, while the disadvantage is it may suffer from clock drift and may require clock synchronisation which is very difficult to do in multicast communication. Another disadvantage is it does not accurately measure RTT due to the asymmetric network path.

To solve clock synchronisation complexity, Mahanti et al. [7] use a combination of a fixed RTT value and queuing delay. Queuing delay is acquired by measuring the time difference in the observed OTT. Having been able to measure and use queuing delay for RTT, receivers would be able to respond to congestion build-up, and receivers behind the same bottleneck may achieve the same layer subscription level. This technique may solve time skew problem, however asymmetric network path problem remain unresolved.

Some researchers [8-11] measure full RTT by using feedback suppression techniques. These techniques use probabilistic feedback scheme to ensure that there is no more than one receiver that sends feedback to the sender at the same time, therefore avoiding too many feedbacks at the sender. However, in the case where too many receivers join a session at the same time, each receiver has to wait for a considerable amount of time before being able to measure a new RTT. The longer a receiver has to wait the lesser the RTT accuracy.

Luby et al. in [12] measure RTT as the time taken from the time of join request to a dynamic layer until the arrival of the first packet from the layer. It measures the round trip time between the sender and first router that route the multicast flows. With the technique, the receivers of have high bandwidth links will join the dynamic layer earlier than the receivers of low bandwidth link. The advantage of this technique is it enables scaleable round trip time for layered multicast communication, while the disadvantage of this technique is that currently there is no video and audio compression technique support dynamic layering.

Basu and Golestani [13] employ a hierarchical technique for RTT estimation. Using this technique, a group of receivers behind the same bottleneck (cluster) will select one receiver as a parent. The sender only needs to communicate with the parents. All receivers send feedbacks to the parents, and the parents aggregate the children feedbacks and send the aggregate feedbacks to the sender. This reduces the amount of feedbacks sent to the sender. RTT is calculated by combining parent-source RTT and child-parent RTT. The advantage of this technique is the accurate full RTT estimation without the need to do clock synchronisation, while the disadvantage is its complexity and processing overhead.

Dynamic layering and hierarchical RTT estimation techniques are very complicated. Therefore, the easiest way to avoid feedback implosion at the sender is by minimising the data packet sent to the sender or not to send feedback. The former is possible with the use of feedback suppression techniques which limits the number of feedback sent to the sender. However, in the case of too many receivers in a layered multicast session, feedback suppression techniques may not work very well. When there are too many receivers, e.g. television content distribution to hundred thousands of receivers, the time each receiver has to wait before being able to send feedback (estimate new RTT sample) would be too long. As a result, the receivers could not perform layer adaptation correctly.

Mitigating receivers' feedback can solve implosion problem at the sender. That is what being implemented in receiver-based LMPs. With no feedback form receivers to the sender, receiver-based LMPs have to resort to one way RTT or double one way RTT estimations. However with unresolved clock drift problem and asymmetry network path, one way RTT measurement could not give accurate RTT estimation

IV. INTERNET GROUP MANAGEMENT PROTOCOL (IGMP) AND MULTICAST ROUTING PROTOCOL

IGMP operates at the edge of the network between a router and its attached hosts [14], while multicast routing protocol operates at the network core between routers of the network.

The IGMP defines the behaviour of hosts and routers regarding joining and leaving a multicast group. IGMP manages multicast group and traffics with query and report messages. An end host that want to receive multicast data stream of a particular group, have to join the group. To join the multicast group the end host has to send an IGMP join message to its closest router (R1) to receive the multicast stream. Upon receiving the IGMP join message, R1 will forward the data packets of the multicast group to the end host if it is already the member of the multicast group. Otherwise, R1 will request the multicast data from its immediate router (R2). Upon receiving the multicast data packets from R2, R1 will forward them to the new member.

1-4244-0220-4/06/$20.00©2006 IEEE

Periodically, the routers will send IGMP query message to the IP multicast to ask the membership renewal from the group members. If there are members want to continue the group membership, they have to answer to the query. The router will continue forward the multicast data packets to all of its attached hosts if there is a member continues the multicast group membership. However, if there is no member continues the membership, the router will stop forwarding the multicast data packets and leave the multicast group.

When a member want to leave the multicast group, and it is the last member of the subnet to leave the multicast group, it has to send IGMP leave message to its closest router. Upon receiving the leave message, the router will stop forwarding the multicast data stream to the member.

V. SCALEABLE RTT ESTIMATION

We propose a new technique for scaleable RTT estimation for layered multicast communication. To implement the technique, we introduce a RTT layer that is used by the sender to send very small packets for RTT estimation. The technique estimates RTT as the time a receiver subscribes to the RTT layer until it receives the first packet from the RTT layer.

A. Basic Operation

Every certain interval receivers estimate a new RTT sample. To synchronize RTT estimation for all receivers, periodically the sender sends a RTT estimation announcement packet to all receivers ordering them to estimate a new RTT sample. The packet is sent through the base layer. After sending the RTT estimation announcement packet, the sender starts sending small packet to the RTT layer. The sender will stop sending packet to the RTT layer when all receivers estimate new RTT sample.

When the receivers receive the RTT estimation announcement packet, they will join the RTT layer and record the joining time. Upon receiving the packet from the RTT layer, they record the reception time of the first packet received from the RTT layer, and estimate a new RTT sample using equation (2). Then, they estimate a new RTT (average RTT) using equation (3).

$$RTT_{sample} = RT_{RTTlayer} - JT_{RTTlayer} \quad (2)$$

where $RT_{RTTlayer}$ is the reception time of the first packet from the RTT layer since the recent join request for RTT layer, and $JT_{RTTlayer}$ is the time of the recent join request for RTT layer.

$$RTT = (1-\alpha).RTT + \alpha.RTT_{sample} \quad (3)$$

where α is the weight factor. Alpha is set based on the frequency of RTT estimation (RTT estimation interval).

After estimating a new RTT sample, the receivers leave the RTT layer. Upon receiving leave requests from the receivers, the network updates the multicast tree.

B. Scalable Communication for RTT Estimation

The techniques achieve scalability by exploiting network multicast features, i.e. IGMP and multicast routing protocols. A shown in Figure 1, the sender sent a RTT estimation announcement packet (green arrow) to all receivers, and then the routers replicate and deliver the packet to all receivers. When the receivers receive the packet, they join the RTT layer by sending IGMP join requests (blue arrow) to Router 2. The join request from Receiver A is the first request received by Router 2. Upon receiving the join request from Receiver A, Router 2 immediately sends join request to Router 1. When Router 2 receives join request from Receiver B and Receiver C, it will not send another join request to Router 1 as the request already sent. This prevents the join requests imploding the upper routers and the sender. Upon receiving join request from Router 2, Router 1 will forward the RTT layer packet (black arrow) to Router 2. Upon receiving RTT layer data packets from Router 1, Router 2 replicates the packets and forwards the packet to all subscribing nodes.

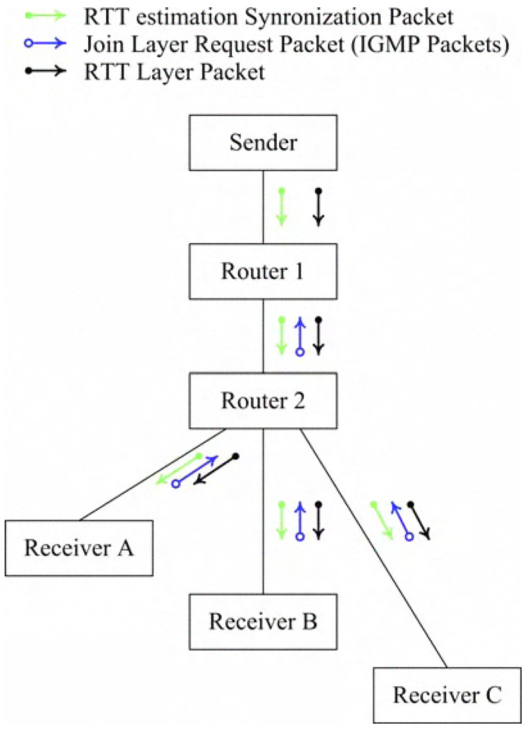

Figure 1: A Multicast Tree with Scaleable RTT Estimation

C. IGMP Assumption

The proposed technique assumes the network employs Internet Group Multicast Protocol version 3, which process multicast join and leave request immediately. Previous IGMP versions take quite some time to process leave request. For example, IGMP version 2 takes 8 seconds to process multicast leave request. To prevent the delay from affecting the accuracy of RTT estimation, the RTT estimation interval should be long enough to ensure that leave request from all receivers have been processed and update in the multicast tree of all routers.

D. Advantages

Besides scalable RTT estimation, this technique has the advantage that the receivers behind the same bottleneck or share the same path estimate the same RTT and consequently estimate the same target rate. As a result they achieve the

1-4244-0220-4/06/$20.00©2006 IEEE

same subscription level. This is a desirable property of layered multicast protocol.

VI. Analytical Analysis

For analytical analysis, we assume a simple network as in Figure 2. The transmission delay between the sender and the immediate router is assumed negligible. The transmission delay between Router 1 and Router 2 is x, the transmission delay between Router 2 and Receiver A is y, the transmission delay between Router 2 and Receiver B is $y+a$, and the transmission delay between Router 2 and Receiver C is $y+a+b$.

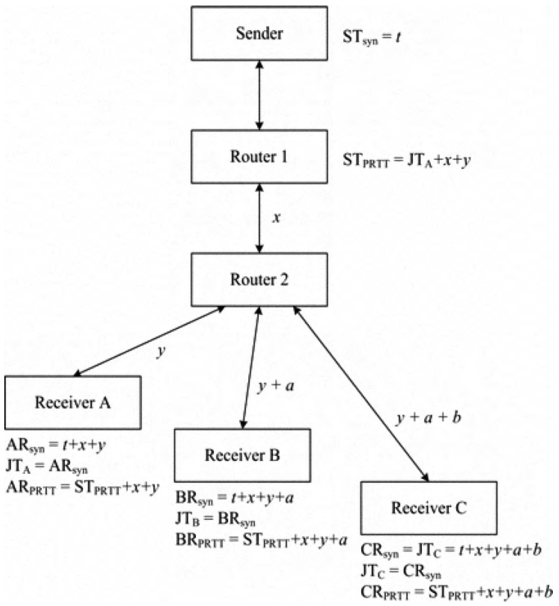

Figure 2: An Example of Scaleable RTT Estimation

Let the sender send a RTT estimation announcement packet at time t (ST_{syn}). Receiver A, B and C receive the RTT estimation announcement packet at time $t+x+y$ (AR_{syn}), $t+x+y+a$ (BR_{syn}), and $t+x+y+a+b$ (CR_{syn}) respectively. Upon receiving RTT estimation announcement packet, receiver A, B and C immediately join RTT layer. Therefore, the time of the issuance of join request to RTT layer by receiver A, B and C are the same as the reception time of RTT estimation announcement packet of receiver A, B and C, i.e. $JT_A = AR_{syn}$, $JT_B = BR_{syn}$, and $JT_C = CR_{syn}$ respectively.

As receiver A is the first node that issues join request for RTT layer, at time JT_A+x Router 2 receives the join request from Receiver A and immediately send join request to Router 1. At time JT_A+x+y Router 1 receives the join request from Router 2 and the router immediately forward RTT layer packet to Router 2 also at time JT_A+x+y (ST_{PRTT}). Router 2 then forwards the RTT layer packet to all subscribing receivers.

Receiver A, B and C receive the first packet from RTT layer at time $ST_{PRTT}+x+y$ (AR_{PRTT}), $ST_{PRTT}+x+y+a$ (BR_{PRTT}), and $ST_{PRTT}+x+y+a+b$ (CR_{PRTT}) respectively. Therefore, using equation 2 we obtain scaleable RTT (SRTT) estimation for Receiver A, B and C as below:

$SRTT_A = AR_{PRTT}-JT_A = 2x+2y$,

$SRTT_B = BR_{PRTT}-JT_B = 2x+2y$,
and $SRTT_C = CR_{PRTT}-JT_C = 2x+2y$.

All the three receivers that share the same path to the sender estimate the same RTT. This enables receivers behind the same bottleneck to estimate the same target rate and achieve the same subscription level, and consequently increase the efficiency of layered multicast protocol.

VII. Simulation Validation

We conduct simulation experiments to validate and evaluate the proposed RTT estimation technique. In particular we are interested in RTT estimation and target rate estimation of receivers behind the same bottleneck. TCP-Friendly Layered Multicast Protocol (TFLMP) [15] has been chosen for the experiment. Two TFLMPs are used in the experiments. The first is the TFLMP with the scaleable RTT estimation as the RTT estimation technique, which we name as TFLMP-1. The second is the TFLMP with round (full) RTT estimation similar to the RTT estimation technique used in [16], which we name it TFLMP-2.

A dumbbell topology as depicted in Figure 3 is used in the experiment. TFLMP sender is connected to the network through Router 1, Router 1 and Router 2 are linked, and Router 2 is connected to Receiver 1 and Receiver 2. The bandwidth of each link is set to be large enough that ensures no packet being dropped due to insufficient bandwidth. This topology represents the environment where two hosts share the same path to the sender but of different distances to the sender. The distance between the sender and Receiver 1 is 30 ms, while the distance between the sender and receiver 2 is 50 ms.

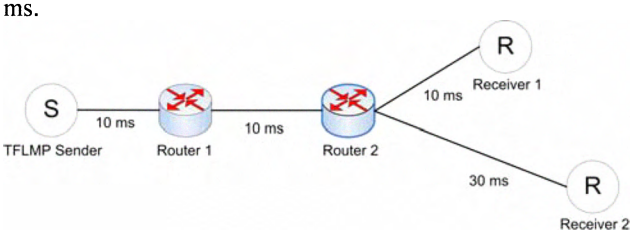

Figure 3: Simulation Topology

TABLE 1: Simulation Scenarios

	Protocol	Receiver 1 Joint Time	Receiver 2 Joint Time
Scenario 1	TFLMP-1	3 second	30 second
Scenario 2	TFLMP-1	30 second	3 second
Scenario 3	TFLMP-2	3 second	30 second
Scenario 4	TFLMP-2	30 second	3 second

Four scenarios are used in the experiments, see Table 1. All scenario use TFLMP with one sender and two receivers. Scenario 1 uses scaleable RTT estimation technique. Receiver 1 joins the multicast session first, while. Receiver 2 joins the multicast session 27 seconds later. Scenario 2 uses scaleable RTT estimation technique. Receiver 2 joins the multicast session first, while Receiver 1 joins the multicast session 27 seconds later. Scenario 3 uses two-way RTT estimation technique. Receiver 1 joins the multicast session first, while Receiver 2 joins the multicast session 27 seconds later. Scenario 4 uses scaleable RTT estimation technique. Receiver

1-4244-0220-4/06/$20.00©2006 IEEE

2 joins the multicast session first, while Receiver 1 joins the multicast session 27 seconds later.

All routers are set to be multicast capable. The Drop Tail queuing policy is used at the routers, and the buffer is set to be large enough so that no packet is dropped due to buffer overflow. Constant bit rate (CBR) is used as TFLMP data source, and we set the packet size of all flows to 1000 bytes. We use periodic error model for artificial packet drop to control the session loss rate. Loss rate is set to 5% for both connections.

VIII. RESULT AND DISCUSSION

We report and discuss the simulation results of scenario 1, scenario 2 and scenario 3. The simulation results of scenario 4 is quite similar to the simulation results of Scenario 3, therefore we omit them in the discussion.

Figure 4 shows the average SRTT estimation of Receiver 1 and Receiver 2 under scenario 1. Receiver 1 joins the multicast session at 3 second and immediately estimates average SRTT of approximately 70 ms. Then, it stabilise at approximately 80 ms until the end of the simulation. At the early period of joining the multicast session, Receiver 1 estimates short average SRTT as at that time the queue at the bottleneck link is still empty. After some time, Receiver 1 achieves the maximum subscription level and the queue at the bottleneck link achieves steady queue length, then the average SRTT stabilise. Receiver 2 joins the multicast session at 30 second, and immediately estimates the same average SRTT as receiver 1. Receiver 2 does not experience the short SRTT estimation at the early period of joining the multicast session because at the time it joins the multicast layer Receiver 1 already achieve maximum subscription level and the buffer of the bottleneck link already achieved steady length.

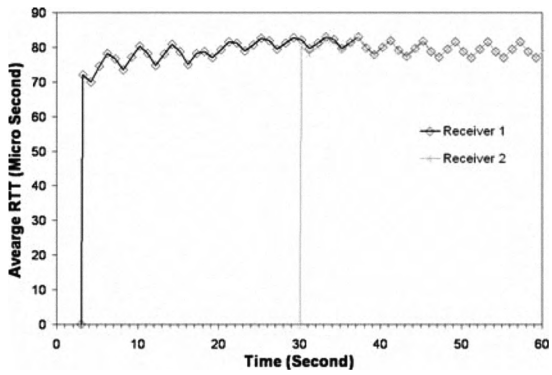

Figure 4: Scenario 1: Average SRTT Estimation

The result shows that despite the longer path to between the sender and Receiver 2 than the path between the sender and Receiver 1, Receiver 2 achieves the same SRTT as Receiver 1. This enables receivers (Receiver 1 and Receiver 2) behind the same bottleneck estimate the same RTT, and consequently achieves the same subscription level. This results in maximisation of bandwidth utility.

Figure 5 shows the estimated target rate of Receiver 1 and Receiver 2 under scenario 1. Upon joining the multicast session, Receiver 1 estimates target rate at about 420 Kbps. Then, it stabilise at approximately 380 Kbps. It estimates higher target rate for the early few seconds due to the lower the lower estimated SRTT as shown in Figure 4. On the other hand, upon joining the session, Receiver 2 immediately estimates the same target rate as Receiver 1.

Figure 6 shows the average SRTT estimation of Receiver 1 and Receiver 2 under scenario 2. Receiver 2 joins the multicast session at 3 second and immediately estimates average SRTT of approximately 100 ms. Then, it stabilise at approximately 115 ms until Receiver 1 joins the multicast session at 30 second, and immediately estimates SRTT of approximately 80 ms.

After Receiver 1 joins the multicast session, Receiver 2 estimates the same SRTT as Receiver 1, and its average SRTT begins to decrease until level with the SRTT of Receiver 1. These shows that in a multicast session with many receivers of different distance to the sender but share the same root path to the sender, SRTT of all receivers will eventually converge to the same level.

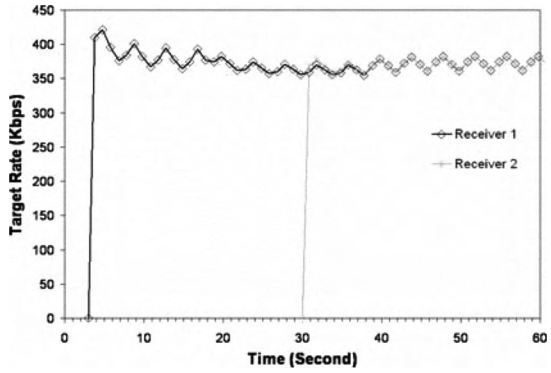

Figure 5: Scenario 1: Target Rate Estimation

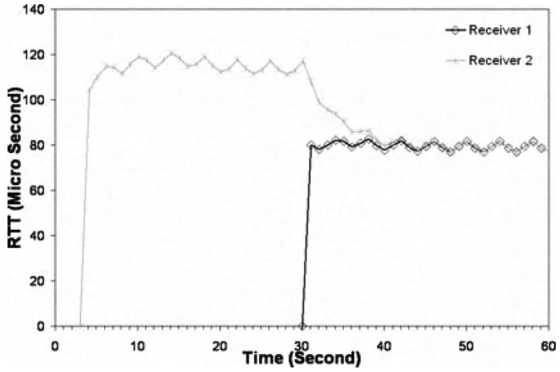

Figure 6: Scenario 2: Average SRTT Estimation

Figure 7 shows the estimated target rate of Receiver 1 and Receiver 2 under scenario 2. Upon joining the multicast session, Receiver 2 estimates target rate at about 280 Kbps. Then, it stabilise at approximately 250 Kbps until Receiver 1 joins the multicast session. Receiver 1 joins the multicast session at 30 second, and immediately estimates target rate of approximately 350 Kbps. After Receiver 1 joins the multicast session, the estimated target rate of Receiver 2 increases until level with the estimated target rate of Receiver 1.

Figure 8 shows the average RTT estimation of Receiver 1 and Receiver 2 under scenario 3. Receiver 1 joins the multicast session at 3 second and immediately estimates

1-4244-0220-4/06/$20.00©2006 IEEE

average RTT of approximately 70 ms. Then, it stabilise at approximately 80 ms until the end of the simulation. Receiver 2 joins the multicast session at 30 second, and immediately estimates average RTT at approximately 115 ms until the end of the simulation.

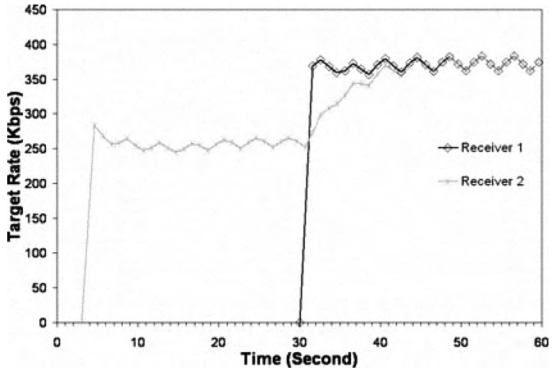

Figure 7: Scenario 2: Target Rate Estimation

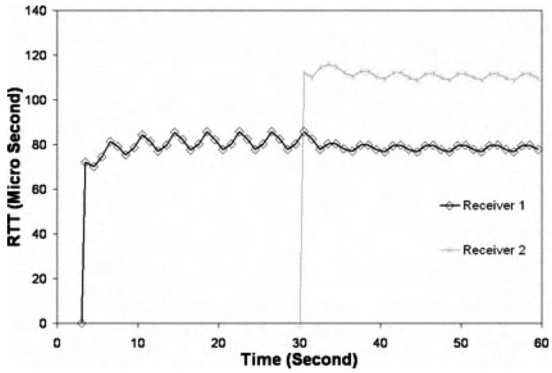

Figure 8: Scenario 3: Average RTT Estimation

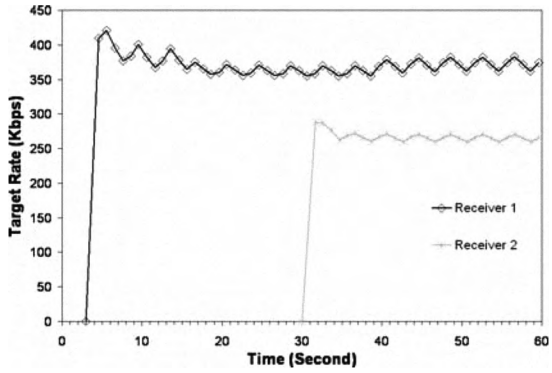

Figure 9: Scenario 3: Target Rate Estimation

Under scenario 3, Receiver 1 and Receiver 2 achieve different RTT because they do not share the same packet for RTT estimation. That is the RTT measurement is the round trip time of from end to another end. Therefore, different receivers of different distance from the sender estimate different RTT.

Figure 9 shows the target rate estimation of Receiver 1 and Receiver 2 under scenario 3. Receiver 1 joins the multicast session at 3 second and immediately estimates target rate of approximately 420 Kbps. Then, it stabilise at approximately 380 Kbps until the end of the simulation. Receiver 2 joins the multicast session at 30 second, and immediately estimates target rate of approximately 300 Kbps. Then, it stabilise at approximately 280 Kbps until the end of the simulation.

IX. CONCLUSION

We propose a new technique for scalable RTT estimation for layered multicast protocol. Analytical analysis and simulation result prove the technique is scalable and enables receivers that share the same network path but with different distance to the sender achieve the same RTT estimation. Consequently, the receivers achieve the same target rate and layer subscription level. This maximizes the utilization of the network resources.

For the future work, we will further test the techniques in more complex environment such as heterogeneous network.

REFERENCES

[1] M. Handley, S. Floyd, J. Pahdye, and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification," *IETF, RFC 3448*, 2003.

[2] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications," Proceeding of ACM SIGCOMM 2000, Stockholm, Sweden, 2000.

[3] J. Byers, M. Frumin, G. Horn, M. Luby, M. Mitzenmacher, A. Roetter, and W. Shaver, "FLID-DL: Congestion Control for Layered Multicast," Proceeding of ACM NGC 2000, Palo Alto, 2000.

[4] J. Byers, M. Luby, and M. Mitzenmacher, "Fine-Grained Layered Multicast," Proceeding of IEEE INFOCOM 2001, Anchorage, 2001.

[5] D. Sisalem and A. Wolisz, "MLDA: A TCP-friendly Congestion Control Scheme," Proceeding of IWQoS 2000, Pittsburgh, 2000.

[6] S. Puangpronpitag, R. D. Boyle, and K. Djemame, "Explicit Rate Adjustment: an Efficient Congestion Control Protocol for Layered Multicast," Proceeding of IEEE ICON 2003, Sydney, Australia, 2003.

[7] A. Mahanti, D. L. Eager, and M. K. Vernon, "Improving Multirate Congestion Control Using a TCP Vegas Throughput Model," *Computer Networks Journal*, vol. 48, pp. 113-136, 2005.

[8] I. E. Khayat and G. Leduc, "A Stable and Flexible TCP-Friendly Congestion Control Protocol for Layered Multicast Transmission," Proceeding of IDMS 2001, Lancaster, UK, 2001.

[9] J. Liu, B. Li, and Y. Zhang, "A Hybrid Adaptation Protocol for TCP-friendly Layered Multicast and Its Optimal Rate Allocation," Proceeding of IEEE INFOCOM 2002, New York, 2002.

[10] J. Liu, B. Li, and Y. Zhang, "An End-to-End Adaptation Protocol for Layered Video Multicast Using Optimal Rate Allocation," *IEEE Transactions on Multimedia*, vol. 6, 2004.

[11] G.-I. Kwon and J. W. Byers, "Smooth Multirate Multicast Congestion Control," Proceeding of IEEE INFOCOM 2003, San Francisco, 2003.

[12] M. Luby, V. K. Goyal, S. Skaria, and G. B. Horn, "Wave and Equation Based Rate Control Using Multicast Round Trip Time," Proceeding of ACM SIGCOMM 2002, Pittsburgh, 2002.

[13] A. Basu and S. J. Golestani, "Estimation of Receiver Round Trip Times in Multicast Communications," *available from* www.bell-labs.com/user/golestani/rtt.ps, 1999.

[14] W. Fenner, "Internet Group Management Protocol Version 2," *IETF, RFC 2236*, 1997.

[15] O. Ghazali and S. Hassan, "Implementation of a TCP-Friendly Layered Multicast Protocol on NS-2 Simulator," Proceeding of NS-2 Workshop of RENTAS 2004, Kuala Lumpur, 2004.

[16] J. Widmer and M. Handley, "TCP-Friendly Multicast Congestion Control (TFMCC): Protocol Specification," *Internet draft,* 2003.

1-4244-0220-4/06/$20.00©2006 IEEE