A Multicast Congestion Control Scheme for Mobile Ad-Hoc Networks

Jun Peng and Biplab Sikdar
Electrical, Computer and Systems Engineering Department
Rensselaer Polytechnic Institute (RPI), Troy, NY 12180
Email:{pengj2,sikdab}@rpi.edu

***Abstract*— This paper presents a multi-rate multicast congestion control scheme for Mobile Ad-hoc Networks (MANETs). Not only does the proposed scheme overcome the disadvantages of existing multicast congestion control protocols which prevent them from being used in MANETs, but it also achieves good performance in other aspects such as fairness with TCP, robustness against misbehaving receivers, and traffic stability. Besides achieving the above advantages, the proposed scheme does not impose any significant changes on the queuing, scheduling or forwarding policies of existing networks.**

I. Introduction

Existing multicast congestion control schemes generally fall into two categories: single-rate and multi-rate. Multi-rate schemes (e.g., [1] [2] [3] [4]) usually offer much more freedom to receivers in choosing appropriate receiving rate than single-rate schemes (e.g., [5] [6] [7]). Because the links of a multicast tree are usually heterogeneous, receivers in a multicast session may have diverse amounts of available bandwidth. So multi-rate schemes have a great advantage over singe-rate schemes in catering to every receiver in a multicast session. This paper presents a new multi-rate multicast congestion control scheme suitable for Mobile Ad-hoc Networks (MANETs).

For transport protocols not specifically designed for MANETs, the main sources of problems in MANETs are high link error rates, limited bandwidth, link access delays, and hand-offs. Almost all existing multicast congestion control schemes will suffer from the same problems as TCP suffers in MANETs (e.g., unnecessarily reducing the transmission rate in response to link errors). This is because they use losses as the indication of congestion but cannot distinguish between link-error losses and congestion losses. Another specific problem for multi-rate schemes is the link access delay in MANETs caused by access competition. Because of the inherent design of the IGMP protocol, the layer-drop latency is already a significant problem in wireline networks for multi-rate schemes [3] [4]. The link access delay in MANETs caused by competition will exacerbate the layer-drop latency problem, because pruning information can reach a upstream router only after the upstream link has been successfully accessed, and in congested situations, there is a significant delay before the upstream link becomes available. Although some schemes such as [3] [4] have made a significant progress in combating this problem, they usually introduce considerable control traffic overhead, which is a serious disadvantage in MANETs (e.g., valuable bandwidth and power are wasted.). Besides the disadvantages specific to MANETs, most existing schemes still have problems in sharing bandwidth fairly with TCP [8] [9] [3] [4] and dealing with misbehaving receivers.

To deal with the above disadvantages of existing schemes, instead of depending on individual receivers to detect congestion and adjust their receiving rates, the scheme proposed in this paper adjusts multicast traffic rate right at each bottleneck of a multicast tree. Specifically, when congestion occurs or is about to occur at a branch, some layers of the multicast sessions traversing the branch are "blocked" from entering the branch; when the branch is lightly utilized, some blocked layers are "released" to traverse the branch.

The proposed scheme overcomes most of the disadvantages of existing schemes. First, link errors cannot cause the proposed scheme to wrongly block a layer, because the queue state at a bottleneck, instead of the loss information at receivers, is used as the metric to adjust the multicast traffic rate at the bottleneck. Second, the link access delay caused by competition in MANETs cannot hinder the rate adjustment of the proposed scheme, because, instead of depending on receivers to request pruning to drop layers, the scheme blocks multicast layers right at each bottleneck of a multicast tree. Third, the proposed scheme only introduces very limited control traffic overhead because of the on-the-spot information collection and rate control. Besides the above features that enable it to work effectively and efficiently in MANETs, the proposed scheme also has good performance in fair bandwidth sharing with TCP, robustness against misbehaving receivers, and traffic stability. Moreover, the proposed scheme does not impose any significant changes on the queuing, scheduling or forwarding policies of existing networks.

The rest of the paper is organized as follows. Section II introduces the proposed scheme, and Section III analyzes the proposed scheme for fairness, effectiveness, and cost. Simulation results are presented in Section IV. The summary appears in Section V.

II. The Multi-rate Congestion Control Scheme

The proposed scheme operates in the following way. When multicast sessions traverse a link, the scheme agent starts to observe the output queue of the link and the traffic passing the link. When the number of packets in the queue, N_{QuPkt}, exceeds a threshold, $QuThresh_2$, some layers of multicast

sessions are blocked from entering the link. However, when N_{QuPkt} is below another threshold, $QuThresh_1$, for a period of time, some blocked layers are released to traverse the link. In other cases, there is usually no layer adjustment. In this way, congestion can be alleviated while free bandwidth can also be claimed. This is only a profile of the scheme. Some important details are missing. For example:

- How is it ensured that the bandwidth of a bottleneck is shared fairly between TCP sessions and multicast sessions?
- How is the layer priority information communicated if the layers of a multicast session have different priorities?

We present the scheme in detail in the rest of this section.

A. Scheme Basics

The proposed scheme retrieves some information about the competing sessions at a bottleneck to assist its operation. Specifically, the number of TCP sessions (N_{TcpSes}), the number of multicast sessions (N_{MctSes}), the number of layers of each multicast session ($N^i_{LiveLayer}, 0 < i \leq N_{MctSes}$), the average per-flow rate of TCP sessions(R_{TcpAvg}) and the average per-flow rate of multicast sessions (R_{MctAvg}) are the information retrieved. In general, all the information can be obtained by analyzing the addresses of the passing packets.

In some applications such as streaming media, a lower layer usually has higher priority than a higher layer. The proposed scheme embeds the layer priority information into the addresses used by the layers of a multicast session. Specifically, in session i, the address of the j^{th} layer is lower than the address of the k^{th} layer if j is less than k ($A_{L^i_j} < A_{L^i_k}$ if $j < k$). Meanwhile, at a bottleneck the proposed scheme distinguishes the priorities of the layers of the same multicast session according to their addresses. Specifically, a layer with a lower address has higher priority than a layer with a higher address ($P_{L^i_j} > P_{L^i_k}$ if $A^i_j < A^i_k$).

Instead of using layer-add and layer-drop at receivers as in most existing schemes, the proposed scheme uses layer-block and layer-release at bottlenecks to solve congestion and to claim bandwidth, respectively. Layer-block is the modification of the multicast routing table to stop a layer from entering a congested link; layer-release is the modification of the routing table to allow a blocked layer to traverse a link. When layer-block is necessary, the multicast session with the maximum number of layers is selected to block a layer. Within this session, the layer with the lowest priority among the unblocked layers is blocked. However, when layer-release is required, the multicast session with the minimum number of layers is selected to release a layer. Within this session, the layer with the highest priority among the blocked layers is released. In addition, receivers also play a small role in layer adjustment: each of them maintains a single empty layer. An empty layer of a receiver is a layer that is blocked somewhere in the network and has no data flowing into the receiver.

B. The Adjustment of the Number of Multicast Layers

This subsection presents the procedures for adjusting the total number of multicast layers (N_{layer}) traversing a bottleneck. The proposed scheme blocks or releases multicast layers at a bottleneck according to the state of the output queue of the link. The queue is classified into three phases: phase 1, phase 2, and phase 3. The phase of a queue is decided by the number of packets in the queue, N_{QuPkt}, and two specified thresholds, $QuThresh_1$ and $QuThresh_2$ ($QuThresh_1 < QuThresh_2$). If $N_{QuPkt} \leq QuThresh_1$, then the queue is in phase 1; if $QuThresh_1 < N_{QuPkt} \leq QuThresh_2$, then the queue is in phase 2; if $N_{QuPkt} > QuThresh_2$, the queue is in phase 3.

The layer adjustment rules are as follows. When the queue is in phase 1, it is checked if the queue has been in phase 1 for a period of time greater than $T_{Observe}$. If it has, a multicast layer is released. Otherwise, nothing is done. Phase 1 is designed to claim free bandwidth spared by TCP sessions. When the queue is in phase 2, the average per-flow rate of TCP sessions (R_{TcpAvg}) and the average per-flow rate of multicast sessions (R_{MctAvg}) are checked. When $R_{MctAvg} < R_{TcpAvg}$, a multicast layers is released. Otherwise, no action is taken. When the queue is in phase 3, R_{TcpAvg} and R_{MctAvg} are also checked. If $R_{MctAvg} > R_{TcpAvg}$, a multicast layer is blocked. Otherwise, no action is taken. The purpose of phase 3 is to detect congestion.

C. Scheme Adaptation

Generally, multicast traffic should be as stable as possible, which is necessary for some specific applications such as streaming media and also good for bandwidth utilization. To avoid various kinds of fluctuation in the number of multicast layers at a bottleneck, three procedures are added to adapt the proposed scheme to various situations.

1) When multicast sessions need to increase their traffic rate continuously, the rate of increase is decreased each time after a layer is released. Specifically, the observation time ($T_{Observe}$) for the next layer release is increased by a factor ($F_{SlowDown} > 1$): $T_{Observe} \longleftarrow T_{Observe} \times F_{SlowDown}$
2) When a layer is blocked right after a layer is released, the observation time ($T_{Observe}$) for the next layer release is increased by another factor ($F_{BackOff} > 1$): $T_{Observe} \longleftarrow T_{Observe} \times F_{BackOff}$
3) A layer is blocked in phase 3 only if the average per-flow rate of multicast sessions is greater than the average per-flow rate of TCP sessions by a ratio threshold (RT_{Block}): $(R_{MctAvg} - R_{TcpAvg})/R_{TcpAvg} > RT_{Block}$. This procedure prevents a layer from being alternatively blocked and released in phase3 and phase 2, respectively.

III. Analysis of the Scheme

This section analyzes the proposed scheme for fairness and link utilization, effectiveness in MANETs, and cost.

A. Fairness and Link Utilization

This subsection shows that the proposed scheme achieves good fairness and link utilization. For simplicity and ease of understanding, the scenario of one multicast session sharing a bottleneck with one TCP session is considered. It is assumed here that the TCP session has large enough window limits, so it can use up its share of bandwidth. The bandwidth of the bottleneck is denoted by W. The instantaneous rate and the average rate of the multicast session are denoted by $Y_1(t)$ and Y_1, respectively. Similarly, the instantaneous rate and the average rate of the TCP session are denoted by $Y_2(t)$ and Y_2, respectively. To achieve ideal fairness and link utilization, the following conditions must be met:

$$Y_1(t) + Y_2(t) = W \;\; and \;\; Y_1(t) = Y_2(t)$$

So ideally both $Y_1(t)$ and $Y_2(t)$ are constantly $W/2$.

The interaction of the multicast session with the TCP session in the scenario above can be analyzed in the following way. First, the rate of the TCP session can be represented as:

$$Y_2(t+1) = \begin{cases} Y_2(t) + a & \text{if } Y_1(t) + Y_2(t) \le W \\ b \times Y_2(t) & \text{if } Y_1(t) + Y_2(t) > W \end{cases}$$

This equation is directly drawn from the AIMD mechanism implemented in TCP, where a and b are constants representing the additive increase step and the multiplicative decrease ratio, respectively. In this equation, $Y_1(t) + Y_2(t) \le W$ and $Y_1(t)+Y_2(t) > W$ represent "not congested" and "congested", respectively. This information is inferred by the TCP sender by detecting losses in the transmission. Upon congestion, the TCP flow multiplicatively decrease its rate by a factor b; otherwise, it additively increases its rate by a step a.

Similarly, the behavior of the multicast session can also be mathematically expressed. With the assumption of large enough window limits, the TCP session has no free bandwidth to spare. As introduced in the previous section, phase 1 is designed to claim free bandwidth spared by some TCP or multicast sessions. Consequently, only phase 2 and phase 3 are meaningful to our analysis. Furthermore, since $QuThresh2$ is usually very close to the queue size, we can further simplify the analysis by distinguishing between phase 3 and phase 2 by observing if the queue is overflowing. Therefore, the behavior of the multicast session can be expressed as:

$$Y_1(t+1) = \begin{cases} Y_1(t) + g & \text{if } Y_1 < Y_2 \;\&\; Y_1(t) + Y_2(t) \le W \\ Y_1(t) - g & \text{if } Y_1 > Y_2 \;\&\; Y_1(t) + Y_2(t) > W \\ Y_1(t) & \text{Otherwise} \end{cases}$$

In the equation above, besides the congestion information, the average per-flow rates (Y_1 and Y_2) are also used. When Y_1 is less than Y_2 and the network is not congested, the multicast session increases its rate by a step g; when Y_1 is greater than Y_2 and the network is congested, the multicast session decreases its rate by a step g; otherwise, no rate adjustment is made. In real networks, the congestion information is conveyed to the scheme agent by queue overflow, while the average session rates are estimated at each bottleneck.

To achieve good fairness and efficiency, whatever the initial values $Y_1(0)$ and $Y_2(0)$ may be, $Y_1(t)$ and $Y_2(t)$ (more generally, Y_1 and Y_2) should converge to a value close to $W/2$. To test this numerically, we set W to 100, while a and b are set to 1 and 0.5 for TCP, respectively. Since a multicast session can adjust its rate only in units of layers, its rate adjustment is coarser. So g is set to a value greater than a, 5. With these settings, the traces of $Y_1(t)$ and $Y_2(t)$ with different starting points are shown in Fig. 1 (C code is used to generate the traces). Four starting points in 4 typical regions are chosen: (1) the first region: $Y_1(0) + Y_2(0) < W \;\; and \;\; Y_1(0) < Y_2(0)$, the point: (6,11); (2) the second region: $Y_1(0) + Y_2(0) < W \;\; and \;\; Y_1(0) > Y_2(0)$, the point: (21,17); (3) the third region: $Y_1(0) + Y_2(0) > W \;\; and \;\; Y_1(0) < Y_2(0)$, the point: (12,151); (4) the fourth region: $Y_1(0) + Y_2(0) > W \;\; and \;\; Y_1(0) > Y_2(0)$, the point: (86,47).

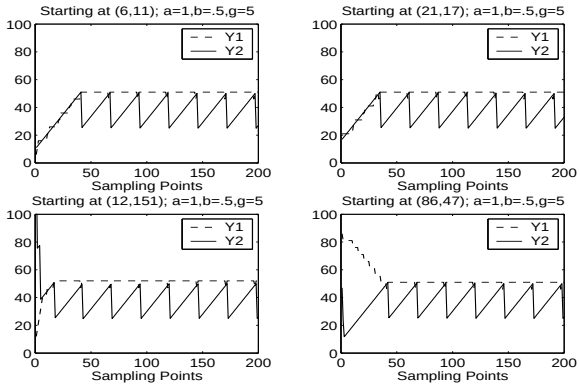

Fig. 1. Traces of $Y_1(t)$ and $Y_2(t)$ When a=1,b=.5 and g=5

From the traces in Fig. 1, both $Y_1(t)$ and $Y_2(t)$ converge and approach $W/2$, irrespective of their initial starting points. Although $Y_2(t)$ has the typical sawtooth fluctuation of a TCP session, $Y_1(t)$, which represents the multicast session rate, has a stable value close to $W/2$. The reason that the multicast session uses the bandwidth more efficiently than the TCP session is that the rate adjustment of the multicast session is assisted by the underlying network (information of Y_1 and Y_2), while the TCP session is not. This test shows that the proposed scheme achieves good fairness and link utilization. In the test above, a, b and g are set to specific values. The values of a and b only affect the behavior of the TCP session significantly. The behavior of the multicast session is mainly affected by its own step parameter g. Higher g means coarser adjustment and therefore higher fluctuation in rate. For space limitations, results corresponding to other values of a, b and g are not shown here.

B. Effectiveness in MANETs

The effectiveness of the proposed scheme in MANETs stems from several factors. Instead of waiting for receivers to request pruning and grafting as in existing schemes, the proposed scheme adjusts multicast traffic rate right at each bottleneck of a multicast tree. Therefore, it is not affected in its rate adjustment by the link access delay caused by

link competition in MANETs, which can adversely affect existing schemes significantly in their rate adjustment (i.e., further increased layer-drop latency). Link errors also cannot decrease the performance of the proposed scheme (i.e., cannot cause it to wrongly block layers), since it uses the queue state at a bottleneck instead of the loss information at receivers as the metric to adjust the multicast traffic rate at the bottleneck. In addition, the proposed scheme only has very limited control traffic overhead. In existing schemes, either poor coordination among receivers or the design of the scheme itself results in frequent branching and pruning, which may produce significant control traffic overhead [4]. Although the receivers of a multicast session need to adjust their empty layers with the proposed scheme, the adjustments are few because the proposed scheme does not have frequent layer adjustment at bottlenecks. Furthermore, all receivers under a bottleneck are well coordinated by the multicast traffic that is effectively controlled at the bottleneck. Without penalty from link errors or link access delay and without excessive control traffic overhead, the proposed scheme works effectively and efficiently in MANETs.

Another feature of the proposed scheme is that misbehaving receivers can neither benefit themselves nor hurt other receivers, since with the scheme, the number of active layers a receiver can receive is solely controlled at the bottleneck along the path from the source to the receiver. In fact, if a receiver intentionally or accidently subscribes to too many layers, the number of layers that have data flowing into the receiver will not change, because the bottleneck will block the excessive layers automatically. Other receivers under the same bottleneck are not affected either. The only consequence is that some limited bandwidth above the bottleneck is possibly wasted (see the next subsection for more details).

C. Cost

The main cost of the proposed scheme arises from retrieving information about competing sessions. All the information can be obtained by analyzing the addresses of passing packets. Since addresses have to be analyzed anyway in packet forwarding, the extra cost introduced by the proposed scheme is arguably not significant. In fact, the forwarding process only needs to put the retrieved addresses of packets into a buffer and another separate process can analyze them to obtain the information needed by the proposed scheme.

Another kind of possible cost of the proposed scheme may come from the empty layer maintained by each receiver. When a receiver maintains an empty layer, some bandwidth above the bottleneck along the path to the receiver may be wasted if no other receiver above the bottleneck needs that layer. However, the maximum amount of bandwidth that may be wasted by session m at link i is limited to the difference between the average bandwidth share for each session at link i and the bandwidth actually used by session m at link i (assuming no free bandwidth at link i).

Last, all the operations of the proposed scheme, in general, do not affect the queuing, scheduling, or forwarding policy of existing networks, so the proposed scheme will not affect existing network structure and applications if it is deployed.

IV. Simulation Results

This section presents the simulation results. The topology for the simulations is shown in Fig. 2. In the MANET, the MAC protocol is 802.11 and the ad-hoc routing protocol is DSDV. This wireline-cum-adhoc topology is chosen so that all traffic of test sessions can be easily configured to traverse a common wireless link. The behavior of the proposed scheme can then be readily observed. There are 5 test sessions: 2 multicast and 3 TCP sessions. The source of each session is in the wireline network, while the destination of each session is in the MANET. Each multicast session has 15 layers and the rate of each layer is 25Kb/s.

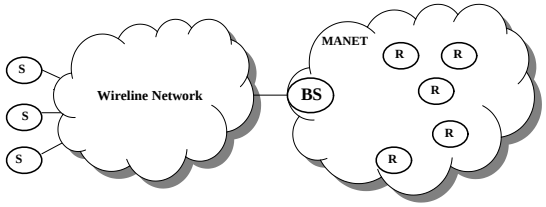

Fig. 2. The Simulation Topology

Four scenarios were considered in conducting the experiments. In the first scenario, all sessions start and stop at the same time. In the second scenario, two TCP sessions start later than the other sessions. In the third scenario, the two multicast sessions start later than the TCP sessions. In the fourth scenario, all sessions start at the same time but two sessions stop earlier than the other sessions. Simulation results show that the proposed scheme is effective in all these MANET scenarios. For space limitations, only the results of the first two scenarios are shown below.

A. Scenario 1: Simultaneous Start and Stop

In this scenario, all sessions start at the beginning of the simulation and stop at the 1500th second. The simulation results are shown in Fig. 3 and Fig. 4. Fig. 3 shows the number of layers and the throughput of each multicast session, while Fig. 4 gives the throughput of each individual TCP session and the average per-flow throughput of TCP sessions.

From these figures, each session gets a throughput close to 20 KBytes/s. Furthermore, after the initial adjustment, the number of layers of each multicast session is stable. This shows that the scheme achieves balance quickly and stays there from then on. So good fairness is achieved in this scenario and the number of layers of each multicast session is stable.

B. Scenario 2: Late Arriving TCP Sessions

This scenario tests if late arriving TCP sessions can get a fair share of bandwidth with the proposed scheme. One TCP session joins other sessions at the 500th second, while another TCP session joins them at the 1000th second. The simulation results are shown in Fig. 5 and Fig. 6.

In the first 500 seconds, there are 1 TCP session and 2 multicast sessions. Each multicast session has a throughput

close to 30 KBytes/s, while the TCP session also has a throughput close to 30 KBytes/s. In the second 500 seconds, there are 2 TCP sessions and 2 multicast sessions, and each session gets a throughput close to 25 KBytes/s. In the last 500 seconds, there are 3 TCP sessions and 2 multicast sessions. In this case each session has a throughput close to 20 KBytes/s. Therefore, late arriving TCP sessions can grab a fair share of bandwidth with the proposed scheme.

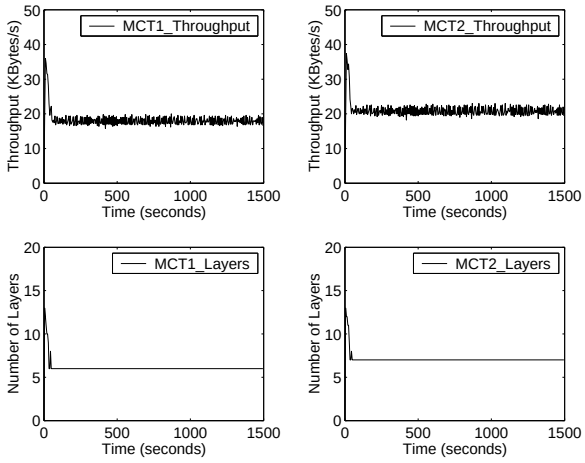

Fig. 3. Throughput and Number of Layers of Multicast Sessions

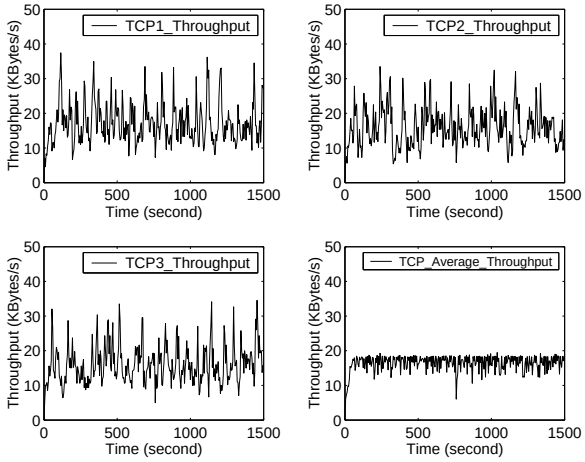

Fig. 4. Throughput and Average Per-Flow Throughput of TCP Sessions

V. Summary

This paper presents a multi-rate multicast congestion control scheme suitable for mobile ad-hoc networks. The proposed scheme overcomes the disadvantages of existing schemes which prevent them from being applied to MANET scenarios (e.g., being affected adversely by link access delays caused by access competition and by high link error rates; having excessive control traffic overhead). In addition, the proposed scheme also has good performance in many other aspects such as fairness with TCP, robustness against misbehaving receivers, and traffic stability. Moreover, the proposed scheme does not impose any significant changes on the queuing, scheduling, or forwarding policies of existing networks.

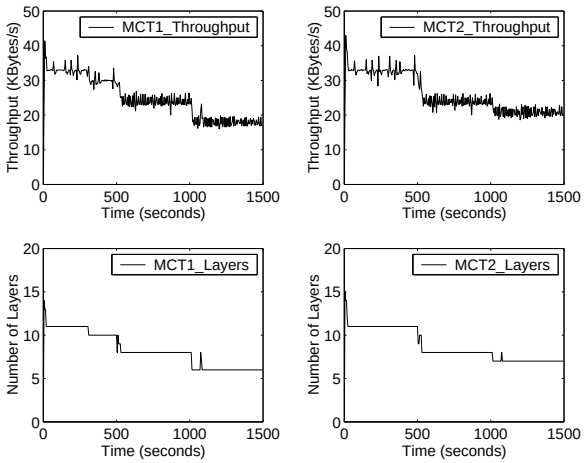

Fig. 5. Throughput and Number of Layers of Multicast Sessions

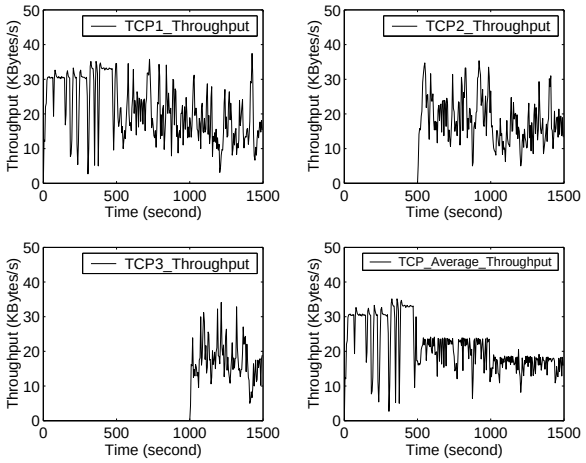

Fig. 6. Throughput and Average Per-Flow Throughput of TCP Sessions

References

[1] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proc of ACM SIGCOMM.*, Aug 1996, pp. 117–130.

[2] L. Vicisano, L. Rizzo, and J. Crowcrof, "Tcp-like congestion control for layered multicast data transfer," in *Proc of IEEE INFOCOM.*, San Franciso, March 1998, pp. 996–1003 Vol. 3.

[3] J. Byers, M. Frumin, G. Horn, M. Luby, M. Mitzenmacher, A. Roetter, and W. Shaver, "Flid-dl: Congestion control for layered multicast," in *Proceedings of NGC 2000*, November 2000, pp. 71–81.

[4] M. Luby and V. Goyal, "Wave and equation based rate control using multicast round trip time," in *Proc of ACM SIGCOMM.*, Pittsburgh, Pennsylvania, USA., Aug. 2002, pp. 191–204.

[5] I. Rhee, N. Balaguru, and G. Rouskas, "Mtcp: scalable tcp-like congestion control for reliable multicast," in *Proc of IEEE INFOCOM.*, March. 1999.

[6] L. RIZZO, "pgmcc: A tcp-friendly single-rate multicast congestion control scheme," in *Proc of ACM SIGCOMM.*, Stockholm, Sweden, Aug 2000, pp. 17–28.

[7] J. Widmer and M. Handley, "Extending equation-based congestion control to multicast applications," in *Proc of ACM SIGCOMM.*, San Diego, CA, Aug. 2001.

[8] R. Gopalakrishnan, J. Griffi oen, G. Hjalmtysson, and C. Sreenan, "Stability and fairness issues in layered multicast," in *Proceedings of the NOSSDAV*, June 1999, pp. 31–44.

[9] A. Legout and E. W. Biersack, "Pathological behaviors for rlm and rlc," in *Proceedings of the NOSSDAV*, North Carolina, USA, June 2000, pp. 164–172.