Router-Assisted Layered Multicast

ZaiChen Zhang and Victor O. K. Li
The University of Hong Kong
Hong Kong, China

***Abstract*— Several layered multicast protocols have been proposed for congestion control in real-time multicast applications. Most of them are pure end-to-end protocols, thus having difficulty in coordinating receivers and coping with traffic variations. In this paper, we propose RALM, a new receiver-driven router-assisted layered multicast protocol. RALM achieves much better performance at the expense of moderate additional complexity in the network. RALM is incrementally deployable. We evaluate RALM through simulations, and compare its performance with RLM, the well known layered multicast protocol.**

I. INTRODUCTION

Multicasting is commonly used in distributing audio and video to multiple receivers on the Internet. Multicast congestion control is much more challenging than unicast due to the following considerations. One is scalability, which is concerned with how the protocol performs when there is a large number of receivers in a multicast group; the other is heterogeneity, since receivers subscribing to the same multicast group and the paths leading to them may have different capacities. Furthermore, a multicast session should be compatible with other sessions, which means that it should be responsive to congestions in the network, and should not use significantly more or less bandwidth than similar unicast or multicast sessions.

The cumulative layered approach [1], [2] has been extensively studied for multicast congestion control. In the basic cumulative layered approach, the sender encodes the original data of a multicast session into several cumulative layers, and sends each layer through a separate multicast group. The basic layer can be independently decoded at the receivers, and higher layers, which provide performance enhancements, can only be decoded together with all of the previous layers. The proposed schemes are usually receiver-driven, that is, a receiver joins as many layers as it can handle. By using the layered approach, inter-receiver fairness (fairness between receivers in the same multicast session) [3] is improved compared with the single rate approach, where the sender sends data to receivers using a single rate that is often restricted to the slowest receiver in the group. The basic layered multicast is illustrated in Fig. 1. In this figure, there are three layers with bandwidths 128Kbps, 256Kbps, and 512Kbps, respectively. Link capacities are shown besides the links. Due to bandwidth limitations, receiver R1 only subscribes to layer 1, receivers R4 and R5 subscribe to both layers 1 and 2, and receivers R2 and R3 are able to receive all three layers.

The fundamental work on receiver-driven layered multicast is the Receiver-driven Layered Multicast (RLM) protocol proposed in [2]. It is a pure end-to-end protocol requiring no additional support beyond multicast delivery from the network. In

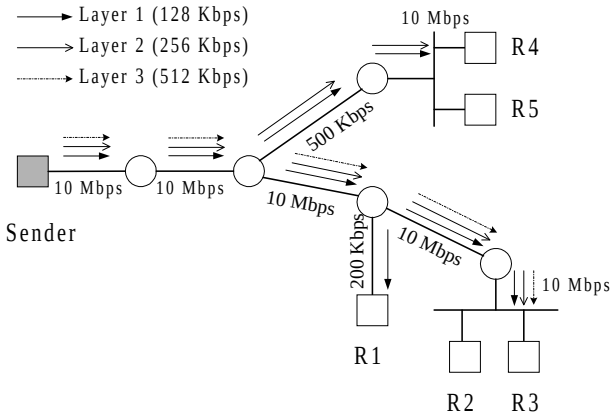

Fig. 1. Illustration of layered multicast.

RLM, the sender sends cumulative layers of a session through multiple multicast groups. A receiver starts by receiving the lowest layer. It subscribes to a higher layer if current layers are received successfully. If the newly added layer causes congestion, the receiver quickly drops it; otherwise, the receiver keeps the layer for enhanced data quality. This procedure is called the join-experiment in RLM.

RLM uses a shared learning mechanism to achieve better scalability. Before conducting a join-experiment, a receiver first multicasts an announcement of the experiment to the entire group. Other receivers will be aware of the experiment and share its result. Uniform dropping is adopted in RLM, which means that when congestion occurs, packets from different layers are dropped randomly. Uniform dropping is easy to implement, and makes the result of a join-experiment available to receivers joining only the lower layers. RLM suffers from several drawbacks. First, the join-experiment is conducted through receivers' joining or leaving multicast groups. It is on the time scale of several seconds, and can hardly catch up with rapid variations of network status. Second, failed experiments will lead to unnecessary packet loss in the network. Furthermore, coordination among receivers' joining/leaving operations, which is crucial for good performance, cannot be easily achieved in a pure end-to-end protocol like RLM.

In this paper, we propose a new multicast congestion control protocol for real-time applications — Router-Assisted Layered Multicast (RALM). This protocol is based on the receiver-driven layered approach, and uses router support to achieve enhanced performance. It requires additional state and processing burden in the network and can be incrementally deployed. We provide an overview of our protocol in Section II. Detailed protocol specifications are presented in Section III. We have implemented RALM in Network Simulator (NS) [4] version 2.1b7a.

0-7803-7400-2/02/$17.00 © 2002 IEEE

Simulation results are given in Section IV. We introduce related work in Section V. Section VI concludes this paper.

II. Overview of the Protocol

RALM is a receiver-driven layered multicast protocol with router assistance. It can be incrementally deployed. If all the routers are unaware of RALM, the protocol defaults to RLM. It outperforms RLM and the additional state and processing required in RALM-aware routers are not excessive. It is also easy to implement, and compatible with current multicast protocols. In the following three subsections, we provide an overview of RALM from the perspective of the sender, router, and receiver, respectively.

A. Sender Operation

In RALM, the sender encodes the original data of a real-time multicast session into a fixed number of layers, and sends each layer to a separate multicast group. In the current version, we assume that the bandwidth of each layer is fixed, as determined by the employed source coding algorithm. In addition, we assume that mechanisms exist for the sender to tell its receivers the bandwidth distribution. In a session, the cumulative bandwidth from layer 1 (the basic layer) to layer $k-1$ is B_l^k, which we call the Lower End Bandwidth (LEB) of layer k. The value of B_l^k should be communicated to receivers joining the group carrying layer k. For the basic layer, B_l^0 is 0.

B. Router Mechanism

A basic idea of RALM is router-initiated suspension/retry for layered multicast. A RALM-aware router monitors the buffer status of each of its outgoing links[1]. If congestion occurs at an outgoing link, the router will immediately suspend some of the current transmitting groups, i.e. stop sending packets of the groups to that outgoing link temporarily. Basically, the choice of which group to suspend is based on the importance of the data they are carrying: groups carrying higher layers (less important) will be suspended before those carrying lower layers (more important) in the same session. Fairness issues between sessions are also considered. A suspended group at an outgoing link will retry when congestion disappears. Details of the suspension algorithm will be provided in the next section.

If a group is suspended at all of its outgoing links, the router will send a leave message upstream for the group. Later, when at least one of the outgoing links decides to retry the group according to the suspension algorithm, the router will rejoin the group through sending a join message upstream.

At an outgoing link, suspended groups that are not likely to successfully transmit later will be "dropped" by the router at the link. No further retry will be conducted for a dropped group unless it is subscribed by a downstream receiver again. If a group is dropped at all of its outgoing links, the router will send a leave message upstream and delete all states related to it.

When a router suspends, retries, or drops a group at an outgoing link, it will send through subcasting[2] a suspend, retry, or drop message to all receivers in the group downstream of the link.

[1] In this paper, we use "outgoing link" and "outgoing interface" interchangeably.

[2] Subcasting refers to multicasting in the subtree of a multicast distribution tree.

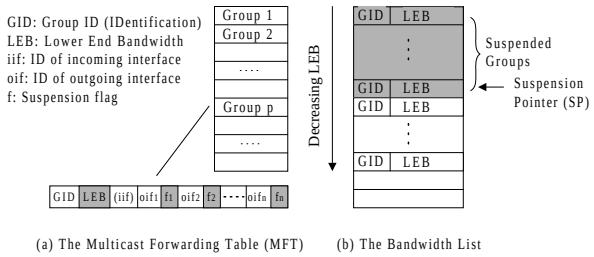

Fig. 2. The MFT and bandwidth list.

C. Receiver Operation

Receivers in RALM perform all RLM operations such as join-experiments and shared learning in a similar way. This is necessary for incremental deployment. There are several additional operations for a RALM receiver:

- Before joining a group, a receiver needs to obtain the LEB information of the group. It then puts this information in an IP option in the IP header of the join message. The IP option will be examined by a RALM-aware router and ignored by a RALM-unaware one.
- In addition to RLM operations, a RALM receiver also reacts to control messages (the suspend, retry, and drop messages) sent from RALM-aware routers. There are additional states in the state machine of the RALM receiver protocol beyond that of RLM. A "Suspended (SSP)" state indicates that there is at least one subscribed layer suspended by a RALM-aware router. In this state, a receiver will not join or leave a layer as in RLM. When losses are observed in the SSP state, the receiver changes to a "Suspension Measurement (SM)" state before dropping layers. Reordering and loss of control messages are also considered. The complete RALM receiver protocol can be found in [5].

From the above description, we can see that RALM is partially receiver-driven: receivers subscribe or drop a layer based on observed packet losses as well as indications from routers. Routers are able to make a drop decision when congestion persists (the "dropping" operation), but can only join or retry a suspended group, i.e. a group already subscribed by receivers.

III. Protocol Details

In this section, we give details of the RALM protocol.

A. The Multicast Forwarding Table (MFT)

RALM-aware routers need to maintain additional state for protocol operations. The MFT in a RALM-aware router is shown in Fig. 2(a). In this figure, iif is used to check whether a packet is from the correct incoming interface. The check is not performed in some multicast routing protocols such as those using bi-directional shared trees. A valid incoming multicast packet will be copied to each of the outgoing links listed in the outgoing interface set in its group's entry. The shadowed entries are additional state required for RALM operations. The LEB value is obtained from an IP option in the IP header of the join message. For multicast traffic not using RALM, it is set to 0. Entries in MFT are listed in decreasing value of LEB. In the oif set, there is a one-bit "suspension flag" f_i for each outgoing

interface oif$_i$. $f_i = 0$ indicates the corresponding interface is in the normal state, and $f_i = 1$ indicates the interface has been suspended. A multicast packet is only copied to an outgoing interface with its suspension flag cleared.

B. The Bandwidth List

For each outgoing interface, groups crossing it and likely to be suspended are cached in a table (called the bandwidth list) in decreasing order of LEB, as shown in Fig. 2(b). The number of entries in the bandwidth list could be very small. We use a bandwidth list with ten entries in all of our simulations.

There is a Suspension Pointer (SP) indicating the most recently suspended group. Groups at and above the SP have been suspended. The suspension algorithm (see below) always retries the group at the SP, after which the SP is moved up by one entry. When congestion occurs, the group immediately below the SP is suspended, and the SP is moved down to this group.

When a suspension of a group is cancelled, another search (the equal LEB search) is conducted to find the end of the block of groups with the same LEB, and the group is moved there. The purpose of this operation is to ensure that groups with the same LEB are suspended fairly. When there is space available at the end of the list, a search is carried out in the MFT from the group with LEB equal to or smaller than the LEB at the end of the list. The first group active at the outgoing interface is added to the list.

The bandwidth list ensures that groups containing layers of the same session will be suspended in the order of layers, i.e. higher layers will be suspended before lower layers. Furthermore, it provides a certain degree of fairness between multicast sessions at each outgoing link at any time, as stated in (1):

$$\max_p B_p^{total} - \min_p B_p^{total} \leq \max_{p,k} B_{p,k}^{layer}, \qquad (1)$$

where B_p^{total} is the total bandwidth currently consumed by session p and $B_{p,k}^{layer}$ is the bandwidth of the kth layer of session p.

C. The Suspension Algorithm

The suspension algorithm determines when to suspend a group and when to retry a suspended group. In the current version of RALM, a group will only be dropped by a router at an outgoing link when the corresponding bandwidth list is full; so the suspension algorithm does not include group dropping.

Basically, when an outgoing link is congested, an active group should be suspended, and when the congestion disappears, one of the suspended groups should be retried. We assume that there are two predefined thresholds P_h and P_l of the buffer, and $P_h > P_l$. They divide buffer usage into three states: High, Normal, and Low. We define two kinds of buffer state changes: "Low to High" and "High to Low." A "Low to High"occurs when the previous change is "High to Low"and the state changes to High again.[3] Similarly, A "High to Low" occurs when the previous change is "Low to High" and the state changes to Low again. We say congestion occurs when the buffer state changes from Low to High, and disappears when the buffer state changes from High to Low. In the former case, we suspend a group, and in the latter case, we retry a group. If the buffer state is still High (Low) after a group is suspended (retried), we suspend (retry) another group. The two thresholds scheme helps improve stability of the protocol.

[3] Initially, the state is Low and a "Low to High" occurs when the state changes to High.

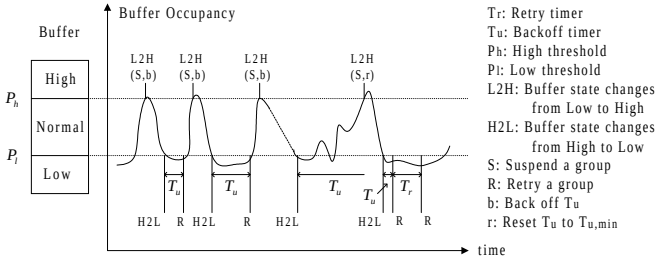

Fig. 3. Illustration of the suspension algorithm.

Three timers are maintained at each outgoing link in RALM: a suspension timer T_s, a retry timer T_r, and a back off timer T_u. T_s and T_r define the minimal time interval between two successive suspensions and two successive retries, respectively. These minimal time intervals are needed to observe the effects of suspensions or retries. T_u is set to a minimal value $T_{u,\min}$ initially. Its value is doubled each time a failed retry is detected, until a maximum value $T_{u,\max}$ is reached. A retry is called failed if it causes a changing of buffer state from Low to High. When a RALM router makes a decision of retrying a suspended group, it must wait T_u to do so. Therefore, when T_u reaches $T_{u,\max}$, the protocol enters a steady state, in which it retries a suspended group approximately every $T_{u,\max}$. The relatively large value of $T_{u,\max}$ guarantees that users of real-time applications will not be annoyed by constant changing of receiving quality in the steady state. When some events, for example, a new group joining at the outgoing interface, or arriving of a burst of traffic, break the steady state, T_u is reset to $T_{u,\min}$. This breaking of steady state is detected by observing a change of buffer state from Low to High when a T_u timer is running. Fig. 3 illustrates the basic operations of the suspension algorithm.

IV. Simulation Results

We have implemented RALM in NS. In this section, we give our initial simulation results of some simple network topologies as shown in Fig. 4. For comparison, we also simulated RLM under the same scenarios.

In all simulations, RALM parameters are: $T_s = T_r = 1.0$ second, $T_{u,\min} = 0.1$ second, and $T_{u,\max} = 30$ seconds. The bandwidth list on each outgoing interface contains 10 entries. Packet size is fixed at 1KB. Drop-tail is used for queue management. If not specified, bandwidth and delay of common links are 1Mbps and 10ms, respectively, bottleneck link bandwidth is 500Kbps, and each queue's limit is 20 packets. For queues at outgoing interfaces of RALM-aware routers, their high threshold and low threshold are set as 15 packets and 5 packets, respectively. Constant Bit Rate (CBR) sources with variable coding delays [2] are simulated in each layer. The CBR rates are $32 \times 2^{m-1}$Kbps, $m = 1, \ldots, 6$ for layer m. We run each simulation for 2000 seconds of simulation time. If there is only one session, it is started at 1 second. If there are multiple sessions,

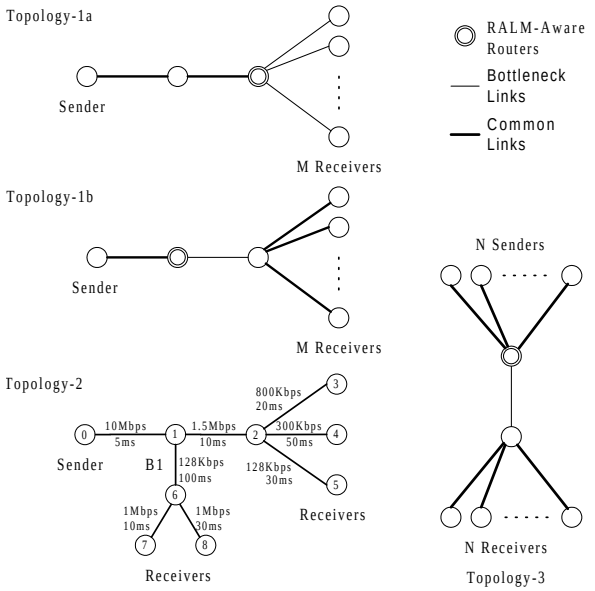

Fig. 4. Network topologies simulated.

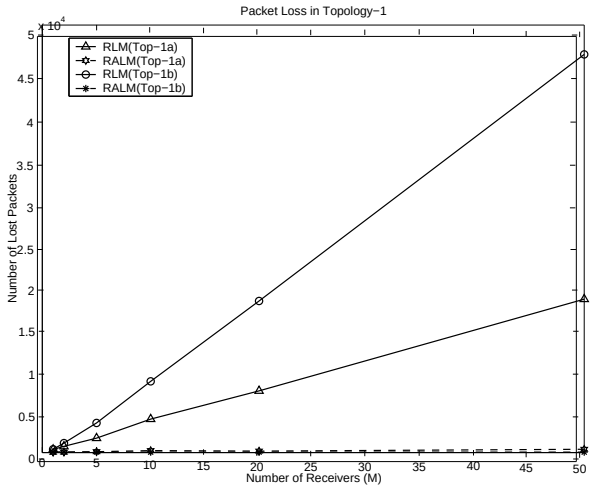

Fig. 5. Packet loss in topology-1.

they are started randomly on the interval [1, 5] seconds. Start times of receivers are randomly chosen on the interval [5, 60] seconds.

To see the effects of RALM, we strategically put RALM-aware routers just above bottleneck links from sender to receivers in all topologies. Since RALM functions are only triggered when the number of buffered packets at a queue reaches the high threshold, if a RALM-aware router is not just above a bottleneck link, its queues will not build up and the protocol will default to RLM. However, this does not mean that if we install many RALM-aware routers in a network, most of them will be redundant. Since in a real network, there are many groups taking different paths, and bottleneck links may change with traffic and status of network connections. On the other hand, if topology and traffic pattern of a network are known and do not change very often, putting RALM-aware routers at strategic locations is enough to take advantage of RALM.

In topology-1, we investigate RALM's scalability to number of receivers in a multicast session. In topology-1a, there are M receivers, within the same session, downstream of the RALM-aware router, each at a separate outgoing interface. By using this topology, we test the scalability of RALM to the number of receivers as well as the number of outgoing interfaces. In topology-1b, all the M receivers are downstream of the same interface of the RALM-aware router. This topology is used to test the scalability of RALM to the number of receivers sharing the same outgoing interface.

In topology-2, we show that RALM works very well in a network consisting of links with heterogeneous bandwidths and delays. Also, the effect of RALM is illustrated by enabling RALM at node 1, at node 2, or at both nodes.

Multiple sessions are simulated in topology-3. Scalability of RALM to the number of sessions and fairness between sessions are investigated. There are N sessions and one receiver in each session. The bottleneck link bandwidth is scaled to the number of sessions, i.e., it is set to $N \times 500$Kbps.

Fig. 5 plots the total number of lost packets versus session size in topology-1. While the total packet loss is proportional to the number of receivers in RLM, it remains constant in RALM. The majority of RALM losses are undelivered packets at RALM-aware routers. This kind of loss occurs when a packet arrives at a RALM-aware router, and finds that all outgoing interfaces of its group are suspended. The router will then free the packet and send a leave message upstream. Therefore, the number of undelivered packets is proportional to product of the bandwidth of the group and the delay of the incoming link to the router, and will not increase with the number of receivers. Since undelivered packets are freed before being put into queues at bottleneck links, they will not waste bottleneck link bandwidth. Therefore, we conclude that RALM has near optimal loss property and scales very well to the number of receivers in a session. Furthermore, in our simulations, in topology-1, throughputs of RLM and RALM are almost the same, so that the desirable loss property of RALM is not achieved by sacrificing throughput.

TABLE I

THROUGHPUTS AND PACKET LOSSES IN TOPOLOGY 2.

ID	RLM	RALM (1 enabled)	RALM (2 enabled)	RALM (1,2 enabled)
	Rcvd/Lost	Rcvd/Lost	Rcvd/Lost	Rcvd/Lost
3	116960/93	118941/100	118468/0	118707/0
4	55751/256	55753/272	55701/0	55714/0
5	24386/288	24290/265	24293/0	24297/0
7	23972/370	24173/0	23919/326	24254/0
8	23946/360	24259/0	23984/334	24340/0
Total	245015/1369	247416/637	246365/660	247312/0

Table I records the observed number of received and lost packets of each receiver in topology-2, which consists of links with heterogeneous bandwidths and delays. When we enable RALM at node 1, receivers 7 and 8, which are downstream of the bottleneck link B1, observe no loss. Similarly, when node 2 is enabled, receivers 3, 4, and 5 observe no loss. When both nodes 1 and 2 are enabled, all receivers take advantage of RALM. The number of undelivered packets at RALM-aware routers in the above three experiments are 0, 39, and 42, respectively.

Fig. 6 plots normalized throughput versus number of sessions in topology-3. We show in this figure the ranges of the throughputs among the different sessions, where range is defined as

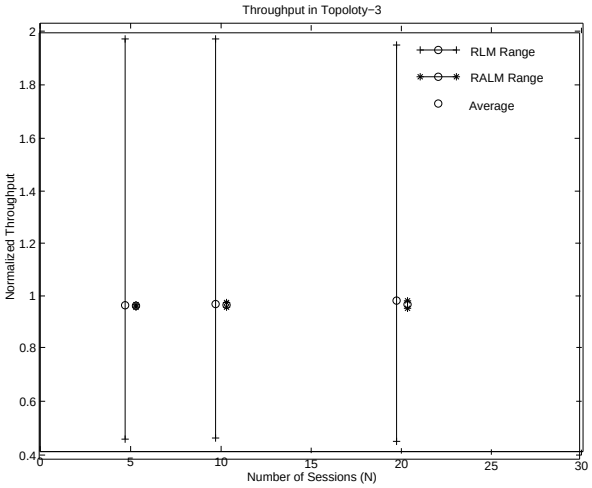

Fig. 6. Throughput in topology-3.

the difference between the highest throughput and the lowest throughput among all sessions. As expected, we see from Fig. 6 that all sessions running RALM have almost the same throughput, even though they start at different times. That is, RALM shares bottleneck link bandwidth fairly among sessions on the link.

V. Related Work

There are many layered multicast protocols proposed. Here we focus on the cumulative layered multicast protocols for real-time applications, which are closely related to our work. There are two categories of these protocols, one is pure end-to-end protocols, such as RLM, Receiver driven Layered Congestion control (RLC) [6], and ThinStreams [7]; the other is protocols with network support, such as packet Pair receiver-driven cumulative Layered Multicast (PLM) [8], priority dropping [9] and Receiver-driven Layered Multicast with Priorities (RLMP) [10]. Our proposed protocol RALM belongs to the latter category.

RLC mimics TCP's congestion control algorithm by using exponentially distributed layers and proper time delay between join attempts, thus providing fairness between multicast and TCP sessions. However, whether this kind of fairness is desirable to real-time applications is questionable. RLC adopts sender-initiated probes to avoid severe congestion introduced by failed join attempts (due to their long leaving delays) and synchronization points to coordinate receivers' join attempts [6]. ThinStreams advocates using small bandwidth layers to avoid oscillations in the network. It infers network conditions by comparing received throughput with the expected one and bases joining/leaving decisions on the difference. By adjusting joining/leaving thresholds with the number of groups joined, ThinStreams provides fairness between multicast sessions.

Better performance can be achieved with network support. The challenge is how to minimize the extra burden introduced in the network and to enable incremental deployment. PLM receivers use a receiver-driven version of packet pair [11] to infer available bandwidth in the network. The assumption is that every router in the network implements a fair scheduler. PLM converges to the optimal link utilization rapidly and enjoys inter-PLM fairness and TCP fairness. Reference [9] investigated priority dropping in the network. In this scheme, routers drop from the highest layer (which has the lowest priority) when congestion occurs. All multicast sessions will then share bandwidth fairly if their layer bandwidth distributions are the same. However, implementing priority dropping in the network is complex. RLMP tries to achieve the good performance of priority dropping while avoiding its high complexity. RLMP uses only two priority levels: the highest subscribed layer has a lower priority and other layers have higher priority. RLMP is stable even under bursty traffic and achieves fairness between competing multicast sessions (by sharing the "loss rate knowledge" among receivers).

In this paper, we proposed RALM, which also relies on network support for enhanced performance. RALM is incrementally deployable in the Internet. By simulations, we have shown that RALM has near-optimal loss property, scales well to the number of receivers and number of sessions, and achieves fairness between RALM sessions. We have also investigated RALM's complexity, fairness issues between RALM and TCP sessions, RALM's stability under bursty traffic, and RALM's parameter settings. The results can be found in [5].

VI. Conclusions

In this paper, we proposed a new layered multicast protocol — RALM. RALM relies on router support to achieve enhanced performance. We gave in this paper our initial simulation results and compared RALM with related protocols.

Acknowledgment

This research is supported in part by the Areas of Excellence Scheme established under the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-01/99).

References

[1] S. Deering, "Internet multicast routing: State of the art and open research issues," Multimedia Integrated Conferencing for Europe (MICE) seminar at the Swedish institute of computer science, Stockholm, Oct. 1993.

[2] S. McCanne and V. Jacobson, "Receiver-driven layered multicast," in Proc. ACM SIGCOMM'96, Aug. 1996, pp. 117-130.

[3] T. Jiang, M. H. Ammar, and E. W. Zegura, "Inter-receiver fairness: A novel performance measure for multicast ABR sessions," in Proc. ACM SIGMETRICS'98, June 1998, pp. 202-211.

[4] The Network Simulator — ns-2, http://www.isi.edu/nsnam/ns.

[5] Z. Zhang and V. O. K. Li, "Router-assisted layered multicast," Multimedia Networking Lab. report No. 2001-7-01, The University of Hong Kong, Hong Kong, July 2001.

[6] L. Vicisano and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," in Proc. IEEE INFOCOM'98, Feb. 1998, pp. 996-1003.

[7] L. Wu, R. Sharma, and B. Smith, "Thinstreams: An architecture for multicast layered video," in Proc. NOSSDAV'97, 1997.

[8] A. Legout and E. W. Biersack, "PLM: Fast convergence for cumulative layered multicast transmission schemes," in Proc. ACM SIGMETRICS'2000, June 2000, pp. 13-22.

[9] S. Bajaj, L. Breslau, and S. Shenker, "Uniform versus priority dropping for layered video," in Proc. ACM SIGCOMM'98, Sept. 1998, pp. 131-143.

[10] R. Gopalakrishnan, J. Griffioen, G. Hjálmtýsson, C. J. Sreenan, and S. Wen, "A simple loss differentiation approach to layered multicast," in Proc. IEEE INFOCOM'2000, pp. 461-469.

[11] S. Keshav, "Congestion control in computer networks," PhD thesis, EECS, University of Berkeley, CA 94720, USA, Sept. 1991.

[12] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," IEEE/ACM Transactions on Networking, vol. 1, no. 4, pp. 397-413, Aug. 1993.