A Model-based Scalable Reliable Multicast Transport Protocol for Satellite Networks

Prawit Chumchu , Roksana Boreli, and Aruna Seneviratne

Original scientific paper

Abstract - **In this paper, we design a new scalable reliable multicast transport protocol for satellite networks (RMT). This paper is the extensions of paper in [18]. The proposed protocol does not require inspection and/or interception of packets at intermediate nodes. The protocol would not require any modification of satellites, which could be bent-pipe satellites or onboard processing satellites. The proposed protocol is divided in 2 parts: error control part and congestion control part. In error control part, we intend to solve feedback implosion and improve scalability by using a new hybrid of ARQ (Auto Repeat Request) and adaptive forward error correction (AFEC). The AFEC algorithm adapts proactive redundancy levels following the number of receivers and average packet loss rate. This leads to a number of transmissions and the number of feedback signals are virtually independent of the number of receivers. Therefore, wireless link utilization used by the proposed protocol is virtually independent of the number of multicast receivers.**

In congestion control part, the proposed protocol employs a new window-based congestion control scheme, which is optimized for satellite networks. To be fair to the other traffics, the congestion control mimics congestion control in the well-known Transmission Control Protocol (TCP) which relies on "packet conservation" principle. To reduce feedback implosion, only a few receivers, ACKers, are selected to report the receiving status. In addition, in order to avoid "drop-to-zero" problem, we use a new simple wireless loss filter algorithm. This loss filter algorithm significantly reduces the probability of the congestion window size to be unnecessarily reduced because of common wireless losses. Furthermore, to improve achievable throughput, we employ slow start threshold adaptation based on estimated bandwidth. The congestion control also deals with variations in network conditions by dynamically electing ACKers.

Index Terms—**Reliable Multicast, Satellite networks**

I. Introduction

Multicasting greatly saves bandwidth and sender resources in applications, which need to distribute information to a selected group of end points. Internet Protocol (IP) provides mechanisms for providing multicasting services in networks which are IP based, such as the Internet. However, it only provides sufficient support for fixed hosts attached to wired terrestrial networks. Moreover, this support does not provide sufficient reliability.

A number of applications need to distribute information to many sites that are widely dispersed from each other. Satellite networks are ideally suited for providing the networking services for this class of applications as they have several attractive characteristics, such as breadth of broadcast reach, everywhere access, low-cost global coverage and flexibility capability. It is also becoming increasingly clear that these services are best provided using IP. Furthermore, there is a need to provide reliable services, as opposed simply best effort services that are offered by IP.

As a result, there has been a considerable effort at developing reliable multicast transport protocols. However, most of the effort has focused on the design of reliable multicast protocols have been for wired terrestrial networks. These designs have not taken into account the inherent characteristics of satellite channel and deployed network topologies, especially, the channel error characteristics and long propagation delays.

In the literature, several reliable multicast protocols for terrestrial networks such as in [12, 20] have been proposed. These protocols are not suitable for satellite networks because of different environments such as high delay, flat network, and high loss rate, etc. In addition, the congestion control algorithms such as in [6, 13, 21, 22] were designed for wired networks, which assume that all losses occur from congestions. This leads to congestion window or transmission rate is unnecessarily decreased when losses occur from wireless links. Reliable multicast transport protocols for satellite networks have been proposed in [5, 7, 11, 14, 17, 24] .

Gouhong et al. [7] proposed a reliable multicast protocol which enables receivers to form clusters. Receivers of a cluster share to recover losses. This protocol does not provide congestion control algorithm. Chen et al. [24] proposed a reliable multicast transport protocol for broadband satellite networks (SRMTP). This protocol is not an end-to-end scheme, which leads to a more complex implementation. Jian et al. [11] have designed a new reliable multicast transport protocol. It is based on ACK-based scheme. In order to reduce feedback implosion and low bandwidth return links, they propose the logical hierarchical groups. They assume some receivers might have terrestrial connections which could be used to create logical groups. Basu et al. [5] implemented a reliable multicast protocol for hybrid satellite link. Error control part in this protocol is based on a NAK-based protocol. With congestion control, they adopted a

Manuscript received May 31, 2005; revised June 28, 2005 and July 08, 2005. The paper was presented in part at the Conference on Software, Telecommunications and Computer Networks (SoftCOM) 2004.

Prawit Chumchu is with Department of Telecommunications Engineering, Mahanakorn University of Technology, Bangkok 10530 Thailand. (Email: prawit@mut.ac.th).

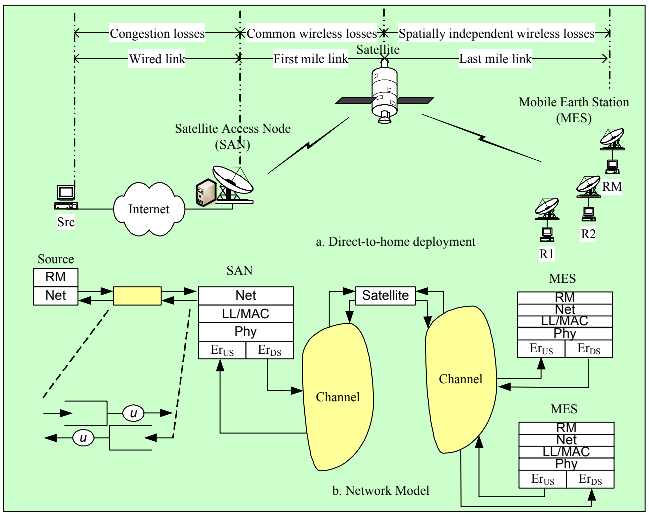
A. Seneviratne and R. Boreli are with National ICT Australia funded through the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

monitor-based flow control [22] which assumes that all losses occur from congestion. This assumption leads to face with "drop-to- zeros" problem. MFTP [17] is an end-to-end reliable multicast transport protocol. The error control of MFTP is a pass-based protocol. The pass-based protocol produces long transfer delay in the case of high loss rate and long delay channel. The congestion control is based on receiver-controlled schemes whose receivers know their capacity before joining multicast group.

The applicability of multicasting protocols has to be considered in view of a specific satellite network and applications of interest, and, as demonstrated by the references, there is no generic "one fit all" solution.

In this paper, we present a new Reliable Multicast Transmission protocol (RMT). This work has been performed as part of the Inmarsat multicasting project, and consequently the network model is based on the Inmarsat satellite system. However, the solution is generic and can be applied to any satellite systems.

The rest of the paper is structured as follows. Section II presents the studied network model. Section III presents overview of considered reliable multicast transport protocols and the RMT framework. Section IV presents the RMT error control part. Section V presents the RMT congestion control part. Section VI presents the extension of the proposed protocol for multiple satellite access nodes. The simulation results are presented in section VII and finally section VIII concludes and summarizes our work.



a. Direct-to-home deployment

b. Network Model

Fig. 1 Network Architecture

II. Network Model

Network architecture is shown in Fig. 1 (a). This model is defined as the direct-to-home deployment in [4]. Fig. 1 (b) shows the protocol stack model of the network architecture. TABLE I defines abbreviations used in the network architecture. It is composed by three network partitions: a wired link: the source and the SAN (Satellite Access Node), a first mile link: the SAN and the satellite, and last mile links: the satellite and MESs (Mobile Earth Stations). It could be analyzed that it is possible to be congested on only in the wired link. This is because in the first mile link, the satellite is designed to forward the multicast packets. The forwarding rate has to be able to forward all incoming traffic. In last mile links, it is assumed there is no congestion. This is because the sending rate from the satellite is much less than MESs' processing rate and also there is no sharing traffic from other network connections.

Packet losses could occur in all three partitions. However, packet error rate is very low for the wired link. If a packet is lost during transmission over the first mile link, all receivers will not receive the packet. This is called common wireless loss or spatially correlated wireless loss. In last mile links, it is assumed that the wireless loss is modeled as spatially uncorrelated wireless loss. This is because MESs locate in different places.

III. Overview of Considered Reliable multicast transport protocols and Rmt

Firstly we brief an overview of operation of MFTP from StarBurst [17]. In MFTP, the sender divides a file into a number of Data Transmission Units (DTUs) called a block. MFTP recommends that the block size be tied to the Maximum Transmission Units (MTU) size so that the bit map which exactly fits the number of DTUs in a block could be sent in one feedback packet. The transmission rate of a block could be set at a given level. MFTP transmits the data by first multicasting all data DTUs. Then, receivers report back what they have received in a feedback packet, using a simple bit map. The packets that have not been received, are multicasted again by the transmitter. This process continues until all receivers have successfully received all the DTUs.

Mechanism *N3* is based on NAK-based schemes with proactive FEC. The first transmission of packet in this scheme contains the original packet as well as parity packets which are the same as in RMT's adaptive forward error correction. If any group members who have not successfully received the packet, are multicasted until *K* rounds. After *K* rounds, any group members who have not successfully received the packet, are unicasted until all group members correctly receive the packet.

The RMT framework could be classified into 2 parts, which are shown in Fig. 2. The first part is the error control part presented in section IV, which is based on a hybrid of ARQ, and Adaptive Forward Error Correction (AFEC). The second part is the congestion control part presented in section V. In the error control part, in order to reduce feed back implosion and improve scalability, the sender adapts the proactive redundancy level according to the number of active receivers and estimated packet loss rate. The congestion control part relies on the "packet-conservation" principle [13]. The proposed congestion control uses a single-rate and window-based scheme. In order to reduce probability of congestion window unnecessarily reduced because of spatially uncorrelated loss in last mile link, we use a simple loss filter algorithm (WLA). This loss filter algorithm uses a set of N_Q receivers, Q (called ACKers) to report receiving status in form of ACK (Acknowledgement) messages. Each ACKer sends ACKs, which contain the sequence number of newest received data packet, a 32-bit bitmap indicating the receiving status of the previous 32 packets, echoed time stamp, and estimated forward bandwidth. Furthermore, we set slow start threshold and current congestion window based on estimated bandwidth when time out or loss similar to TCPW [8] while standard TCP [13] sets slow start threshold to half of the current congestion window because the sender assumes that someone is sharing the network. In addition, our protocol deals with ACKer changes and responses to variations in network conditions such as ACKer failure.

TABLE I
Abbreviations

Abbreviations	Meaning
RM	Reliable Multicast Transport Protocols
Net	Network layer
LL/MAC	Data link layer: Logical Link Layer and Media Access Control layer
Phy	Physical Layer
Er_{US}	SAN's Error module for return traffic
Er_{DS}	SAN's Error module for forward traffic
Er_{UM}	MES's Error module for forward traffic
Er_{DM}	MES's Error module for return traffic

IV. Error Control Part

The operation of error control part of RMT based *N3* defined in [19] is described as follows. The sender multicasts a block of *n* packets, which consists of *k* original packets and *h* parity packets. The selection of *h* and *k* will be presented in subsection A. Multicast receivers check the packets they receive for errors through unique sequence numbers and unique FEC block numbers. This is done either during the reception of the next block or the session message if no data is being received. The session messages are periodically sent in a new packet or attached a field in data packets. If a multicast receiver has only received *x* distinct packets and if *x* is less than *k*, i.e. there are losses; it requests *k-x* new parity packets. If a receiver needs to repair a packet in the reported block, it sets a reschedule sending NAK timer. If the requested parity packets are not received before the timer expires, the receiver resends the NAK. This process continues until the receiver successfully receives the requested packets. After receiving NAK, the sender sends new parity packets or buffers for transmission schedule to facilitate the repairing of any lost packets. The transmission rate is adapted following the congestion control part described in Section V.

A. Adaptive FEC Algorithm in the error control part

The proposed protocol uses the adaptive redundancy algorithm shown in Fig. 3. It takes the wireless loss parameters as input, and calculates the number of required redundancy packets by averaging the probability of a multicast packet needing retransmission after adding FEC, $q(j)$. The value of $q(j)$ is calculated for the temporally uncorrelated loss model in (1). The criteria parameter C_R determines the number of redundancy packets, *h,* to be retransmitted. The value of C_R is $1/R\sum_{j=1}^{R} q(j)$. The value of *h* is iteratively determined by increasing *h* until the value of C_R is less than X/R, where X is average packet loss rate of all receivers and *R* is the number of receivers. Let *q*(*j*) be the probability that an original packet is requested after FEC recovery, given by:

$$q(j) = \frac{1}{k}\sum_{i=0}^{k-1}(k-i)p^{n-i}(j)(1-p(j))^{i}\begin{pmatrix} n \\ n-1 \end{pmatrix}, \qquad (1)$$

where $p(j)$ is packet loss rate for MES *j* calculated in (2).

B. Wireless loss parameter estimation (LE)

Each receiver estimates the wireless loss parameters. A receiver calculates the loss based on the gap of packet sequences or multicast session message. Therefore for each FEC block, the receiver calculates the probability of loss; $p_b(j)$ from the number of losses divided by the number of packets in the FEC block for the first transmission i.e. excluding retransmitted packets. The average packet loss rate for receiver *j,* $p(j)$, could be recursively calculated using the exponentially weighted moving average filter shown by following equation:

$$p(j) = (1-z)p(j) + zp_b(j), 0 < z < 1, b = 0,1,2....N_B \qquad (2)$$

where N_B is the number of FEC blocks and *z* is a smoothing factor, which is greater than zero and less than one. In simulation we set *z* to be 0.1.

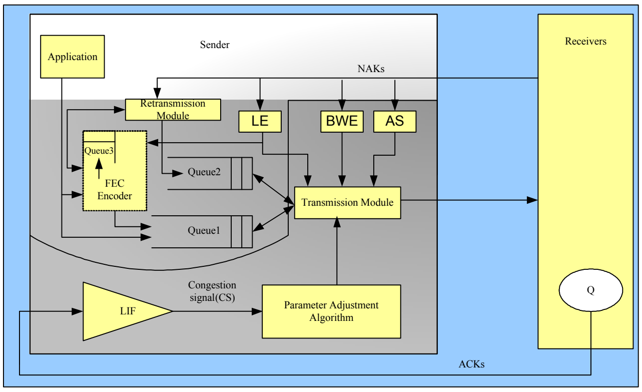


Fig. 2 Reliable Multicast Architecture

C. Negative acknowledgements

At the end of a block of FEC (detected by receiving multicast session or by receiving a starting packet of a new FEC block), a receiver calculates the number of requested parity packets for the FEC block. To avoid NAK implosion, the receiver waits for a random time, t_F, which is randomly selected between 0 and t_B before sending the NAK packet to the sender. After sending a NAK, the receiver starts retransmission timer which expires at next 2 SRTTs (estimated round trip time). Each NAK packet consists of FEC block index, the number of requested packet in the FEC block, estimated packet loss rate, number of requested rounds for the FEC block and echoed time stamps plus t_F (for the first requested round only). The echoed time of first requested round is used to estimate round trip time by the sender.

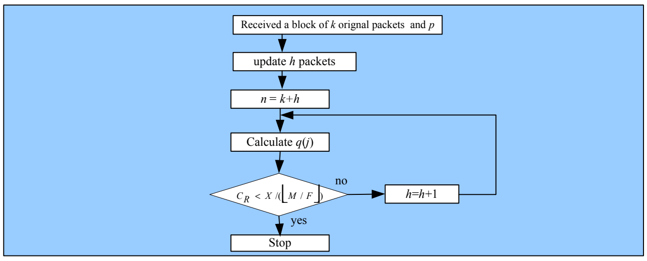

Received a block of *k* orignal packets and *p*
update *h* packets
$n = k+h$
Calculate $q(j)$
$C_R < X/(\lfloor M/F \rfloor)$
no
$h=h+1$
yes
Stop

Fig. 3 the Adaptive FEC Algorithm

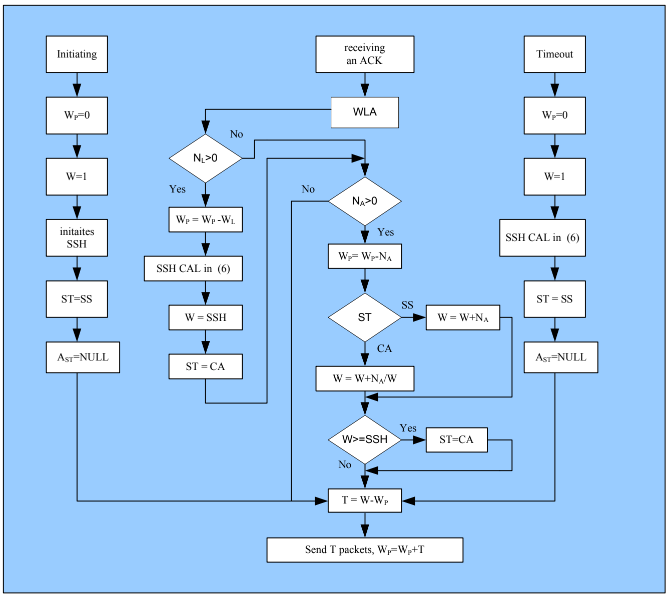


Fig. 4 Window Regulation

V. Congestion Control Part

The proposed congestion control scheme tries to achieve fast response and eliminates the so-called "drop-to-zero" problem i.e. the session rate dropping to value of wrong loss estimation in the presence of uncorrelated losses. In addition, the scheme eliminates unnecessarily decrease in the multicast session rate when losses occur from wireless link errors. Furthermore, the scheme still retains scalability because we use only a small number of receivers to be ACKers. In order to achieve these objectives, we design the proposed congestion scheme as follows. We use a simple loss indication filter, which could filter last mile losses out. The remaining losses are the losses occurred from wired link and the first mile link. After using the loss indication filter algorithm, the communication could apply existing wireless TCP protocols to adjust congestion control parameters. In our protocol, we use a window-based controller, which will be described in next subsection.

A. *Window Regulation in Congestion Control Part*

The operation of window regulation in our proposed congestion control algorithm is shown in Fig. 4. The sender maintains the estimated number of packets outstanding in the path (W_P), the congestion control window size (*W*), the token count (*T*), slow start threshold *(SSH)* and congestion control state *(ST)*. In our protocol, we assume that *SSH* and *W* are in unit of packets. The congestion control states could be divided in to two states: Slow Start (*SS)* and Congestion Avoidance (*CA*). The operation of the congestion control is presented as follows. After receiving an ACK, the sender performs loss filter algorithm (WLA) described in next section. The WLA gives the number of lost packets, $N_{L,}$ and the number of successfully received packets, N_A. If the number of loss packets is greater than 0, the sender calculates the slow start threshold (*SSH*) as in (3) and sets *W* to be *SSH*. Then it sets the congestion control state to Congestion Avoidance. If the number of successfully received packets is more than zero, the sender increases the window the same as AIMD (Additive Increase Multiplicative Decrease). To avoid stalls, the sender has timer to restart probing the network after it could not send packets for a limited time. In our simulation, we set the time interval to 2 *SRTTs*.

$$SSH = \begin{cases} E_{BW} xSRTT / M_{SS} & if\ E_{BW} xSRTT / M_{SS} > 2 \\ 2, & otherwise \end{cases} \tag{3}$$

Where E_{BW} and *SRTT* are estimated bandwidth and average round trip time which are described in (5) and (6), respectively. The variable M_{SS} is the data packet size in bits.

B. *Acknowledgements*

For each data packet, N_Q receivers are in charge of sending positive acknowledgement (ACKs). Each ACK contains the sequence number of the newest received data packet, a 32-bit bitmap indicating the receiving status of the previous 32 packets, and echoed time stamp.

C. *A New Simple Wireless Loss Indication Filter*

Three types of loss filter algorithms have been proposed in the literature. The basic loss indication is Worst Estimate-Based Tracking approach (WET) [21]. The second one is the Random Listening algorithm (RLA) [23]. The third one is the Linear Proportional Response approach (LPR) [6].

In [21] the sender elects a group representative (*Acker* whose throughput is the worst). This algorithm is not suitable for satellite networks because of several reasons as follows. The worst receiver might change rapidly. The source unnecessarily reduces the congestion control window because of wireless link errors with high probability. To reduce these problems, we present a simple loss filter algorithm, Wireless loss Listening Algorithm (WLA) shown in Fig. 5. The objective of the proposed loss filter algorithm is to pass only losses occurring between the multicast sender and the satellite. The multicast sender distinguishes whether the packet losses occur between the multicast source and the satellite or the packet losses occur in the last mile link. This could be done by comparing the receiving status from all ACKers. If a packet is lost for all active ACKers, the sender assumes that the loss occurred between the sender and the satellite. In order to avoid packets out of sequences, A packet is assumed to be lost if the sender has not received an ACK for the packet in a number of subsequence ACKs from all ACKers, *dupack threshold*. This value is set to 3 in our simulations otherwise stated.

D. Bandwidth Estimation (BWE)

In our congestion control, the sender sets slow start threshold following estimated forward bandwidth similar to TCPW [8]. The operation of forward bandwidth estimate is described as follows. The sender estimates the bandwidth using the exponentially weighted moving average filter shown in

$$r_i(j) = (1-y)r_{i-1}(j) + yt_i(j)\ , \tag{4}$$

where y is a constant between 0 and 1. The value $t_i(j)$ is $M_{SS}/(A_i - A_{i-1})$, A_i is the time that an ACK packet has been successfully received and A_{i-1} is the time that the previous ACK packet was received. After the sender could estimate average bandwidth as in

$$E_{BW} = \frac{1}{N_{Q_s}}\sum_{j\in Q_s} r_i(j) \tag{5}$$

where Q_s and N_{Q_s} are the set of ACKers whose t_i are received in predefined calculating time and the number of Q_s, respectively.

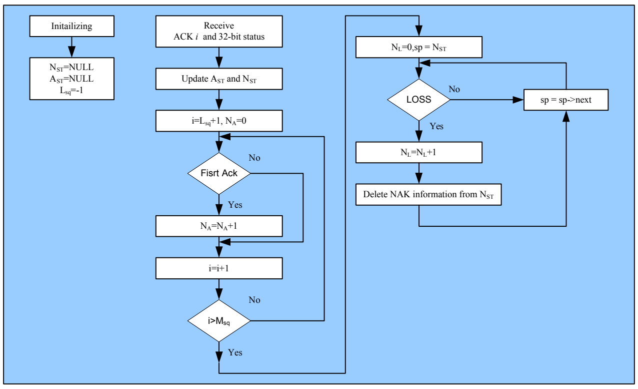


Fig. 5 WLA Operation

E. Round Trip Time Estimation at the sender

The average round time estimation of receiver *j*, $SRTT(j)$, could be recursively calculated using the exponentially weighted moving average filter shown by following equation.

$$SRTT(j) = (1-g)SRTT(j) + gRTT(j) \tag{6}$$

where $RTT(j)$ is sample round trip time, g is a constant between 0 and 1. In simulation we use g = 0.1. The sample time is calculated from echoed time stamp sent by receivers in form of an ACK or a NAK.

After predefined calculating time, the sender could estimate average round trip time as in:

$$SRTT = \frac{1}{N_{Q_s}}\sum_{j\in Q_s} SRTT(j) \tag{7}$$

where Q_s and N_{Q_s} are the set of active ACKers whose new SRTTs are received in predefined calculating time and the number of Q_s, respectively.

F. Acker Election and Tracking (AS)

In order to track the high capacity receivers to be ACKers, we use normalized throughput to track ACKers. The operation of ACKers election is performed by choosing the N_Q receivers whose normalized throughputs are highest. We use the N_Q high capacity receivers to be ACKers because the receivers could send ACKs quickly and efficiently. This leads to the high achievable throughput. This selection differs from WET, which selects the worse receiver to be the ACKer. This is because in wired network, the sender adjusts the rate to the slowest receivers. The normalized throughput derived in [16] and simplified by [21] is calculated as:

$$T_{nm}(j) = \frac{1}{SRTT(j)\sqrt{p(j)}} \tag{8}$$

where $SRTT(j)$ and $p(j)$ are estimated round trip time in (6) and estimated packet loss rate for receiver j in (2), respectively. To reduce the number of ACKer switches, the selection of new ACKers is shown as follow. For every ending of a predefined calculating time (in simulation, we set the predefined calculating time to be 4 SRTTs), the sender sorts the normalized throughputs' all active receivers except the current ACKers. The sender considers only the first N_Q highest throughput receivers (say, $T^1_{nm}(x_1),\dots,T^{N_Q}_{nm}(x_{N_Q})$) where $x_1,\dots,x_{N_Q}$ are the receivers in the first N_Q highest throughput receivers. Also the sender sorts the normalized throughputs of current ACKers who pass a report test (say $T^1_{Anm}(y_1),\dots,T^{N_Q}_{Anm}(y_{N_{Qs}})$, where $y_1,\dots,y_{N_{Qs}}$ are the ACKers ordered in descending throughputs. The variable N_{QS} is the number of ACKers passed the report test. The report test is performed by counting number of received ACKs ($C_{ACK}(j)$) from ACKer, *j* during the predefined calculating time, If $C_{ACK}(j)$ is less than 50 percent of number of packet sent during the predefined calculation time, the ACKer *j* does not pass the report test. Based on this report test the predefined calculating time should be more than 2 SRTTs. This report test is utilized to solve the problem of ACKer failure. Then the sender elects new ACKers by comparing the normalized throughputs of the N_{QS} receivers and the N_{QS} current ACKers as:

$$y_i = \begin{cases} x_i \quad ,i=1,\ldots,N_F, N_F = N_Q - N_{QS} \\ y_{i,} T^i_{Anm}(y_{i-N_F}) \geq cT^i_{nm}(x_i) \ ,i = N_F+1,\ldots,N_Q \\ x_i, \qquad otherwise,\ i = N_F+1,\ldots,N_Q \end{cases} \tag{9}$$

where c is constant between 0 and 1. If the value c is 0, the ACKers are fixed at the start connection of the multicast session. On the other hand, if the value of c is 1, the ACKers are frequently changed.

The protocol operation for ACKer tracking is described as follows. The sender sends 2 lists: one for list of dedicated ACKers and one for list of current ACKers. This list could be sent in form of a new packet or added to data packets. The first list is to notify new ACKers and the last list is to notify current ACKers. If a receiver receives the list of dedicated ACKers consisting of its address, it starts sending ACKs. If the sender receives the first ACK from the receiver, it changes the list of current ACKers by replacing old ACKer with the new ACKer. If any current ACKers receive the list of current ACKer not consisting of theirs addresses, the ACKers stop sending ACK. Therefore based on this operation, for any time, there are at least N_Q receivers who send ACKs.

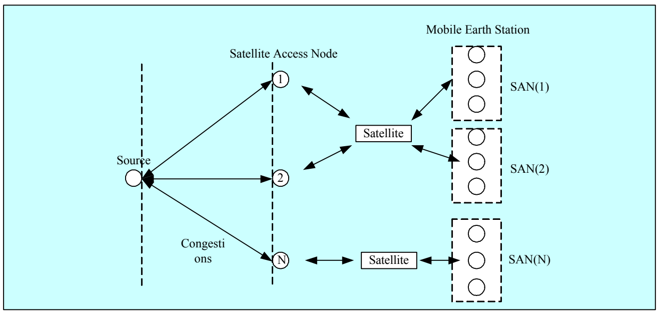


Fig. 6 Multiple-SAN Network Structure

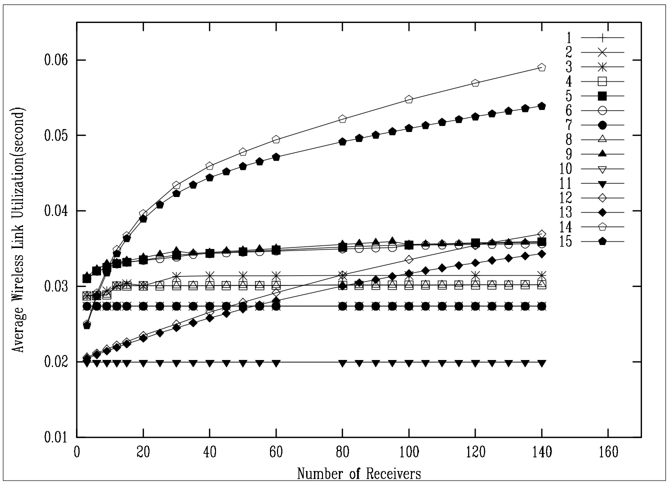


Fig. 7 Average Wireless Link Utilization per Original Packet

VI. Extension for Multiple Sans

In this section, we provide the extension for M Satellite Access Nodes shown in Fig. 6. This scenario is deployed for satellite networks such BGAN [1] and IPDS [2]. We assume the sender knows serving SAN of each receiver. The congestion control is similar to one SAN scenario. The source performs M wireless loss filter modules and M ACKer Selections (AS). One WLA and one AS are installed for one SAN. If the transmission module receives a CS (loss) from any WLA module, it sets up congestion window following (4). The error control part performs the same as in one SAN scenario.

VII. Simulation results and Numerical results

In this section, we present simulation results of RMT over BGAN scenario 2 using NS2 [3]. The detail of BGAN simulation scenario could be found in [9, 10]. For this scenario, we selected a bearer type 34 (422.4 kbps) for forward bearer. We fixed return bearer types to 76(168 kbps) for all simulated receivers. We used return slot plans, 7 and 9, which also use 5-millisecond slot sizes [1]. Both common wireless losses and last mile wireless losses are modeled as temporally uncorrelated losses. Simulation results for temporally correlated losses could be found in [9]. The results of simulation in Sections A, B, E, F, and G are averaged from about 1000 iterations. The results of simulation in Sections C and D are collected from only one iteration. The objective of this simulation is to present performance of RMT in terms of scalability, achievable throughput and Goodput, the proposed wireless loss filter algorithm, and the proposed congestion control algorithm.

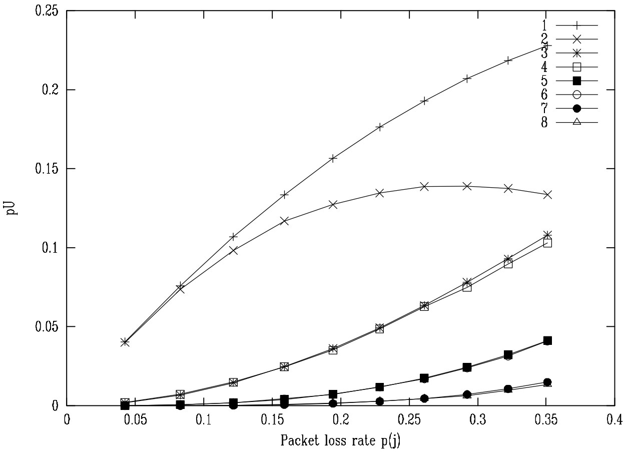


Fig. 8 Probability of congestion window unnecessarily reduced because of spatially uncorrelated wireless losses, p_U

A. Scalability

In this subsection, we present the scalability of RMT protocol in terms of average wireless link utilization. Fig. 7 shows the average wireless link utilization per original packet. Curve 1, curve 3 and curve 5 present RMT's simulation results of average wireless link utilization per original packet when last mile BERs of 0, 10^{-6} and 10^{-5}, respectively. Curve 2, curve 4 and curve 6 present RMT's numerical results detailed in [19] of average wireless link utilization per original packet when last mile BERs of 0, 10^{-6} and 10^{-5}, respectively. Curve 7, curve 8 and curve 9 present N3 (with K=1) 's numerical results of average wireless link utilization per original packet when last mile BERs of 0, 10^{-6} and 10^{-5}, respectively. Curve 10, curve 12 and curve 14 present MFTP's simulation results of average wireless link utilization per original packet when last mile BERs of 0, 10^{-6} and 10^{-5}, respectively. Curve 11, curve 13 and curve 15 present MFTP's numerical results of average wireless link utilization per original packet when last mile BERs of 0, 10^{-6} and 10^{-5}, respectively. Details of MFTP analysis could be found in [19]. It could be seen that the average wireless link utilization of RMT is virtually independent of number of receivers and less than that of *N3* (with K=1) and MFTP.

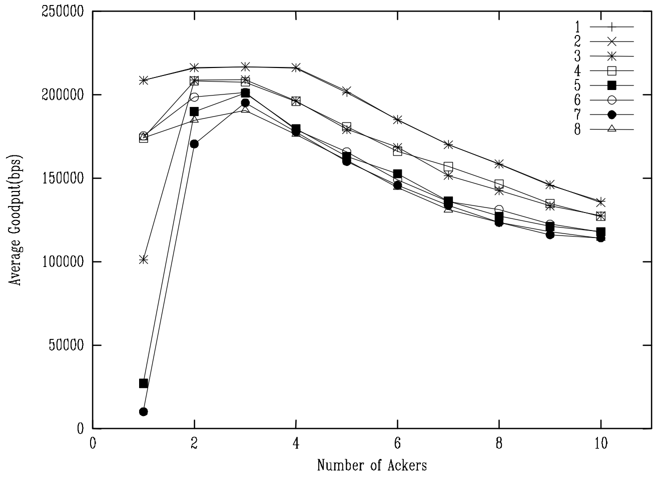


Fig. 9 RMT Goodputs versus numbers of ACKers for different wireless loss rates in last mile link

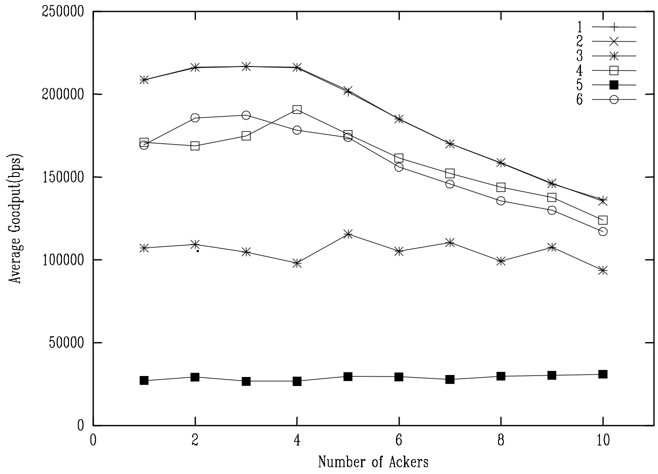


Fig. 10 RMT Goodputs against number of ACKers for different common wireless loss rates

B. *Congestion Control Part*

This experiment is to show the improvement of the proposed WLA and BWE for congestion control. Fig. 8 shows the probability of congestion window unnecessarily reduced because of common wireless losses, p_U . Curve 1 and Curve 2 present the probability of WET obtained form analysis in [19] and simulation. Curve 3 and Curve 4 shows the probability of WLA obtained form analysis in [19] and simulation when number of ACKers is 2. Curve 5 and Curve 6 show the probability of WLA obtained form analysis in [19] and simulation when number of ACKers is 3. Curve 6 and Curve 7 show the probability of WLA obtained form analysis in [19] and simulation when number of ACKers is 4. The difference between simulation results and numerical results is that because in analysis, we do not consider packet losses during time out. It could be seen from the figure, the probability of WLA is much less than that of WET. For WLA, if the higher the number of ACKers is, the less the probability is. However, if the number of ACKers is too high, the feed back implosion occurs. This leads to the achieve Goodput is significantly reduced.

Next experiment is to investigate the effect of number of ACKers and congestion control with bandwidth estimation. The parameters of experiment used are described following. The number of receivers is 10. The number of transferred packet is 1000 packets. Packet size is 500 bytes. Simulation results are shown in Fig. 9 and Fig. 10. Curve 1, curve 3, curve 5, and curve 7 show Goodputs of RMT without bandwidth estimation when last mile BERs of 0, 10^{-6}, $5x10^{-6}$ and 10^{-5}, respectively. Curve 2, curve 4, curve 6 and curve 8 show Goodputs of RMT with bandwidth estimation when last mile BERs of 0, 10^{-6}, $5x10^{-6}$ and 10^{-5}, respectively. From Fig. 9, it could be seen that the Goodputs of RMT with BWE are higher than those of RMT without BWE. Also, it could be seen that the number of ACKers of about 3, RMT gives the highest Goodput for all packet loss rates. This is because if the number of ACKers is lower, the probability p_U is higher. If the number of ACKers is too high, the feed back implosion occurs.

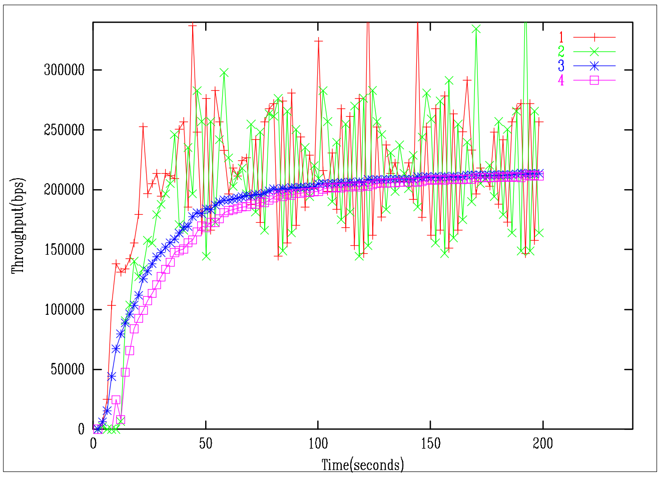


Fig. 11 Intra protocol Fairness without wireless loss

Fig. 10 shows RMT Goodput against number of ACKers for different common wireless loss rates. Curve 1, curve 3, and curve 5 show Goodputs of RMT without BWE when first mile BERs of 0, 5x10^{-6} and 10^{-5}, respectively. Curve 2, curve 4, and curve 6 show Goodputs of RMT with BWE when first mile BERs of 0, 5x10^{-6} and 10^{-5}, respectively. It could be seen that Goodputs of RMT without BWE are much less than those of RMT with BWE. This is because in the case of common wireless loss, the WLA could not filter wireless losses out. Therefore to improve Goodputs, RMT with BWE performs much better.

C. *Intra Protocol Fairness*

One of the requirement of reliable multicast transport protocols [15], it should be fair to other flows. In this experiment, we test 2 flows of RMT to test its fairness. The simulation scenario is the same as in Fig. 1 The number of MESs is 6. The first 3 MESs are attached RMT flow 1 receivers. The link queue size is 32 packets. The last 3 MESs are attached RMT flow 2 receivers. The number of ACKers is 3. The multicast sources of flow 1 and flow 2 started at 2 second and 10 second, respectively. Fig. 11, Fig. 12 and Fig. 13 show the evaluation of two competing RMT sessions through the same bottleneck without wireless loss, with common wireless loss and with spatially uncorrelated wireless loss, respectively. Curve 1 and curve 3 present instance throughput and aggregated throughputs of flow 1. Curve 2 and curve 4 present instance throughput and aggregated throughputs of flow 2. It could be seen that there is a good share of bandwidth.

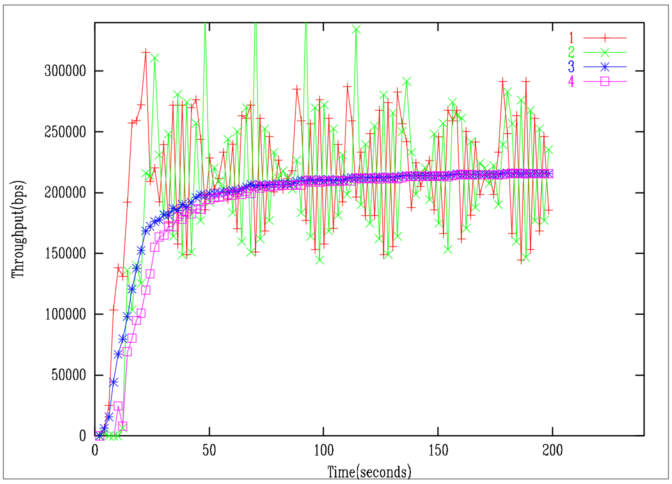


Fig. 12 Intra protocol Fairness with common wireless loss BER=10^{-6}

D. *Inter Protocol Fairness*

To show inter protocol fairness, in this subsection, we test RMT performance in presence of competing a TCP flow. The simulation scenario is presented in Fig. 1. The number of MESs is four. The first 3 MESs are attached RMT receivers. The last MES is attached TCP sink. The sources of RMT and TCP are attached at the source node. The link queue size is 32 packets. In order to verify the behavior of competing TCP and RMT flows, we have run a large number of simulations with two types of flows. Fig. 14, Fig. 15 and Fig. 16 show the instant throughput of TCP and RMT without wireless loss and with wireless loss, respectively. The legends of Fig. 14, Fig. 15 and Fig. 16 are 1: TCP instance throughput with maximum window of 32 packets, 2: RMT instance throughput with maximum window of 32 packets, 3: TCP instance throughput with maximum window of 64 packets, 4: RMT instance throughput with maximum window of 64 packets.

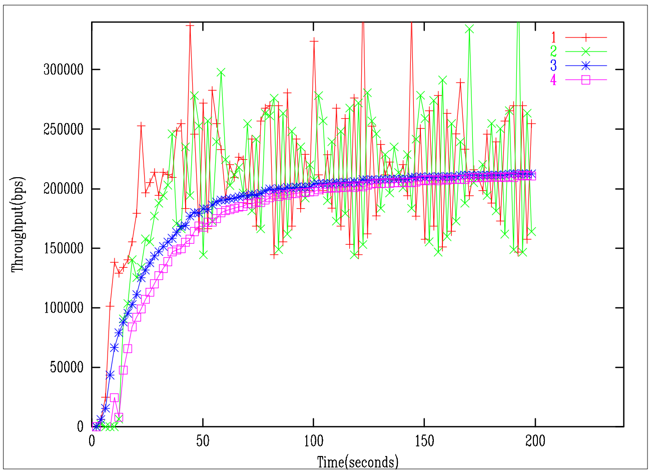


Fig. 13 Intra protocol Fairness with spatially and temporally uncorrelated wireless loss in last mile link, BER=10^{-6}

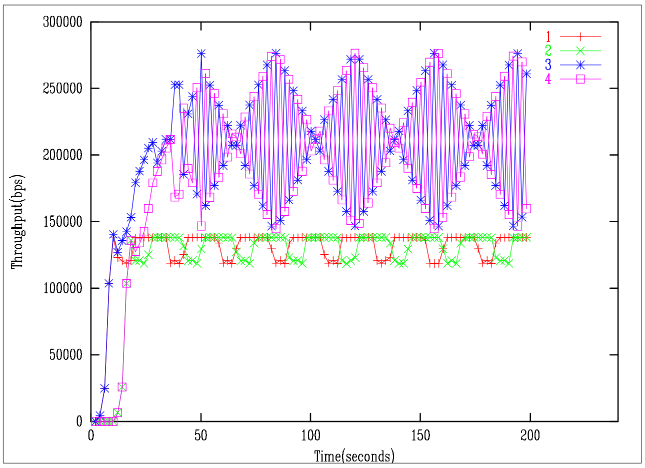


Fig. 14 Inter protocol Fairness with non wireless loss

E. *ACKer Tracking*

In the case of one SAN, if any current ACKers provide low performance such as high loss rate or connection failure. The sender should elect new ACKers to report receiver packets instead of the problem ACKers. This experiment tests the performance of ACKer tracking algorithm. The number of receivers is 10. The initialized number of Ackers is 3. The last mile wireless loss rate is 5x10^{-6}. Fig. 17 shows the instant throughput of the RMT with different *c* values. Fig. 18 shows average number of ACKers with different *c* values. It could be seen that the higher the c value is, the higher the average number of ACKers is. The higher number of Ackers is, the lower the achievable throughput is. From Fig. 18, the number of ACKers is high that initialized ACKers, because right after ACKer switching, the old ACKer has to send ACKs until it receives a release notification from the sender. The sender sends the release notification after receiving any ACK from the new ACKer. The processing of changing ACKers takes at least one round trip time.

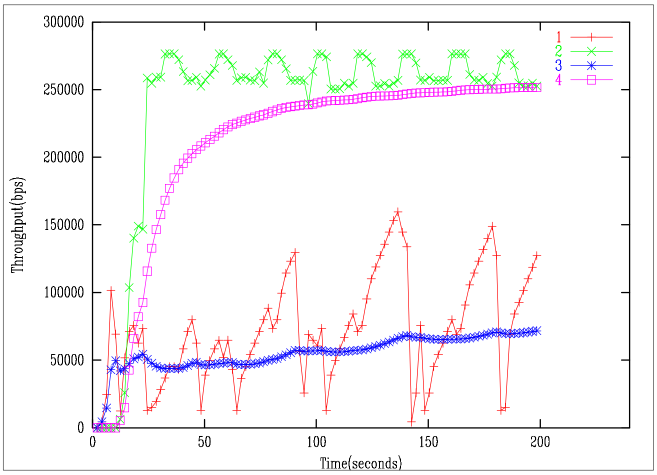


Fig. 15 Inter protocol Fairness with common wireless losses with BER=10^{-6}

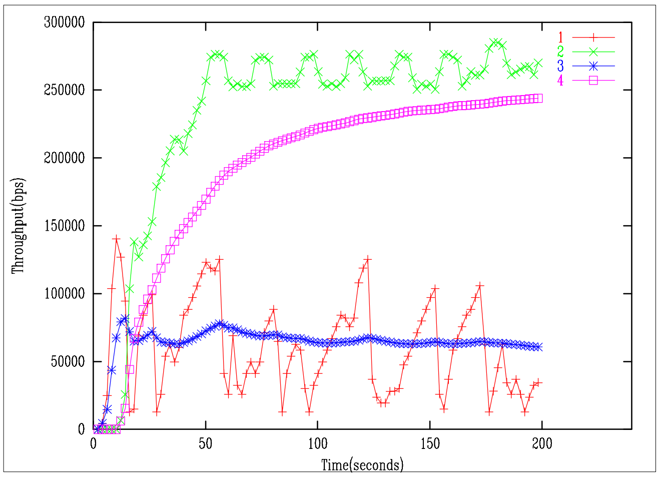


Fig. 16 Inter protocol Fairness with spatially and temporally uncorrelated losses with BER=10^{-6}

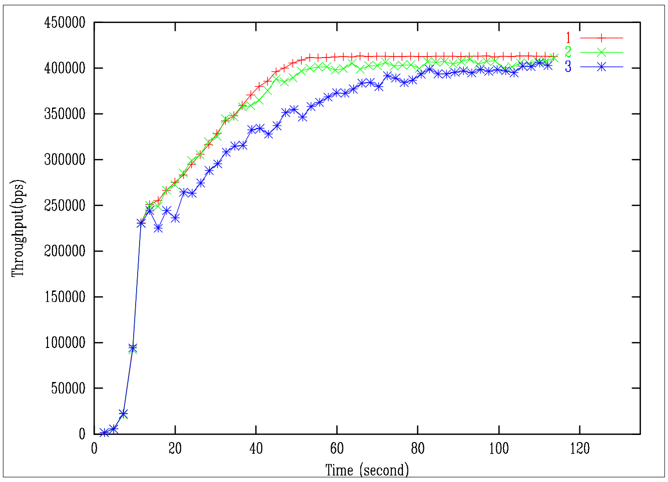


Fig. 17 Throughputs for different value of c, 1: c=0, 2: c=0.5, 3: c=1.0

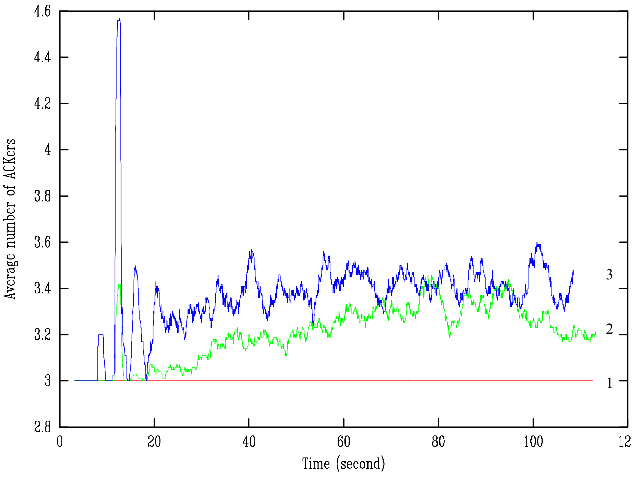


Fig. 18 Average number of ACKers for different values of c, 1: c=0, 2: c=0.5, 3: c=1.0

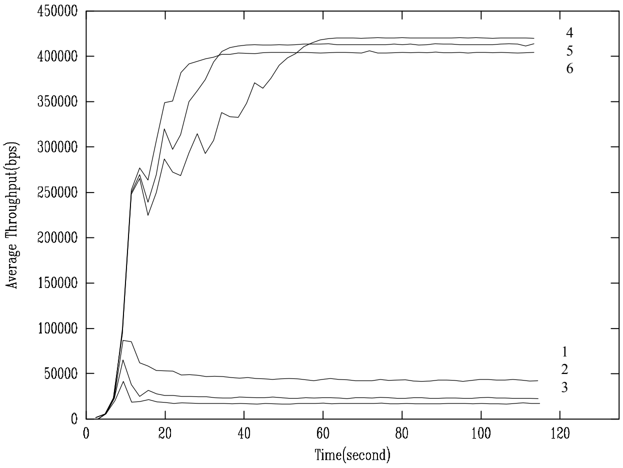


Fig. 19 RMT instance throughputs and TCP instance throughputs

F. RMT throughputs compared to TCP throughputs

In this section, we present RMT throughputs for different wireless loss models. We compare the throughput of RMT to that of TCP. TCP is selected because it is the de facto standard for reliable unicast transport in the Internet today. In simulation of RMT, we tested 3 receivers. All receivers are ACKers. In simulation of TCP, we tested one connection. The common first mile wireless BER is 10^{-6}. Fig. 19 shows the comparison throughputs of RMT and TCP. Curve 1, curve 2 and curve 3 present instance receiving throughputs of TCP when loss rates are 10^{-6}, $5x10^{-6}$ and 10^{-5}, respectively. Curve 4, curve 5 and curve 6 present instance receiving throughputs of RMT when loss rates are 10^{-6}, $5x10^{-6}$ and 10^{-5}, respectively. It could be seen that our protocol performs much better than TCP. This is because our protocol reduces the probability of congestion window unnecessarily reduced because of wireless losses. In addition, our protocol employs bandwidth estimation to set congestion window and slow start threshold in a congestion episode.

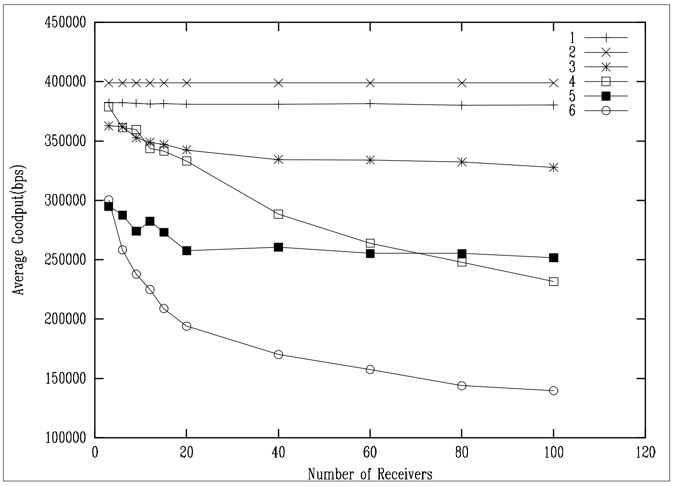


Fig. 20 RMT Goodputs and MFTP Goodputs when transmission rate is less than forward bandwidth

G. RMT Goodputs compared to MFTP Goodputs

In this section, we compare RMT Goodputs to MFTP Goodputs. The transmission rate in MFTP is not dynamically adapted to network conditions. It could set transmission rate at the sender. The MFTP transmission rate is 422000 bps. The number of transferred packets is 5000 packets whose size is 1000 bytes. The results are shown in Fig. 20. Curve 1, curve 3 and curve 5 show the Goodputs of RMT when last mile loss BERs are 0, 10^{-6} and 10^{-5}, respectively. Curve 2, curve 4 and curve 6 show the Goodputs of MFTP when last mile loss BERs are 0, 10^{-6} and 10^{-5}, respectively. It could be seen from the figure that at BER=0, the Goodput of MFTP is higher than that of RMT because MFTP could send full link all connection time while RMT has slow start phase and congestion avoidance phase. At BERs of 10^{-6} and 10^{-5}, the Goodputs of RMT are significantly higher than those of MFTP. This is because RMT employs the proposed adaptive forward error correction while MFTP is round (pass)-based transmission without forward error correction.

VIII. CONCLUSION AND FUTURE WORKS

We have presented a model-based reliable multicast transport protocol for satellite networks. The protocol is designed to improve reliable multicast transport protocol performance in terms of scalability, throughput and Goodput and also be fair with other traffic in the networks. This improvement could be achieved by following algorithms:

Scalability is improved by using the proposed adaptive forward error correction. It has been shown in both simulation results and numerical results that the average wireless link utilization and the average number of transmission are virtually independent to the number of receivers.

Feedback implosion is significantly reduced by using a new hybrid adaptive forward error correction and NAK-based ARQ in error control part and the proposed window-based congestion control using a few highest throughput receivers to send ACKs in congestion control part. Simulation results and numerical results show that the wireless link utilization is independent of the number of receivers.

Drop-to-zero problem occurring from last mile wireless losses is significantly reduced by using the proposed wireless loss filter algorithm. The simulation results and numerical results show that the probability of congestion window unnecessarily decreased is much lower than that of TCP and WET.

Common wireless loss is solved by using setting slow start threshold and congestion window following estimated bandwidth similar to [8] when a loss occurs. The simulation results show that the achievable throughput is significantly higher that of TCP.

Long delay is improved by using a new adaptive forward error correction algorithm. This yields that achievable RMT Goodput is much higher than that of MFTP.

Fairness is achieved by mimicking TCP which relies on "packet-conservation" principle.

We are in process of designing multicast connection management. In addition, we intend to pursue the development of control theoretical models that will enable us to study the stability of RMT as a function of the various systems parameters.

REFERENCES

[1] BGAN specification, physical layer. 2003. INMARSAT.

[2] PDS specification, physical layer. 2003. INMARSAT.

[3] UCB/LBLN/VINT Network Simulator (NS), http://www.isi.edu/nsnam.

[4] G. Akkor, M. Hadjitheodosiou, and S. J. Baras. IP Multicast via Satellite: A Survey. 2003. University of Maryland, College Park, Institute for Systems Research *Technical Report* TR_2003-4.

[5] Basu, P. and Kanchanasut, K., "A reliable multicast protocol for unidirectional satellite link", *In Proceeding of Applications and the Internet Symposium*, Jan. 2003.

[6] Bhattacharyya, S., Towsley, D., and Kuros, J., "Novel loss indication filtering approach for multicast congestion control," *Computer Communications Special Issue on Multicast Deployment*, vol. 24, no. 5-6, 2001.

[7] Cao, G. and Wu, J., "Reliable multicast via satellites", *In Proceeding of International Conference on Information Technology: Coding and Computing*, Apr. 2001.

[8] Casetti, C., Gerla, M., Mascolo, S., Sanadidi, M. Y., and Wang, R., "TCP Westwood: End-to-End Congestion Control For wired/Wireless Networks," *Wireless Networks*, vol. 8, no. 5, 2002.

[9] P. Chumchu and A. Seneviratne. Analysis of reliable multicast protocols in satellite networks . 2004. Inmarsat Report.

[10] P. Chumchu and A. Seneviratne. NS-based GEO satellite network simulator. 2004. INMARSAT Report.

[11] Fang, J. and Akyildiz, F. I., "A reliable multicast transport protocol for satellite IP networks", *In Proceeding of IEEE GLOBECOM 2002*, Nov. 2003.

[12] Floyd, S., Jacobson, V., Liu, G. C., McCanne, S., and Zhang, L., "A reliable multicast framework for light-weight sessions and application level framing," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, 1997.

[13] Jacobson, V., "Congestion avoidance and control", *In Proceeding of ACM SIGCOMM*, Aug. 1988.

[14] Jung, M, Nonenmacher, J., and Biersack, E. W., "Reliable multicast via satellite: uni-directional vs. bi-directional", *In Proceeding of KiVS'99* , Mar. 1999.

[15] A. Mankin, A. Romanow, S. Bradner, and V. Paxson. IETF Criteria for evaluating reliable multicast transport and application protocols. RFC 2357 .

[16] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," *Computer Communications Review*, 1997.

[17] K. Miller and K. Robertson. StarBurst Multicast File Transfer protocol (MFTP) specification. 1998. IETF, IETF-Draft <draft-miller-mftp-spec03. txt>.

[18] P.Chumchu, R.Boreli, and A.Seneviratne, "A New Scalable Reliable Multicast Transport Protocol for Satellite Networks (RMT)", *In Proceeding of Softcom 2005*, Sep. 2004.

[19] P.Chumchu, R.Boreli, and A.Seneviratne, "Performance Analysis of reliable multicast transport Protocol for GEO Satellite networks", *In Proceeding of MASCOT 2005*, Sep. 2004.

[20] Paul, S., Sabnani, K. K., Lin, J.-H. C., and Bhattacharyya, S., "Reliable multicast transport protocol (RMTP)," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, Jan. 1997.

[21] Rizzo, L., "pgmcc: a TCP-Friendly single-rate multicast congestion control scheme", *In Proceeding of ACM SIGCOMM*, Jan. 2000.

[22] Sano, T., Shiroshita, T., Takahashi, O., and Yamashita, M., "Monitoring-based flow control for reliable multicast protocols and its evaluation", *In Proceeding of Performance, Computing, and Communications Conference*, 1997.

[23] Wang, A. H. and Schwartz, M., "Achieving Bounded Fairness for Multicast and TCP traffic in the Internet", *In Proceeding of ACM SIGCOMM*, Jan. 1998.

[24] Wang, C. and Leung, V. C. M., "A reliable multicast transport protocol for broadband satellite networks", *In Proceeding of IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions*, Jun. 2002.

Prawit Chumchu is a lecturer at Mahanakorn University of Technology. He received his Bachelor degree (B.Eng) with honour from Thammasat University and Master degree (M.Eng) from Mahanakorn University of technology, Thailand. He has a Ph.D. from University of New South Wales, Australia. His research interests are in digital signal processing for telecommunication, telecommunication systems and mobile computing.

Dr Roksana Boreli is a Senior Research Engineer in NICTA's Networks and Pervasive Computing (NPC) and an Associate Adjunct Professor at the School of Electrical and Telecommunication Engineering, University of New South Wales (UNSW).

Aruna Seneviratne is the program leader of the networks and pervasive computing program at the National ICT Australia. He holds the Mahanakorn Chair of Telecommunication in the School of Electrical Engineering and Telecommunication at the University of New South Wales. He has a Ph.D. from the University of Bath, UK, and a B.Sc (Hons) degree from Middlesex Polytechnic. Since graduating he has held academic appointments at the University Bradford (UK), Curtin University, Australian Defense Force Academy (UNSW) and the University of Technology, Sydney. Outside academia he as worked at the Standard Telecommunication Laboratories (UK), Muirhead and Co (UK), and Telecom Australia (Telstra). He has also spent time at the MASI Laboratory at the *University of Pierre Marie Curie* (Paris), and the Rodeo Group at *INRIA* (Nice) as a visiting professor. His current research interests are in Quality of Service Management particularly in mobile data communication systems.