

Performance Analysis of Concurrent Multicast / TCP Traffic Over Lossy High Delay Links

Sven Ehlert, Dorgham Sisalem
Fraunhofer FOKUS Institute, Germany
{ehlert,sisalem}@fokus.fraunhofer.de

Adam Wolisz
Technical University Berlin, Germany
wolisz@tkn.tu-berlin.de

Abstract -- In this paper we investigate the performance of TCP-friendly congestion controlled multicast traffic over Satellite links. As an example for a multicast congestion control protocol we simulate Multicast Enhanced Loss-Delay Based Adaptation (MLDA) over links showing the typical loss and delay characteristics of satellite links. In the context of our investigation we concentrate on the performance of MLDA over satellite links in terms of bandwidth utilization and fairness towards different TCP versions (Reno, Westwood, Hybla). Finally we show that through the use of a loss differentiation scheme the performance of MLDA can be increased considerably.

Keywords: Satellite, Multicast, TCP-friendly, Error Differentiation, Simulation, MLDA, Reno, Westwood, Hybla, Inter-Arrival-Scheme, ns.

I INTRODUCTION

Satellites are an ideal carrier for broadcasting data to a wide user population, however, they differ from classic terrestrial networks by an increased delay over a lossy link. Besides congestion, common reasons for satellite connection losses are different weather conditions (storm, rain, clouds) which prevent the correct reception of packets.

While there has been a lot of study regarding congestion control in multicast fixed networks [1, 2], there has been close to none in the area of wireless and satellite communication. To deploy multicast over satellite networks, congestion control is deemed mandatory, not only to achieve fair bandwidth sharing with unicast traffic and to prevent overutilization of receivers, but also to adapt to their different capabilities. which might range from fixed high-bandwidth distribution nodes to tiny mobile agents with limited connection abilities.

In this paper we evaluate the possibility and performance of multicast content distribution with concurrent TCP traffic over satellite links. To achieve this, we utilize an advanced multicast scheme, namely the Multicast Enhanced Loss-Delay Based Adaptation (MLDA) algorithm [3]. It has been shown that this scheme already delivers TCP-friendly behavior under classic terrestrial networks [3]. We use MLDA as our multicast delivery scheme and evaluate its performance over a heterogeneous terrestrial / satellite network using simulation tools. We determine the TCP-friendliness of the protocol, and also its capability to make maximum use of the satellite connection. For the TCP-friendly evaluation, we use the standard, Reno version [4], an enhanced TCP version for lossy links, Westwood [5], and a dedicated satellite TCP version, Hybla [6].

Furthermore, we show how to increase bandwidth utilization with lossy links through an error differentiation scheme. Such a scheme helps to determine the cause for packet loss, which may

be in our case link congestion or general link failure. We use the Inter-Arrival-Scheme (IAS) [7] for error differentiation.

II BACKGROUND

A. MLDA

The Multicast Enhanced Loss-Delay Based Adaptation (MLDA) algorithm [3] is a rate-based, layered multicast congestion control scheme designed specifically for IP multicast. Congestion control is achieved by exchanging messages between the sender and the multicast receivers (end-to-end); intermediate router support is not required. It utilizes an enhanced version of the Real-Time Transmission Protocol (RTP) [8] for data delivery. This allows for easy adaption for different application needs, e.g. MLDA has already been adapted to employ a reliable multicast carrier protocol [9].

As a layered multicast scheme the transmission stream is divided into multiple layers to serve different receiver capabilities. Subscribed multicast receivers will always receive a base layer essential for the transmission. Further layers contain additional information to enhance the multicast transcription. All layer rates are dynamically adjusted for optimal performance. Such a multi-rate design is viable to accommodate for receiver heterogeneity.

MLDA is a hybrid sender and receiver-based adaptation scheme with basic message flow as follows.

1. The sender periodically transmits reports containing information about the sent layers.
2. After receiving a sender report each receiver in ($j : j = 0; 1; \dots; n$) measures the loss and delay of the incoming data for a period of time and determines a TCP-friendly bandwidth share (τ_j) the sender could utilize on the path connecting the sender and receiver.
3. Based on the calculated share and the rates of the layers as reported in the sender reports the receivers decide to join a higher layer, stay at or leave the current one.
4. Further, the receivers schedule the transmission of reports indicating their calculated bandwidth share after a random period of T_{wait} . Finally, if a report from another receiver with rate indication similar to τ_j was seen before T_{wait} expires, the receiver suppresses the transmission of its report.
5. Based on the receiver reports, the sender adjusts the sizes of the different layers.

Note, that MLDA does not prescribe a certain algorithm for calculating the congestion state, LDA+ [10] is used in MLDA simulations performed in [3]. LDA+ is an additive increase and multiplicative decrease algorithm [11], that uses loss and bottle-

neck measurements and an analytical model [12] to determine the TCP rate. The work done at [3] suggests MLDA to be efficient in terms of bandwidth utilization and loss reduction, and also to be TCP-friendly [13].

B. TCP Flavors

The original Transmission Control Protocol (TCP) [14] has evolved over time to include several algorithms for congestion control. An actual version of TCP (TCP Reno [4]) includes four mechanisms to handle congestion, known as *Slow Start* (SS) / *Congestion Avoidance* (CA), and *Fast Retransmit* / *Fast Recovery*.

In the network probing phase, the sender increases its window size (w) with each incoming ACK size due to Equation 1 (left side):

$$W^{Reno}_{i+1} = \begin{cases} W_i + 1 \\ W_i + \frac{1}{W_i} \end{cases} \quad W^{Hybla}_{i+1} = \begin{cases} W_i + 2^\rho - 1 & SS \\ W_i + \frac{\rho^2}{W_i} & CA \end{cases} \quad (1)$$

This mechanisms leads to an exponential increase of the window size during Slow Start and a linear increase during Congestion Avoidance. However, TCP's throughput rate can decrease considerably over shared high latency links [15]. Due to the long round trip delay the TCP connection can consume a large value of their window size, that is a large amount of packets have been sent but not yet acknowledged.

The TCP Hybla approach [6] tries to remedy this situation by eliminating the SS and CA algorithm's dependency on the Round Trip Time (RTT). Specifically, window increase rate should match that of a reference connection with a lower reference RTT, defined as RTT_0 . The modified SS / CA algorithm looks like Equation 1 (right side), where ρ is a normalized round trip time defined as RTT / RTT_0 . As TCP Hybla's behavior is highly dependent on the appropriate value for RTT_0 , we will always supply this value to the Hybla name. Thus, the default configuration of TCP Hybla with a RTT_0 value of 25 ms will be denoted as Hybla-25.

TCP Westwood [5] was developed to operate especially on lossy links. It estimates available bandwidth by monitoring the returning ACKs. It uses the bandwidth estimates to set the slow start threshold and the congestion window consistent with the effective bandwidth. Thus, random losses from wireless links do not trigger a rapid rate change, instead the rate estimation is only slightly changed.

C. The Inter-Arrival-Scheme

While in wired networks losses usually occur due to overload situations, satellite networks — as a wireless network — encounter additional losses due to different weather conditions or direct view coverings. Congestion control schemes like MLDA apply loss detection schemes to calculate a fair bandwidth share — the more losses are detected, the more the transmission rate will be reduced. Thereby in the context of satellite

networks the throughput can be reduced unnecessarily if satellite transmission errors are not previously discriminated. More precisely, congestion control actions should be triggered in satellite networks only when a packet loss is caused by congestion. This can be achieved through an *Error Differentiation* scheme.

The *Inter-Arrival Scheme* (IAS) presented by Biaz et al. [7] uses the interarrival time between consecutive packets for differentiating between losses caused by network congestion and other random losses occurring in wireless networks. More precisely, this scheme discriminates wireless losses (l_{WRLS}) and congestion losses (l_{CONG}) using the minimum inter-arrival time (T_{min}) between two consecutive packets. Let T_g denote the time between the arrival of two packets at the receiver side and n the number of packet losses during this time interval, then (Equation 2):

$$\text{if } ((n + 1)T_{min} \leq T_g \leq (n + 2)T_{min}) \text{ then } l_{WRLS} = n \\ \text{else } l_{CONG} = n \quad (2)$$

Work done at [16] shows that the integration of IAS with LDA significantly improves the performance of LDA over wireless links.

III PERFORMANCE EVALUATION TROUGH SIMULATION

A. System Topology and Setup

For our simulations we implement the MLDA and IAS algorithm for ns (Network Simulator) [17]. Our simulation topology is chosen to reflect a simple bulk transfer scenario over satellites (branch_{sat}). With bulk transfer we easily see the limits caused by the available bandwidth, which will directly influence TCP-friendliness. Additionally, for comparison reasons we add in certain situations a terrestrial low-latency network (branch_{ter}) (see Figure 1). Original data traffic originates at the sender, leading through a public router to the orbiting satellite.

The earth station directly communicates with the Geostationary Earth Orbit (GEO) satellite which results in a transmission latency for one way of 125 ms [19], resulting in a total Round Trip Time (RTT) of 500 ms.

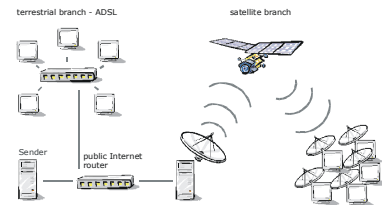


Figure 1. The simulation scenario

To model possible packet losses, we deploy a loss model with bit error rates (BER) of 10^{-8} and 10^{-6} to the downstream satellite link [18]. Receivers_{sat} are supposed to be full-duplex capable with a downlink bandwidth of 300 kbits/s and an uplink of 8 kbits/s [20]. Full duplex capabilities is a requirement for reliable data transfer (TCP) with repair transmissions.

For the additional terrestrial branch we chose an ADSL broadband connection with a downstream of 768 kbits/s and an

upstream of 128 kbits/s. As of today this is becoming a common broadband access in Europe available for end users [22]. In the whole branch_{ter}, the latency for links is set to 10 ms to get a reasonable contrast to the high-latency satellite connection.

For the rate estimation part of MLDA we use in this study the aforementioned LDA+. The packet size is set to 512 bytes for TCP and MLDA traffic. We use three different layers for content distribution. The size of the base layer is set to the minimum reported transfer rate, while the enhancement layers are equally sized so that the additive transfer rate of all three layers match the maximum reported rate. Note that only 90% of the total available bandwidth is used, to provide a reserve for possible overestimation of available resources.

All nodes receive multicast traffic from the sender throughout the whole simulation time of 600 seconds. At every branch we further deploy a TCP agent (Reno, Westwood, Hybla, depending on the simulation) to receive traffic from the sender. We increase the window size of the TCP agents to 64 to accommodate for the high satellite latency.

B. Satellite-Only Traffic Evaluation

We evaluate MLDA at first with only branch_{sat} attached in our topology, with a reasonable low error rate of 10^{-8} . In this setup all nodes receive MLDA multicast traffic, and we conduct individual simulations for each TCP version. To denote a certain receiver in our simulation we use as an index the kind of traffic this receiver handles, e.g. we call a receiver that is served both by MLDA multicast traffic and TCP Reno unicast traffic with receiver_{MLDA,Reno}.

In Figure 2, we can see the transfer rate over time of the Reno simulation. We see the MLDA and TCP Reno transfer rate at receiver_{MLDA,Reno} (Receiver 1) and the MLDA transfer rate at an arbitrary receiver_{MLDA} (Receiver 2). Similar, Figure 3 depicts the Westwood simulation.

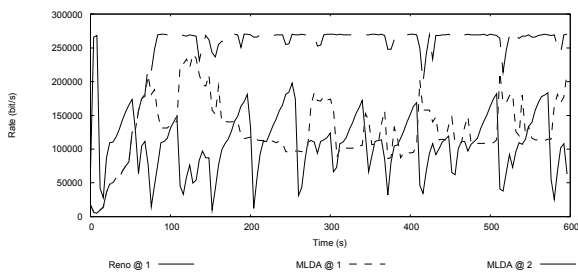


Figure 2. Transfer Rate of MLDA and TCP Reno at branch_{sat} with BER = 10^{-8}

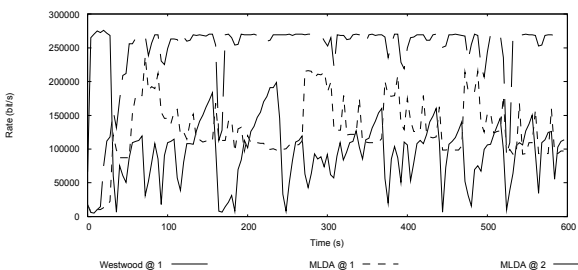


Figure 3. Transfer Rate of MLDA and TCP Westwood at branch_{sat} with BER = 10^{-8}

What we can see from these figures is the very similar behavior of both Reno and Westwood. This results from the fact that Westwood is supposed to enhance TCP performance under higher loss rates than those with a BER of 10^{-8} . Also we can see here the variation in both Reno's and Westwood's bandwidth allocation over time, that is the increasing throughput rate that drops considerably after some time. This is intrinsic to the TCP algorithm and a main reason for TCP's low performance over high delay links [15].

From these figures we can further observe the following: With TCP Reno, and to a lesser degree also TCP Westwood, multiple losses through congestion result in a decrease of the window size at the TCP agent, which can only slowly increase again. This prohibits these TCP versions from reaching their fair share, leading to the MLDA connection craving a higher bandwidth share [24].

We can also see that the receiver_{MLDA} can nearly use the full available bandwidth for MLDA traffic. Periodically short rate reductions occur when MLDA re-adjusts its base layer size due to congestion. Shortly after the rate reduction the effect is compensated by re-adjusting the enhancement layer sizes. It seems that the higher round trip time of a satellite does not affect general MLDA operation negatively.

We compare this to the Hybla-25 simulation, see Figure 4. Here, at receiver_{MLDA,Hybla-25} we can see a nearly constant transfer rate for the TCP stream of about 50% of the available bandwidth, however MLDA traffic reception suffers severely at the same receiver. Additionally, the MLDA transfer rate at receiver_{MLDA} is influenced in a decreasing way, the multicast MLDA connection is severely influenced by the concurrent TCP traffic. Even if this node should accommodate nearly the whole bandwidth with MLDA traffic, it very seldom utilizes more than half of the available bandwidth. This undesirable behavior is mainly due to the fact that we only use three layers in the MLDA stream, which imposes a granularity problem [23].

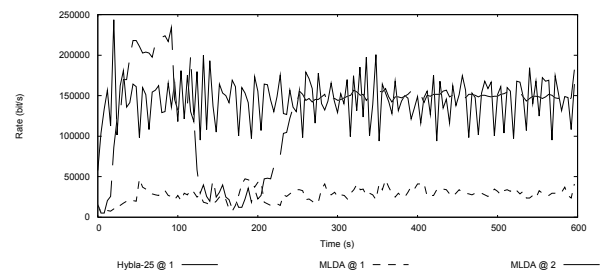


Figure 4. Transfer Rate of MLDA and TCP Hybla-25 at branch_{sat} with BER = 10^{-8}

However, if we lower the RTT_0 value for Hybla, it should be fairer to other connections. In an additional simulation we set this value arbitrarily to 270 ms, which is exactly half of our satellite RTT. From the inspection of Figure 5 two characteristics become immediately clear. First, the bandwidth utilization of receiver_{MLDA} is again nearly at the upper limit, and second, the bandwidth share of TCP and MLDA is approximately equal at receiver_{MLDA,Hybla-270}. Thus, for this special case the RTT_0 value of 270 ms serves our purpose better.

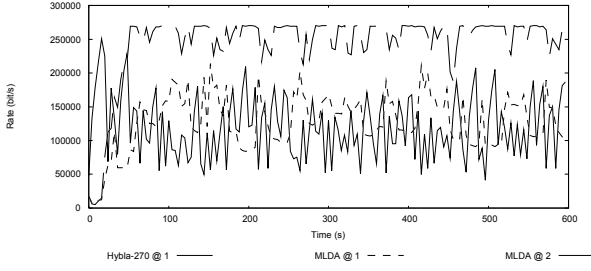


Figure 5. Transfer Rate of MLDA and TCP Hybla-270 at $\text{branch}_{\text{sat}}$ with $\text{BER} = 10^{-8}$

Still, all presented figures can only show an individual test run over time and do not take any variations into account. To get more reliable results, we run these simulations 10 times. In Table 1 we have listed the average bandwidth utilization at different receivers with different TCP versions. Additionally, we determine f as indicator for a fair bandwidth distribution between TCP and MLDA. f is defined as $\tau_{\text{TCP}} / \tau_{\text{MLDA}}$, and a value of 1.0 would indicate an optimum equal distribution between TCP and MLDA.

Table 1. Transfer Rate at $\text{BER} = 10^{-8}$ (in brackets the percentage of the available bandwidth utilized).

	Reno	Westwood	Hybla-25	Hybla-270
τ_{MLDA} at receiver _{MLDA} (in kbit/s)	244.68 (82%)	228.56 (76%)	114.91 (38%)	249.81 (83%)
τ_{MLDA} at receiver _{TCP,MLDA} (in kbit/s)	135.22 (45%)	132.31 (44%)	29.59 (10%)	124 (41%)
τ_{TCP} at receiver _{TCP,MLDA} (in kbit/s)	111.08 (37%)	112.01 (37%)	144.28 (48%)	125.26 (42%)
f	0.82	0.85	4.88	1.01

Our previous observations match the computed average values. Only in the Hybla-270 simulation, MLDA can nearly utilize the full available bandwidth at receiver_{MLDA} (83%, remember that we cut only allow for 90% as a reserve) and gain nearly an optimum fairness value with $f=1.01$. All other simulations fall short from reaching TCP fairness.

C. Receiver Heterogeneity Evaluation

To evaluate MLDA in a heterogeneous topology, we run another simulation with both, $\text{branch}_{\text{sat}}$ and $\text{branch}_{\text{ter}}$, attached. The average bandwidth use is depicted in Figure 2, we consider TCP-Reno traffic with $\text{branch}_{\text{ter}}$ and TCP-Hybla-270 traffic with $\text{branch}_{\text{sat}}$. We can see a nearly optimum usage of the bandwidth resource at $\text{branch}_{\text{ter}}$: receivers_{MLDA} can exploit 84% of the available bandwidth for MLDA traffic, while receiver_{Reno,MLDA} can utilize a fair amount of both with $f = 1.07$. $\text{branch}_{\text{ter}}$ seems to make optimum use of the available resources.

Table 2. Transfer Rate for $\text{branch}_{\text{sat}}$ and $\text{branch}_{\text{ter}}$ (% of available bandwidth utilized).

	$\text{branch}_{\text{sat}} / \text{Hybla-270}$	$\text{branch}_{\text{ter}} / \text{Reno}$
τ_{MLDA} at receiver _{MLDA} (in kbit/s)	202.99 (68%)	648.0 (84%)
τ_{MLDA} at receiver _{TCP,MLDA} (in kbit/s)	192.09 (64%)	326.94 (43%)
τ_{TCP} at receiver _{TCP,MLDA} (in kbit/s)	74.25 (25%)	349.29 (46%)
f	0.39	1.07

The situation is different when considering $\text{branch}_{\text{sat}}$. The fairness issue towards TCP has already been discussed, however receivers_{MLDA} only exploit 68% of the full available bandwidth of the satellite link (300kbit/s). In fact, the bandwidth share of

MLDA is nearly equal for both receiver_{MLDA,Hybla-270} and receiver_{MLDA} (they differ only by 10 kbit/s). We can explain this fact with the layer granularity of MLDA: MLDA traffic is transported using three different layers. The base layer size is shaped to adopt to the lowest measured possible transfer rate. While the enhancement layers need to be adjusted to reflect all other configurations. Simply, with two enhancement layers it is not possible to serve all different applications; in our case the satellite link is completely congested for a short time whenever it tries to join an enhancement layer. Increasing the number of layers will result in better utilization of different receiver capabilities [24]. However, this approach increases the complexity of the scheme, as the receivers need to synchronize more layers.

Additionally, MLDA multicast traffic exploits much more bandwidth at $\text{branch}_{\text{sat}}$ than the unicast Hybla-270 connection, resulting in an unusable value of $f=0.39$ (a test-run with TCP Reno shows an even worse value of $f=0.31$). This results again from the low granularity of the receiver layers — joining an additional layer at the satellite branch will result in a temporarily over-utilization of the link's bandwidth, as the layer sizes are very large.

D. Satellite Traffic with Higher Losses Evaluation

We further concentrate our attention only to the satellite link, albeit with a different error-rate of 10^{-6} to evaluate less favorable transmission conditions, see Table 3.

Table 3. Transfer Rate at $\text{BER} = 10^{-6}$ (in brackets the percentage of the available bandwidth utilized).

	Reno	Westwood	Hybla-25	Hybla-270
τ_{MLDA} at receiver _{MLDA} (in kbit/s)	127.71 (43%)	123.79 (42%)	51.08 (17%)	123.2 (41%)
τ_{MLDA} at receiver _{TCP,MLDA} (in kbit/s)	125.56 (42%)	125.01 (42%)	26.12 (9%)	116.61 (39%)
τ_{TCP} at receiver _{TCP,MLDA} (in kbit/s)	90.05 (30%)	101.18 (34%)	143.76 (48%)	106.57 (36%)
f	0.72	0.81	5.50	0.92

In comparison to the transfer rates at $\text{BER} = 10^{-8}$ (Table 1), we note a decrease in TCP-friendly bandwidth distribution at all Reno versions, albeit with Hybla-270 still being the fairest (with $f=0.92$). Additionally, Westwood's fairness values decreases less (from 0.85 to 0.81) than Reno (from 0.82 to 0.72) due to its specialization on lossy links. Thus, with higher error rates fairness between unicast and multicast traffic seems to be harder to achieve.

E. Improving Performance Through IAS

Looking again at Table 3, we can see a serious total throughput decrease at receivers_{MLDA} with $\text{BER} = 10^{-6}$. The total bandwidth utilized is roughly half compared to the same simulation with $\text{BER} = 10^{-8}$. Interestingly, the transfer rate is almost equal, no matter if concurrent TCP traffic occurs or not (See both τ_{MLDA} rows). This is due to MLDA's bandwidth calculation depending on the error loss rate. The error rates at receiver_{MLDA} and receiver_{MLDA,TCP} are roughly the same, that is the losses detected at receiver_{MLDA,TCP} detected due to congestion and BER are the same as the BER-losses detected at receiver_{MLDA}.

Overall, the drastic loss in throughput rate at higher BERs are due to the false declaration of link errors as congestion errors. To counter these effects we deploy the error-differentiating IAS (Section C). Figures of the simulation run with IAS enabled for MLDA traffic are given in Table 4.

Table 4. Transfer Rate at BER= 10^{-6} (IAS Enabled)

	Reno	Westwood	Hybla-270
τ_{MLDA} at receiver _{MLDA} (in kbit/s)	160.66 (53%)	163.99 (55%)	160.92 (54%)
τ_{MLDA} at receiver _{TCP,MLDA} (in kbit/s)	140.02 (50%)	148.42 (49%)	140.48 (47%)
τ_{TCP} at receiver _{TCP,MLDA} (in kbit/s)	82.36 (27%)	95.08 (32%)	95.23 (32%)
f	0.55	0.81	0.92
Increase of τ_{MLDA} at receiver _{MLDA} compared to the IAS-less simulation	26%	32%	31%

The effects of the IAS can be most vividly seen at the receivers_{MLDA} when comparing the same simulation at BER= 10^{-6} without the IAS (Table 3). Indeed, we can see an average increase in transfer rate ranging from 26% to 32%, depending on the TCP version. Using the IAS at BER= 10^{-8} has only a negligible effect, as only few errors due to BER occur.

With IAS enabled, certain errors are declared by the IAS as link failure errors, which accordingly aren't taken into consideration by MLDA's bandwidth adaption algorithm. However, IAS declares a rather high amount of link errors falsely as congestion errors [7]. As a result, these errors are still counted in MLDA's bandwidth adaption algorithm, hence only a maximum of 55% of the possible bandwidth is used by MLDA, even if no theoretically 90% should be possible in the MLDA-only connection. Further improvements of the IAS might thus gain an even higher transfer rate increase.

IV CONCLUSIONS AND FUTURE WORK

We have seen that the throughput rate of an MLDA-controlled flow can decrease significantly when utilized over high-error satellite links. This is due MLDA's behavior to falsely reduce its transfer rate when detecting link errors. However, to a certain degree this can be compensated with the help of an error differentiation scheme. We have shown this by implementing the Inter-Arrival Scheme [7]. Further research is needed to see if and how a less conservative error discrimination scheme than IAS could improve throughput rate.

Improving MLDA's performance during link-failure situations is thus a rather straightforward task. However, taking additionally TCP-friendliness over such lossy (satellite) links into account is a completely different task, as the characteristics of the TCP connection has to be respected. The three examined TCP versions (Reno, Westwood, Hybla) show deficiencies in one or another case. Both, Reno and Westwood, cannot adequately cope with a satellite's long delay link [15]. While Hybla successfully improves TCP throughput over such long delay links, it's especially designed for stand-alone connections and thus doesn't cope optimally with concurrent traffic. We've seen that through the modification of its RTT₀ value the fairness level can be increased, however, much more research is needed to see

if it is feasible to set this parameter dynamically depending on different network conditions and how to determine the right value for it.

Unless a dedicated TCP version for both concurrent traffic and satellite traffic is available, no multicast solution can be thoroughly TCP friendly. Regarding MLDA, it adopts reasonably well to a satellite distribution scenario and can increase content distribution capacity by adequately shaping traffic to different receiver needs.

BIBLIOGRAPHY

- [1] Widmer, J., Denda, R., and Mauve, M. "A Survey on TCP-Friendly Congestion Control", IEEE Network, May 2001
- [2] Matrawy, A., and Lambadaris, I., "A Survey of Congestion Control Schemes for Multicast Video Applications", IEEE Communications Surveys and Tutorials, Volume 5, No.2, 2003
- [3] Sisalem, D., Wolisz, A., "MLDA: A TCP-friendly Congestion Control Framework for Heterogeneous Multicast Environments", IWQoS 2000, Jun 2000
- [4] Allman, M., Paxson, V., Stevens, W., "TCP Congestion Control", IETF, RFC 2581, Apr 1999
- [5] Casetti, C., Gerla, M., Lee, S.S., Mascolo, S., and Sanadidi, M., "TCP with Faster Recovery", Proceedings of MILCOM 2000, Los Angeles, CA, Oct 2000
- [6] Caini, C., Firrincielli, R., "A New Transport Protocol Proposal for Internet via Satellite: the TCP Hybla", ASMS Conference, Frascati, Italy, Jul 2003
- [7] Biaz, N., and Vaidya, N., "Discriminating Congestion Losses from Wireless Losses Using Inter-Arrival Times at the Receiver", IEEE Symposium ASSET'99, Richardson, TX, USA, 1999
- [8] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V., "RTP: A Transport Protocol for Real-Time Applications", IETF, RFC 1889, Jan 1996
- [9] Ehler, S., "Performance Analysis of Reliable Multicast Protocols Over Heterogeneous Satellite / Terrestrial Networks Using Simulation", Diploma Thesis, Technical University Berlin, Telecommunication Group, Oct. 2004
- [10] Sisalem, D., and Wolisz, A., "LDA+: Enhanced Loss-Delay Based Adaptation Algorithm", ICME 2000, Aug 2000
- [11] Chiu, D., Jain, R., "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks". Computer Networks and ISDN Systems, vol. 17, pp. 1-14, Jun 1989.
- [12] Padhye, J., Firoiu, V., Towsley, D., and Kurose, J., "Modeling TCP Throughput: A Simple Model and its Empirical Validation", in ACM SIGCOMM '98, Vancouver, Oct 1998.
- [13] Bekar, H., Sisalem, D., Wolisz, A., "Comparison and Evaluation of TCP-friendly Multicast Congestion control Protocols", To be published, 2005
- [14] Postel, J. "Transmission Control Protocol", IETF, RFC 793, Sep 1981
- [15] Floyd, S., "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic", Computer Communications Review, Vol.21, No.5, pp. 30-47, Oct 1991
- [16] Mujica, V.E., Sisalem, D., Popescu-Zeletin, R., and Wolisz, A., "TCP-Friendly Congestion Control over Wireless Networks", Proc. of European Wireless 2004, Barcelona, Spain, Feb 2004
- [17] "The Network Simulator - ns", tool and development home page, <http://www.isi.edu/nsnam/ns/>
- [18] Luglio, M., Stepanek, J., Gerla, M. "TCP Performance Using Splitting Over the Satellite Link", 8th Ka Band Utilization Conference, Baveno, Italy, pp. 45-52, Sep 2002
- [19] Chetty, P. R. K., "Satellite Technology and Its Applications", TAB Books, USA, 1988
- [20] ASMS-Task Force Technical Group, "Satellite Mobile System Architectures", Approved Report, Version 3, Jul 2002
- [21] Rappaport, T.S., "Wireless Communications Principles & Practice", Prentice Hall, 1996
- [22] European Economic Interest Grouping (EEIG), "European Information Technology Observatory (EITO) 2002", Mar 2002, <http://www.eito.com>
- [23] Rubenstein, D., Kurose, J., Towsley, D., "The Impact of Multicast Layering on Network Fairness", SIGCOMM'99, Cambridge, USA, Aug. 1999
- [24] Rubenstein, D., Kurose, J., Towsley, D., "The Impact of Multicast Layering on Network Fairness", SIGCOMM'99, Cambridge, USA, Aug. 1999