A Delay Comparison of Reliable Multicast Protocols: Analysis and Simulation Results

Christian Maihöfer**, Reinhold Eberhardt

DaimlerChrysler Research And Technology,
Communication Technology (RIC/TC)
P.O. Box 2360, 89013 Ulm, Germany
{christian.maihoefer|reinhold.eberhardt}@daimlerchrysler.com

Abstract. We compare packet delivery delays from a probabilistic analysis of a flat and two tree-based reliable multicast protocols with simulation results of SRM, RMTP and TMTP variations. Analytical and simulation results show identical behavior with varying packet loss probabilities, number of receivers, and transmission rates. Even the absolute delays of the analytical and simulation results are almost identical which shows that carefully mathematical modelling can provide results of the same quality as simulation does.
A second focus of our work is to identify an optimal branching factor of the control tree established by tree-based reliable multicast protocols. Our analytical and simulation results show that a branching factor which is too small can significantly increase delay.

1 Introduction

The performance of protocols can be analyzed by mathematical analysis, simulation, or measurement either in the Internet or with the help of large testbeds. Measurement in the target environment shows the actual performance of a protocol and its available implementation, however, especially for multicast groups, measurements are difficult to perform since they usually require coordination of many multicast sites. This is why most multicast protocols are analyzed with mathematical analysis or with simulation.

Mathematical analysis requires a well understood and usually simplified model of the environment and the considered protocol. If this is successful, mathematical analysis allows to change parameters and monitor their effect on a protocol's behavior in a quite easy way. Another advantage of mathematical analysis is their usually complete publication, allowing other persons to check the correctness and the assumed system model.

With simulations, the environment and the considered protocols can become more complex and hence, more realistic. However, large simulation systems like NS-2 [1] allow to change a lot of parameters, possibly influencing the results to a great extend. Moreover, it is difficult the check the correctness of simulations, since the implementations are often extensive and in many cases, the simulation

** Part of this work was done during the author's stay at the University of Stuttgart, Institute of Parallel and Distributed High Performance Systems (IPVR)

code is not public. This is why we believe that both mathematical analysis and simulation are useful.

In this paper we compare analytical delay results of reliable multicast protocols with NS-2 simulation studies of SRM [2], RMTP [3] and TMTP [4] variations. The packet delivery delay is an important issue for multimedia applications. For example real time applications like interactive distributed simulations, distributed games, or the delivery of MPEG I-frames [5] benefit from guaranteed reliability and low delays. Besides analyzing the delay between sender and receiver we determine the round trip delay between sending a data packet and receiving the last corresponding control packet at the sender. The round trip delay determines the time after a data packet can be removed from memory and influences the sending rate and throughput if the sender uses a window based sending scheme [6]. Furthermore, knowledge about this delay is important to adjust the retransmission timeout at the sender.

For the analytical model as well as for the simulation model we tried to describe the reality as good as possible. This includes data packet and control packet loss, asynchronous local clocks, TTL scoping for retransmissions and spatial loss correlation. Our results show that analytical delays and simulation delays are almost identical and show identical behavior with varying number of receivers, transmission rates, and packet loss probabilities. Besides comparing analytical and simulation results, an important focus of our work was to determine an optimal branching factor of the control tree in tree-based reliable multicast protocols. The optimal branching factor depends on several parameters like packet loss probability, protocol class and whether average delivery delay, or round trip delay is of interest. We can conclude, though, that a tuned branching factor can significantly reduce delay.

The remainder of this paper is structured as follows. In the next section related work is discussed. In Section 3 we discuss the analyzed and simulated protocol classes. In Section 4 we introduce our assumed system model followed by an overview of the delay analysis. Numerical and simulation results are presented in Section 5. Finally, we conclude with a brief summary.

2 Background and Related Work

The first comparative delay analysis of sender- (ACK) and receiver-initiated (NACK) approaches was presented by Yamamoto et al. [7] and DeCleene [8]. Yamamoto et al. have analyzed the expected average delivery delay and showed that receiver-initiated protocols with NACK suppression provide best scalability. However, their analytical model for this class was simplified in assuming that all receivers are perfectly synchronized and thus only one NACK is sent back to the sender in case of packet loss. While the analysis in [7] is independent of the network topology, in [8] a delay analysis of generic ACK- and NACK-based protocols operating over star and linear topologies was presented. In [9] the effect of local recovery and retransmission of parity packets on bandwidth and delay of NACK-based protocols is examined. While the bandwidth analysis is made in detail, the delay analysis is rather brief and comparatively simple. For example, they do not consider queuing delay in detail and neglect feedback processing.

They concluded that local recovery techniques and parity packets outperform other approaches.

Our analysis is based on the delay analysis of Yamamoto et al. [7] and on the basic analytical work of Pingali et al. [10] and Levine et al. [11]. Pingali et al. have done the first comparative analysis of reliable multicast protocols. They have compared the processing requirements of flat protocol classes. Levine et al. have extended this work to the class of ring- and tree-based approaches.

To our knowledge, our work is the first comparative delay analysis of generic classes of tree-based reliable multicast protocols, which considers feedback traffic and queuing delays. Furthermore, this is the first work that compares analysis results with simulation results. In our previous work in [12] we have analyzed two tree-based reliable multicast protocols but without considering aggregated acknowledgements (see next section) and without comparing the analytical results with simulation studies.

3 Protocol Classification and Description

In this section we briefly describe and classify the reliable multicast protocols analyzed and simulated in this paper. A more detailed and more general description for some of these classes can be found in [10], [11] and [9].

The first considered protocol class is a receiver-initiated protocol. Receiver-initiated protocols return only negative acknowledgments (NACKs) from receivers to the sender instead of positive acknowledgements (ACKs). When a receiver detects an error, e.g. by a wrong checksum, a skip in the sequence number or a timeout while waiting for a data packet, a NACK is returned to the sender. A returned NACK asks for a packet retransmission from the sender. We assume that retransmissions are sent using multicast. Our considered protocol class uses multicast NACKs, which is known as NACK-avoidance scheme. A receiver that detects an error sends a multicast NACK provided that it has not already received a NACK for this data packet from another receiver. Thus, in optimum case, only one NACK is received by the sender for each lost data packet. In conformance with [11] we denote this protocol class as N2. An example for such a protocol is the scalable reliable multicasting protocol (SRM) [2].

The next two protocol classes are tree-based protocols. In tree-based protocols, the members of a multicast group are organized in a so-called control tree to overcome the well-known acknowledgment implosion problem of flat approaches, i.e. overwhelming of the sender by a large number of acknowledgments. Since acknowledgments are propagated along the edges of the control tree in a leaf-to-root direction, the implosion problem can be avoided by limiting the branching factor of a node and thus the number of acknowledgment packets.

The first considered scheme is denoted as H3 in conformance with [13]. For both tree-based protocol classes we assume that the initial sender is the root of the control tree and that the initial transmission is multicasted to the global group. Global group denotes the whole multicast group in contrast to a local group, which is described below. H3 uses ACKs sent by receivers to their parent in the control tree, called group leader, in order to indicate correctly received packets. Each group leader that is not the root node also sends an ACK to its

parent as soon as a data packet has been received. If a timeout for an ACK occurs at a group leader, a multicast retransmission is invoked for this local group. A local group encompasses a group leader and its directly attached children. Such a retransmission can be sent to a separate multicast address for this local group or sent to the global group address and limited in scope by the TTL value.

Additionally to normal ACKs, H3 uses aggregated ACKs, so called AACKs. In contrast to normal ACKs, they are sent to confirm the correct message delivery for a whole subhierarchy of the control tree. A group leader sends an AACK to its parent not before it has received an AACK from *each* child node. Lead nodes in the control tree send AACKs immediately after correct packet reception. AACKs are necessary to guarantee reliable delivery even in case of node failures, since group leaders remove a packet from memory and thus preventing retransmissions not before their complete subhierarchy in the control tree has acknowledged correct message reception (see [13] for more details). An example of a protocol similar to our definition of H3 but without AACKs is RMTP [3]. RMTP uses subtree multicasting to limit the retransmission scope, which is an extension to regular routers and protocol class H3.

The second tree-based scheme H4 is based on NACKs with NACK suppression similar to N2 and on AACKs. NACKs are used to start a retransmission. Aggregated ACKs are used to announce the receivers' state and allow group leaders to remove data from memory. TMTP [4] is an example for class H4 but without AACKs. For the protocol simulations of RMTP and TMTP we have implemented AACKs to allow a fair comparison between the mathematical analysis and the simulation.

4 Delay Analysis

In this section we present an overview of the delay analysis exemplified by protocol H3, since the analysis of all three protocols is quite lengthy. The other protocols are analyzed in a technical report [14].

4.1 System Model

We assume the following system model for our analytical evaluations. A single sender multicasts a data packet to a set of R identical receivers. With probability q_D the multicast packet is corrupted or lost during the transmission to a single receiver. With probability p_A for ACKs and p_{AA} for AACKs, a control packet is corrupted or lost. According to measurements in the MBone [15] we assume spatially dependent losses in our analysis.

Finally, we assume that nodes do not fail and that the network is not partitioned, i.e. retransmissions are eventually successful. All nodes work exclusively for the multicast protocol and no background load is considered.

4.2 Analytical Approach

Our goal is to determine the delays between the initial generation of a packet at the sender and correct reception at a receiver as well as the reception of the

last control packet at the sender. These delays are determined by the necessary processing times for a packet at the sender and receivers, transmission delays, timeout delays to wait for a data or control packet and finally the number of necessary transmissions for correct reception of data and control packets.

The processing time at a node is determined by the load of such a node, i.e. the processing of data and control packets. We first determine the rates for initial sending and arrival of packets. Arrival times are modelled as a poisson distribution, which results in exponentially distributed inter-arrival times. As we assume general distributed service times this queue type is defined as $M|G|1$ queue [16].

The number of necessary data packet transmissions M is determined by the packet loss probabilities q_D and p_A. M has already been obtained for the various protocol classes in our processing and bandwidth requirements analysis [13].

Given the average processing times and the number of transmissions we can determine the delay experienced by a single data packet.

4.3 Analysis

For a delay analysis of tree-based protocols we distinguish among sender, leaf node receivers and group leaders. Although the sender is a group leader as well, here and in the following we will denote only inner nodes as group leaders.

Mean Waiting Times at the Sender (Root Node) First, we have to determine the mean waiting time for a packet between generation and completion of processing or sending. The mean waiting time is determined by the load of a node, i.e. the processing of incoming and outgoing packet flows. The sender has to process the following five arriving packet flows:

1. Data packets from the higher protocol layer that are transmitted for the first time. This packet flow is referred to as λ_t^S and has an assumed rate of λ. The processing time for a data packet is assumed to be X.
2. Data packets that are retransmitted due to packet loss. This packet flow is referred to as λ_r^S and has rate $\lambda(E(M^{H3})-1)$, since every packet is on average $(E(M^{H3})-1)$-times retransmitted. $E(M^{H3})$ is the expected total number of transmissions per packet until all multicast group members have received it correctly.
3. ACK control packet flow λ_a^S received by the sender with rate $\lambda E(\widetilde{L}_a^{H3})$. The processing time for an ACK packet is assumed to be Y.
4. The AACK flow λ_{aa}^S with rate $\lambda E(\widetilde{L}_{aa}^{H3})$ and processing time Y.
5. Finally, the AACK query flow λ_q^S to ask for AACK retransmissions in case of AACK packet loss with rate $\lambda E(L_{aaq}^{H3})$ and processing time Y.

The expected total number of necessary transmissions $E(M^{H3})$ to receive the data packet correctly at all receivers, the number of received ACK packets $E(\widetilde{L}_a^{H3})$, and received AACK packets, $E(\widetilde{L}_{aa}^{H3})$, and finally the number of sent AACK query packets, $E(L_{aaq}^{H3})$, are given in [13] and not repeated here.

The load on the sender is given by the traffic intensity ϱ, which is generally the product of the traffic rate λ and mean processing time for a request (data transmission, retransmission or request) $E(S)$:

$$\varrho = \lambda E(S). \tag{1}$$

$$\varrho_S^{H3} = (\lambda_t^S + \lambda_r^S)E(X) + (\lambda_a^S + \lambda_{aa}^S + \lambda_q^S)E(Y). \tag{2}$$

As explained in Section 4.2, the system can be modeled as a $M|G|1$ queue. The Pollaczek-Chintchine formula gives the mean number of requests to be processed $E(L)$ [16]:

$$E(L) = \varrho + \frac{\varrho^2 + \lambda^2 Var(S)}{2(1-\varrho)}. \tag{3}$$

With the formula of Little [16]:

$$E(L) = \lambda E(T), \tag{4}$$

the mean waiting time of a request in the system $E(T)$ is (see Eq. 1, 3, 4):

$$E(T) = E(S) + \frac{\varrho^2 + \lambda^2 Var(S)}{2\lambda(1-\varrho)}. \tag{5}$$

The mean waiting time for a packet until processing starts is:

$$E(W) = E(T) - E(S) = \frac{\varrho^2 + \lambda^2 Var(S)}{2\lambda(1-\varrho)}. \tag{6}$$

With Eq. 1 and $Var(X) = E(X^2) - (E(X))^2$ [16]:

$$E(W) = \frac{\lambda E(S^2)}{2(1-\varrho)} \tag{7}$$

$$E(W_S^{H3}) = \frac{(\lambda_t^S + \lambda_r^S)E(X^2) + (\lambda_a^S + \lambda_{aa}^S + \lambda_q^S)E(Y^2)}{2(1-\varrho_S^{H3})}. \tag{8}$$

Mean Waiting Times at a Receiver (Leaf Node) A receiver has to process the following two flows. There is the data reception flow λ^R with rate $\lambda E(M^{H3})(1-q_D)$ which automatically triggers an ACK or AACK flow and therefore results in processing time of $X+Y$. The flow of AACK queries λ_q^R with rate $\lambda E(\widetilde{L}_{aaq}^{H3})$ triggers the replying of AACKs, which results in total processing time $Y+Y$. $E(\widetilde{L}_{aaq}^{H3})$ is given in [13]. The load on a receiver is then:

$$\varrho_R^{H3} = \lambda^R E(X+Y) + \lambda_q^R E(Y+Y). \tag{9}$$

The expectation of the waiting time at the receiver until processing starts is (see Eq. 7):

$$E(W_R^{H3}) = \frac{\lambda^R E((X+Y)^2) + \lambda_q^R E((Y+Y)^2)}{2(1-\varrho_R^{H3})}. \tag{10}$$

Mean Waiting Times at a Group Leader (Inner Node) The load on an inner node is the sum of the sender's load without the initial transmission and a receiver's load:

$$\varrho_G^{H3} = \lambda_r^S E(X) + (\lambda_a^S + \lambda_{aa}^S + \lambda_q^S)E(Y) + \lambda^R E(X+Y) + \lambda_q^R E(Y+Y). \quad (11)$$

The mean waiting time of a packet at an inner node follows to:

$$E(W_G^{H3}) = \frac{\lambda_r^S E(X^2) + \left(\lambda_a^S + \lambda_{aa}^S + \lambda_q^S\right)E(Y^2) + \lambda^R E((X+Y)^2) + \lambda_q^R E((Y+Y)^2)}{2(1-\varrho_G^{H3})}. \quad (12)$$

Overall Delay We assume that T is the group leader timeout delay, τ is the network propagation delay, h is the maximum and $\tilde{h}$ is the average number of hierarchy levels of the control tree and B is the branching factor of the control tree, i.e. the maximum number of child nodes per group leader. If no retransmission is necessary, the delay from the initial transmission $E(I)$ is:

$$E(I) = E(W_S^{H3}) + E(X) + \tau + E(W_G^{H3}) + E(X). \quad (13)$$

Note that a simplifying pessimistic assumption we made is that the receiver is always a group leader and therefore take $E(W_G^{H3})$ in the above equation.

Now we want to determine the delay for a hierarchical retransmission on condition that the parent node has received the packet correctly. The time for a hierarchical retransmission $E(H)$ is:

$$E(H) = \left(E(M_r^{H3}|M_r^{H3} > 1) - 1\right)\left(T + E(W_G^{H3}) + E(X)\right) + \tau_H + E(W_G^{H3}) + E(X). \quad (14)$$

M_r^{H3} is the number of necessary transmissions for a single receiver r until correct reception (M_r^{H3} is calculated [13]) and τ_H is the network propagation delay for a hierarchical retransmission. For obtaining the overall delay, we determine the probabilities that no data loss occurs, that a node misses a packet but the parent node is able to retransmit it, that a node and its parent misses that packet and the next parent retransmits it and so forth and multiply these probabilities with the expected delays. The overall delay is then:

$$E(S_\phi^{H3}) = \Big[\sum_{i=o}^{\tilde{h}-2} q_D^i(1-q_D)\left(E(I) + iE(H)\right)\Big] + q_D^{\tilde{h}-1}\left(E(W_S^{H3}) + E(X) + (\tilde{h}-1)E(H)\right). \quad (15)$$

Besides the delay for delivering data packets we examine the round trip delay of AACKs, i.e. the mean time between sending a packet and receiving the corresponding AACKs at the sender. Note that this delay is used to manage the sending window and release buffer space. Furthermore, it limits the throughput if a window based sending scheme is used [6]. Before a receiver sends an AACK it must receive the data packet before. The mean waiting time between sending a packet and receiving the last AACK at the sender is given by:

$$\begin{aligned} E(S_{RTD}^{H3}) = \\ &\underbrace{\left(E(M^{H3}) - 1\right)\left(T + E(W_G^{H3}) + E(X)\right)}_{\textit{sending transmissions}} + \underbrace{E(X) + E(W_G^{H3}) + \tau_H + E(W_G^{H3}) + E(X)}_{\textit{receiving last successful retransmission}} \\ &+ (h-1)\Big[\underbrace{E(\widetilde{L}_{aaq}^{H3})\left(T + E(W_G^{H3}) + E(Y)\right)}_{\textit{send AACK queries}} + \underbrace{E(Y) + \tau_H + E(W_G^{H3}) + E(Y)}_{\textit{send and receive successful AACK}}\Big]. \end{aligned} \quad (16)$$

5 Comparison of Analysis and Simulation Results

We have implemented the SRM [2], RMTP [3] and TMTP [4] reliable multicast protocols in the NS2 [1] network simulator environment to compare them with the analysis. Recall that SRM is a receiver-initiated protocol, RMTP is a tree-based sender-initiated protocol and TMTP is a tree-based receiver-initiated protocol with NACK suppression.

In contrast to the specification of RMTP we have implemented no subcast mechanism, as this is not available with general routers. Instead we used TTL-limited multicast to send retransmissions. A further significant difference is that we send acknowledgments as soon as a data packet is received rather than periodically. Besides normal ACKs we have additionally implemented aggregated ACKs. As a consequence of the aggregated ACKs, this protocol is of class H3.

In contrast to the specification of TMTP we have implemented AACKs rather than so-called early ACKs. TMTP uses early ACKs to advance the flow control window. An early ACK is sent after the corresponding data packet has been received. This means, a group leader does not need to wait for ACKs from all its children in order to send an early ACK to its parent. While this specification allows to loose data in case of node failures, we have implemented AACKs to cope with such situations. As a consequence of the NACK with NACK suppression and AACK scheme, this protocol is of class H4.

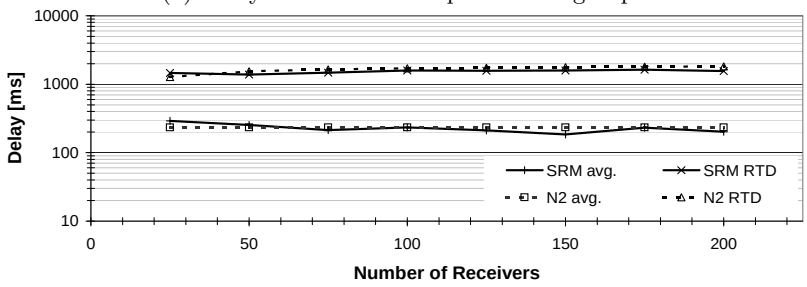
In conformance with the specifications we use a rate and window based sending scheme for flow and congestion control. TMTP defines a periodic interval at which each receiver unicasts an ACK (here AACK) to its parent and suggests to set it on the round trip time to the farthest receiver. In our analysis we have determined the round trip time between sending a data packet and receiving the last corresponding control packet at the sender under the assumption that the control packets are sent immediately. Therefore, our TMTP implementation sends AACKs immediately after receiving a data packet rather than periodically.

Note that our intention is to compare analytical with simulation results and to determine an optimal branching factor rather than to compare SRM, RMTP and TMTP, which makes only limited sense due to the modifications described above and due to the limited simulated group sizes.

For our simulations we have used two networks generated by Tiers [17] with 250 and 1000 nodes. All nodes in the network use DVMRP routing. Each link in the network is configured with probability 0.02 for packet loss. We have measured an average end-to-end packet loss probability for data packets of about 12% for the 250 node network and 15% for the 1000 node network. The average propagation delay was measured to be about 70ms for the 250 node network and 130ms for the 1000 node network. These measured values were used to configure the analysis parameters. While we have varied the sending rate for our simulations, the flow control window size was always 10.

Figures 1-3 show the results of our simulations in comparison with the numerical results of the analysis. The solid lines display the results from the simulation whereas the dotted lines display the numerical results from our analysis. The sending rate for results with varying group size is $1\frac{1}{s}$, the network consists of 250 nodes and the branching factor is 10. For results with varying sending rate, the group size is 100 receivers. For results with varying branching factors,

(a) Delay of SRM with respect to the group size



(b) Delay of SRM with respect to the sending rate

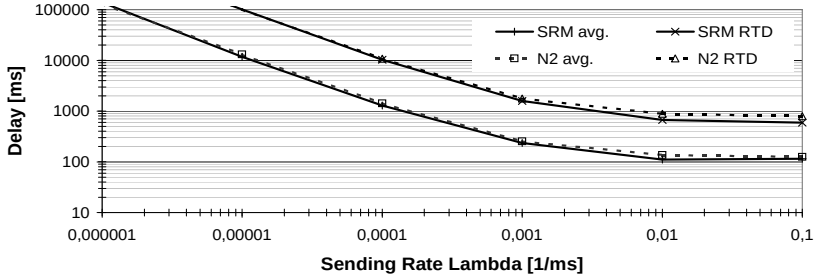


Fig. 1. Analytical vs. simulated delays of SRM

the network consists of 1000 nodes, the multicast group of 200 receivers and the sending rate is $1\frac{1}{s}$.

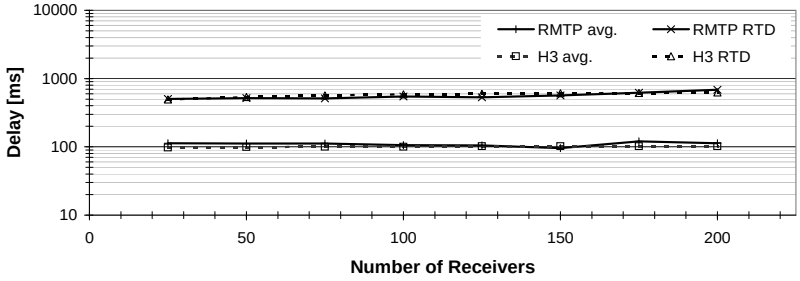
All results show that the average delivery delay is almost independent of the group size within the simulated range, even for the non-tree based SRM approach.[1] The round trip delay increases only moderately with increasing group size. Although the simulation results show some fluctuations caused by the random packet loss probabilities it is notable that even the absolute delays of the simulations are very close to the predicted results of the analysis.

With respect to varying sending rates we can see that while the average delivery delay and round trip delay of RMTP are independent of the sending rate, the results for SRM and TMTP show high delays with low sending rates. This is caused by the receiver initiated loss detection, i.e. a packet loss is detected not before the first packet with higher sequence number is successfully received. With low sending rates, this delay becomes rather high. This is why in most cases sender-initiated protocols like RMTP result in lower delays.

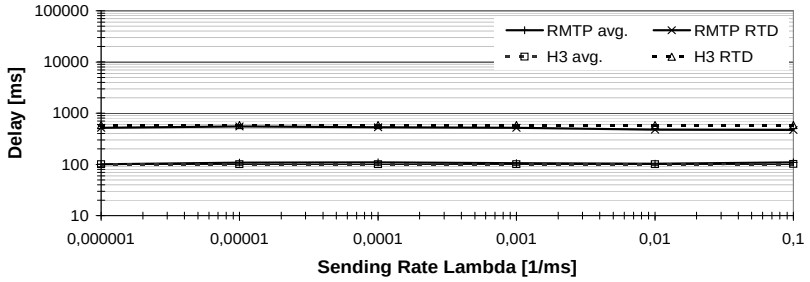
The last results in Figure 2.c and 3.c show the results with a varying branching factor. As predicted by the analytical results, average delay is hardly influenced by the branching factor. For the round trip delay, though, we see that both the analytical as well as simulation results for RMTP and TMTP show a decrease with increasing local group size due to the decreased height of the

[1] However, we know from analytical results that SRM is earlier saturated from feedback implosion and hence cannot support very large group sizes (e.g. see [12]).

(a) Delay of RMTP with respect to the group size



(b) Delay of RMTP with respect to the sending rate



(c) Delay of RMTP with respect to the branching factor

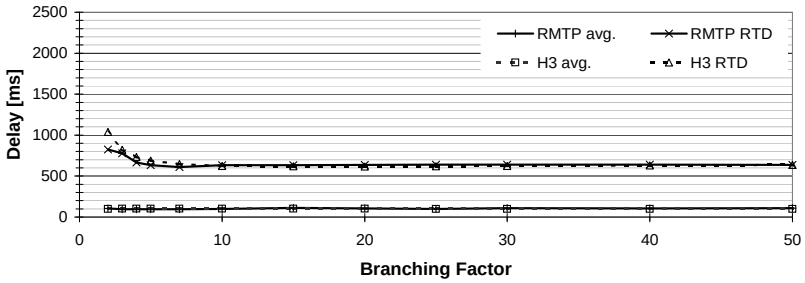


Fig. 2. Analytical vs. simulated delays of RMTP

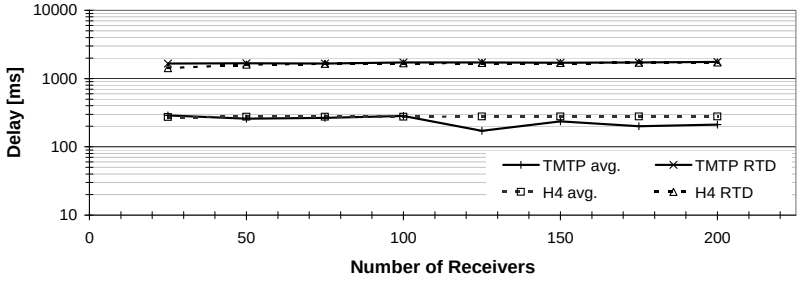
control tree. This means, a larger branching factor allow higher throughput if a window-based sending scheme is used.

In further simulation runs we have varied the packet loss probability. For example, we have decreased link loss probability to 0.002, resulting in an average end-to-end packet loss probability for data packets of about 1.5% in the simulated network, which has decreased packet delay of RMTP's analysis and simulation results by approximately 25%. More important than the absolute results, though, all protocols show again the same results for both analysis and simulation.

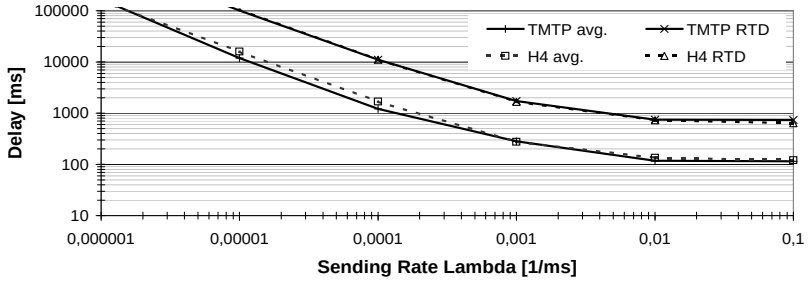
6 Summary

We have presented a probabilistic approach to analyze the average packet delivery delay as well as the round trip delay for reliable multicast protocols. A major

(a) Delay of TMTP with respect to the group size



(b) Delay of TMTP with respect to the sending rate



(c) Delay of TMTP with respect to the branching factor

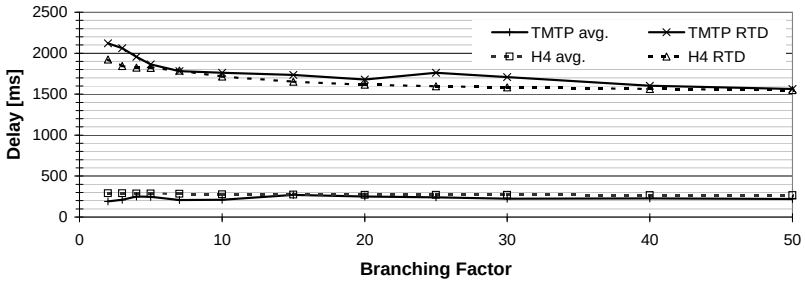


Fig. 3. Analytical vs. simulated delays of TMTP

focus of our work was to compare the gained analytical results with simulation results to prove their correctness. The comparison of a flat and two tree-based reliable multicast protocols has shown identical behavior of analysis and simulation results with varying packet loss probabilities, number of receivers, and transmission rates. Even the absolute delays of the analytical and simulation results are almost identical which shows that carefully mathematical modelling can provide results of the same quality as simulation does.

An important second focus of our work was to determine an optimal branching factor of the control tree established by tree-based reliable multicast protocols. The optimal branching factor depends on several parameters like protocol class and whether average delivery delay or round trip delay is of interest. We can conclude, though, that a branching factor which is too small can significantly increase round trip delay which limits a protocol's throughput.

References

1. Bajaj, S., Breslau, L., Estrin, D., Fall, K., Floyd, S., P. Haldar, M.H., Helmy, A., Heidemann, J., Huang, P., Kumar, S., McCanne, S., Rejaie, R., Sharma, P., Varadhan, K., Xu, Y., Yu, H., Zappala, D.: Improving simulation for network research. Technical Report 99-702, University of Southern California, USA (1999)
2. Floyd, S., Jacobson, V., Liu, C., McCanne, S., Zhang, L.: A reliable multicast framework for light-weight sessions and application level framing. IEEE/ACM Transactions on Networking **5** (1997) 784–803
3. Paul, S., Sabnani, K., Lin, J., Bhattacharyya, S.: Reliable multicast transport protocol (RMTP). IEEE Journal on Selected Areas in Communications, special issue on Network Support for Multipoint Communication **15** (1997) 407–421
4. Yavatkar, R., Griffioen, J., Sudan, M.: A reliable dissemination protocol for interactive collaborative applications. In: The Third ACM International Multimedia Conference and Exhibition (MULTIMEDIA '95), New York, USA, ACM Press (1995) 333–344
5. Pejhan, S., Schwartz, M., Anastassiou, D.: Error control using retransmission schemes in multicast transport protocols for real-time media. IEEE/ACM Transactions on Networking **4** (1996) 413–427
6. Allman, M., Glover, D., Sanchez, L.: Enhancing TCP over satellite channels using standard mechanisms. Internet Engineering Task Force, Network Working Group, Request for Comments, RFC 2488
   URL: ftp://ftp.isi.edu/in-notes/rfc2488.txt (1999)
7. Yamamoto, M., Kurose, J., Towsley, D., Ikeda, H.: A delay analysis of sender-initiated and receiver-initiated reliable multicast protocols. In: Proceedings of IEEE INFOCOM Conference on Computer Communications, Los Alamitos, IEEE Press (1997) 480–488
8. DeCleene, B.: Delay characteristics of generic reliable multicast protocols. Technical Report TR-08150-3, Logicon TASC (1996)
9. Nonnenmacher, J., Lacher, M., Jung, M., Carl, G., Biersack, E.: How bad is reliable multicast without local recovery. In: Proceedings of IEEE INFOCOM Conference on Computer Communications, New York, USA, IEEE Press (1998) 972–979
10. Pingali, S., Towsley, D., Kurose, J.F.: A comparison of sender-initiated and receiver-initiated reliable multicast protocols. In: Proceedings of the Sigmetrics Conference on Measurement and Modeling of Computer Systems, New York, USA, ACM Press (1994) 221–230
11. Levine, B., Garcia-Luna-Aceves, J.: A comparison of reliable multicast protocols. Multimedia Systems **6** (1998) 334–348
12. Maihöfer, C., Rothermel, K.: A delay analysis of tree-based reliable multicast protocols. In: Proceedings of the Tenth International Conference on Computer Communications and Networks, Scottsdale, USA, IEEE Press (2001) 274–281
13. Maihöfer, C., Rothermel, K.: Optimal branching factor for tree-based reliable multicast protocols. Computer Communications **25** (2002) 1018–1027
14. Maihöfer, C., Rothermel, K.: A delay analysis of tree-based reliable multicast protocols. Technical Report TR 2001/03, University of Stuttgart (2001)
15. Yajnik, M., Kurose, J., Towsley, D.: Packet loss correlation in the MBone multicast network. In: Proceedings of IEEE Global Internet, London, UK, IEEE Press (1996) 94–99
16. Kleinrock, L.: Queueing Systems, Volume II: Computer Applications. Wiley Interscience, New York, USA (1976)
17. Calvert, K., Doar, M., Zegura, E.: Modeling internet topology. IEEE Communications Magazine **35** (1997) 160–163