A Delay Analysis of Reliable Multicast Protocols

Christian Maihöfer and Kurt Rothermel
University of Stuttgart, Institute of Parallel and Distributed High Performance Systems (IPVR),
Breitwiesenstr. 20-22, D-70565 Stuttgart, Germany
{christian.maihoefer|kurt.rothermel}@informatik.uni-stuttgart.de

Keywords — reliable multicast, sender-initiated, receiver-initiated, SRM

Abstract — We present a delay analysis of one sender- and two receiver-initiated classes of reliable multicast protocols. Besides the average delivery delay we consider the delay to reliably deliver all packets and the round trip delay. The former two examines the delay between generation of a packet at the sender and correct reception at a randomly chosen receiver or all receivers, respectively. The latter is the delay between generation of a packet at the sender and reception of all acknowledgment packets at the sender.

Our numerical results show that receiver-initiated protocols provide significantly better scalability for large receiver groups and transmission rates compared to the sender-initiated protocol. However, the delay of the sender-initiated protocol within its scalability range is substantially lower compared to the receiver-initiated protocols.

To assess the quality of our analytical model we have compared the analytical results with a SRM-like protocol simulation. We show that the analytical and simulation results are strongly correlated, which demonstrates the appropriateness of our analytical model.

I. Introduction

If a reliable multicast protocol is used for real time applications, the resulting message delivery delay is an important issue. For example real time applications like interactive distributed simulations, distributed games, or the delivery of MPEG I-frames [1] benefit from guaranteed reliability. Besides time constraints of some applications, low delays are vital for providing high throughput with a window based sending scheme [2].

Known multicast protocols try to achieve reliable data delivery by forward error correction or retransmission schemes or by a combination of both approaches. As FEC alone cannot guarantee reliability we focus in this paper on protocols using a retransmission scheme. We distinguish among the three well-known protocol classes sender-initiated, receiver-initiated and receiver-initiated with feedback suppression.

The sender-initiated protocol class in denoted as (A1). Receivers return a positive acknowledgment (ACK) to the sender in order to indicate successful reception of the data packet. If the sender misses an ACK packet from one or more receivers, a multicast retransmission is sent to the whole group. An example for a sender-initiated protocol is the Xpress Transport Protocol (XTP) [3]. In contrast to sender-initiated protocols, receiver-initiated protocols return only negative acknowledgments (NAKs) instead of ACKs. As in the sender-initiated protocol class, we assume that retransmissions are sent using multicast. In protocols of class (N1), a receiver sends a unicast NAK to the sender after it has detected a missing or corrupted data packet. A missing packet can be detected by a gap in the sequence numbers. An example for (N1) is PGM [4]. Protocol (N2) uses multicast NAKs with NAK suppression. If an error is detected a NAK is sent to the whole group scheduled at a random time in the future provided that not already a NAK has been received for this data packet. Thus, in ideal case, only one NAK is received by the sender for each lost data packet. An example for such a protocol is the scalable reliable multicasting protocol (SRM) [5].

As the number of control messages is reduced with the use of negative acknowledgments and with the use of the NAK suppression scheme, scalability for a larger number of group members is improved, which has already been shown by previous work [6], [7], [8]. In this paper we present a delay analysis of the discussed protocol classes. In contrast to previous delay analysis we assume a more realistic system model as explained in Section II and we analyze several delay values. Besides average and maximum 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 if the sender uses a window based sending scheme [2]. Furthermore, knowledge about this delay is important to adjust timers at the sender and receivers. To assess the analytical results we have implemented a protocol of class (N2) in the network simulator NS-2 environment and compare the analytical results with simulation results.

Analogous to previous work, our results show that in terms of scalability (N2) is superior to (N1) and (N1) is superior to (A1). However, we have obtained more accurate absolute delays, which indicate that within its scalability range, (A1) provides substantially lower delays compared to receiver-initiated protocols. This results from the faster loss detection of sender-initiated protocols by missing ACK packets at the sender. Recall that receiver-initiated protocols detect a missing packet by a gap in the sequence number, which can result in a rather long delay. By analyzing the maximum delay and round trip delay additionally to the average delivery delay, we can provide a more accurate characterization of the considered protocol classes, which is especially important for applications with time bounds for the delivery delay.

The remainder of this paper is structured as follows. In the next section related work is discussed. In Section III we introduce our assumed system model followed by the detailed delay analysis. Numerical results are presented in Section IV and compared with simulation results in Section V. Finally, we will conclude with a brief summary.

II. Background and Related Work

The first comparative analysis of reliable multicast protocols was done by Pingali et al. [9]. They have compared the processing re-

quirements of the same classes of sender- and receiver-initiated protocols we will use in our analysis. Levine et al. [10] have extended this work to the class of ring- and tree-based approaches and showed that tree-based approaches are superior in terms of scalability. In [7] a more realistic system model including loss of control packets was analyzed and further protocol classes were introduced. Besides processing requirements, bandwidth efficiency was subject to several analytical studies for example by Kasera et al. [11], Nonnenmacher et al. [12] and Maihöfer et al. [8].

Regarding delay analysis, the first comparative delay analysis of sender- and receiver-initiated approaches was presented by Yamamoto et al. [6] and DeCleene [13]. Yamamoto et al. have analyzed the expected average delivery delay and showed that receiver-initiated protocols with NAK 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 NAK is sent back to the sender in case of message loss. While the analysis in [6] is independent of the network topology, in [13] a delay analysis of generic ACK- and NAK-based protocols operating over star and linear topologies was presented. In [12] the effect of local recovery and retransmission of parity packets on bandwidth and delay of NAK-based protocols is examined. While the bandwidth analysis is made in detail, the delay analysis is rather brief. They concluded that local recovery techniques and parity packets outperforms other approaches.

Our paper follows the work in [6] but extends previous work in four significant ways. First, we consider the loss of control packets rather than assuming reliable delivery. Second, we assume that local clocks are not synchronized. Both assumption especially affects the NAK suppression scheme. Third, besides average delivery delay we examine the threshold delay and the round trip delay. Threshold delay is the delay to reliably deliver all packets with a certain probability. For applications with time bounds for the delivery of messages, threshold delay should be considered rather than average delay. In most cases threshold delay gives a more realistic impression of the delay behaviour of reliable multicast protocols. For example, for low packet loss probabilities and within the scalability range, the average delivery delays of all classes are rather similar and only moderately higher than the message propagation delay of the network, which means they are rather similar to unreliable protocols without retransmissions. In contrast to average delay, threshold delay allows to compare the protocols and the performance of their retransmission schemes in more detail. Round trip delay examines the delay until all receivers have acknowledged correct reception to the sender. The round trip delay determines when to remove packets from the sender's buffer space. Furthermore, it may limit the throughput of a protocol if a window-based sending scheme is used, since the round trip delay determines the delay to advance the window [2]. Note that according to our definition only protocol class (A1) provides such a signaling. Nevertheless, we will examine the expected round trip delay if an additional signaling scheme would be applied to (N1) and (N2), which is necessary if a window-based sending scheme is used. In such a case, usually periodic positive acknowledgments are used. Even without an explicit signaling, the expected delays can be rather helpful to appropriately configure the delay to delete data packets at the sender if it cannot keep all data packets for possible retransmissions in memory. Finally, we have implemented a SRM-like protocol [5] similar to protocol class (N2) in the network simulation environment NS2 [14] and compared the results.

III. Analysis

A. System Model

We assume the following system model for our analytical evaluations. A single sender multicasts a message to a set of R identical receivers. With probability q_D the multicast message is corrupted or lost during the transmission to a single receiver. With probability p_A for ACKs, p_N for unicast NAKs and q_N for multicast NAKs, a control message is corrupted or lost. We assume that nodes do not fail and that the network is not partitioned, i.e. retransmissions are finally successful. All nodes work exclusively for the multicast transport protocols and no background load is considered.

B. Analytical Approach

Our goal is to determine the delays between the initial generation of a packet at the sender and the 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 modeled 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 [15].

The number of necessary data packet transmissions M is determined by the packet loss probabilities q_D, p_A, p_N and q_N. M has already been determined for the various protocol classes in our processing and bandwidth requirements analysis [7], [8].

Given the average processing times and the number of transmissions we can determine the delay experienced by a single data packet. A summary of the frequently used notations is given in Table 1.

C. Sender-initiated Protocol (A1)

C.1 Mean Waiting Times at the Sender

As a first step we have to determine the mean waiting time for a packet between generation or arrival and completion of processing or sending. The waiting time is determined by the load on the sender; the load is given by the packet flows processed by the sender under the considered protocol. The sender of protocol (A1) has to process the following three arriving packet flows:

1. Data packets that are transmitted for the first time. This packet flow is referred to as λ_t^S and has rate λ. The processing time for a data packet is X.
2. Data packets that are retransmitted due to packet loss. This packet flow is referred to as λ_r^S and has rate $\lambda(E(M^{A1})-1)$, since every packet is $(E(M^{A1})-1)$-times retransmitted. $E(M^{A1})$ is the total number of necessary transmissions to correctly receive a data packet at all receivers.
3. Control packets that are received by the sender with flow λ_a^S and rate $\lambda R E(M^{A1})(1-q_D)(1-p_A)$. R is the number of receivers. Each receiver returns an ACK for every transmission provided that no data packet loss occurs with probability $(1-q_D)$. The sender receives an ACK packet provided that no ACK loss occurs with probability $(1-p_A)$. The processing time for an ACK packet is Y.

Table 1
FREQUENTLY USED NOTATIONS FOR THE ANALYSIS

R	Size of the receiver set.
W_S^w, W_R^w	Waiting time for the sender or receiver. $w \in \{A1, N1, N2\}$
λ	Sending rate for data packets.
λ_t^S	Initial transmission flow from the sender.
λ_a^S, λ_n^S	ACK or NAK packet flow received at the sender.
λ_r^S	Retransmission flow at the sender.
λ_t^R	Data packet reception flow at the receiver.
λ_n^R	NAK flow from a receiver.
$\lambda_{n,r}^R$	NAK flow received at a receiver.
ϱ_S^w, ϱ_R^w	Total load on the sender or receiver.
T_S, T_R	Sender or receiver timeout delay.
τ	Packet propagation delay in the network.
S_ϕ^w, S_γ^w	Mean time between the initial arriving of a data packet at the sender and the correct reception at a random receiver or at all receivers with probability γ.
S_{RTD}^w	Mean time between the initial arriving of a data packet at the sender and the correct reception of all control packets at the sender.
X	Processing time for data packets.
Y	Processing time for control packets.
q_D	Probability for multicast data loss at a receiver.
p_A, p_N, q_N	Probability for unicast ACK, NAK or multicast NAK loss.
M^w, M_r^w	Total number of necessary transmissions to receive a data packet correctly at all receivers or at a random receiver r in the presence of data and ACK or NAK loss.
O^w, O_r^w	Number of necessary rounds to correctly deliver a packet to all receivers or to receiver r.
$O_e^w, O_{e,r}^w$	Total number of empty rounds or empty rounds for receiver r, respectively.

The expected total number of necessary transmissions $E(M^{A1})$ to receive the data packet correctly at all receivers is given in [7]:

$$E(M^{A1}) = \sum_{i=1}^{R} \binom{R}{i} (-1)^{i+1} \frac{1}{1-\tilde{p}^i}, \tag{1}$$

with probability for a retransmission $\tilde{p}$:

$$\tilde{p} = q_D + (1-q_D)p_A, \tag{2}$$

i. e. either a data packet is lost (q_D) or the data packet is received correctly and the ACK is lost $((1-q_D)p_A)$.

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) $E(S)$ [15]:

$$\varrho = \lambda E(S). \tag{3}$$

The load on the sender (ϱ_S^{A1}, traffic intensity) is then the sum of the packet rates:

$$\varrho_S^{A1} = \lambda E(M^{A1})E(X) + \lambda R E(M^{A1})(1-q_D)(1-p_A)E(Y). \tag{4}$$

As explained in Section III-B the system can be modeled as a $M|G|1$ queue. With the help of the Pollaczek-Chintchine formula and the formula of Little the mean waiting time for a packet until processing starts is [15]:

$$E(W) = \frac{\lambda E(S^2)}{2(1-\varrho)}, \tag{5}$$

where S is the processing time for a request. Given this formula, the waiting time for protocol (A1) at the sender is:

$$E(W_S^{A1}) = \frac{(\lambda_r^S + \lambda_t^S)E(X^2) + \lambda_a^S E(Y^2)}{2(1-\varrho_S^{A1})}. \tag{6}$$

C.2 Mean Waiting Times at a Receiver

The only packet flow at the receiver is the reception of data packets, which are acknowledged by an ACK, λ_t^R, with rate $\lambda E(M^{A1})(1-q_D)$. The processing time is $X+Y$ since the arrival of a packet is followed by replying an ACK packet to the sender. Note that X and Y are independent random variables. The load on the receiver is:

$$\varrho_R^{A1} = \lambda E(M^{A1})(1-q_D)\Big(E(X)+E(Y)\Big). \tag{7}$$

The mean waiting time of a packet at the receiver until processing starts is (see Eq. 5):

$$E(W_R^{A1}) = \frac{\lambda_t^R E\Big((X+Y)^2\Big)}{2(1-\varrho_R^{A1})}. \tag{8}$$

With X and Y are independent random variables, $E(V+W) = E(V)+E(W)$, $Var(V) = E(V^2) - (E(V))^2$ and $Var(V+W) = Var(V)+Var(W)$:

$$E(W_R^{A1}) = \frac{\lambda_t^R \Big(E(X^2)+E(Y^2)+2E(X)E(Y)\Big)}{2(1-\varrho_R^{A1})}. \tag{9}$$

C.3 Overall Delay of Protocol (A1)

Now that we have analyzed the mean waiting times at the sender and receiver we are able to determine the delay of a specific data packet. This delay includes the waiting time after arrival or generation of a data packet at the sender, the time needed to send a packet, the network propagation delay, the time needed to receive a packet at a receiver and the waiting time for processing.

T_S is assumed to be the constant sender timeout delay after which a missing ACK is assumed to be lost or the corresponding data packet, respectively, and a retransmission is invoked. τ is assumed to be the constant network propagation delay for a packet. The probability that a receiver needs j transmissions until correct reception of a data packet is $P(M_r = j) = q_D^{j-1}(1-q_D)$. M_r is the necessary number of transmissions to correctly receive a data packet at a random receiver r. $E(S_\phi^{A1})$ is assumed to be the mean time between the initial arrival or generation of a data packet at the sender and the correct reception at a receiver. Yamamoto et al. [6] have shown that $E(S_\phi^{A1})$ equals:

$$E(S_\phi^{A1}) = \Big[\sum_{j=1}^{\infty} q_D^{j-1}(1-q_D)\Big(j(E(W_S^{A1})+E(X)) + (j-1)T_S\Big)\Big] + \tau + E(W_R^{A1}) + E(X). \tag{10}$$

By taking the expectation of the geometric distribution, this can be simplified to:

$$E(S_\phi^{A1}) = \frac{E(W_S^{A1}) + E(X) + q_D T_S}{1-q_D} + \tau + E(W_R^{A1}) + E(X). \tag{11}$$

With $E(M_r) = \frac{1}{1-q_D}$ and $E(M_r) - 1 = \frac{q_D}{1-q_D}$ [7], where $E(M_r)$ is the number of necessary transmissions for a single receiver r:

$$E(S_\phi^{A1}) = \underbrace{E(M_r)\Big(E(W_S^{A1})+E(X)\Big) + \Big(E(M_r)-1\Big)T_S}_{\text{time at the sender}} + \underbrace{\tau}_{\text{transport}} + \underbrace{E(W_R^{A1})+E(X)}_{\text{time at a receiver}}. \tag{12}$$

Besides the average delivery delay we can determine the expected maximum delay to reliably deliver all data packets with a certain probability, denoted as γ. We will call this delay threshold delay $E(S_\gamma^{A1})$. With $E(M)$ denoting the number of transmissions, $\gamma = 1 - {q_D}^{E(M)}$ holds. Now, $E(S_\gamma^{A1})$ can be obtained as follows:

$$E(M) = \frac{ln(1-\gamma)}{ln(q_D)}, \gamma \geq 1 - q_D \tag{13}$$

$$E(S_\gamma^{A1}) = E(M)\Big(E(W_S^{A1}) + E(X)\Big) + \Big(E(M)-1\Big)T_S + \tau + E(W_R^{A1}) + E(X). \tag{14}$$

Finally, we want to examine the delay for receiving all ACK packets at the sender. This delay is important since in many cases a window based sending scheme is used and upon reception of all ACKs the corresponding data packet can be deleted and the sending window advanced. We obtain the round trip delay $E(S_{RTD}^{A1})$ as follows:

$$E(S_{RTD}^{A1}) = \underbrace{\Big(E(M^{A1})-1\Big)\Big(T_S + E(W_S^{A1}) + E(X)\Big)}_{\text{sending transmissions}} + \underbrace{E(X) + E(W_S^{A1}) + \tau + E(W_R^{A1}) + E(X)}_{\text{receiving last successful retransmission}} + \underbrace{E(Y) + E(W_R^{A1}) + \tau + E(W_S^{A1}) + E(Y)}_{\text{send and receive last ACK}}. \tag{15}$$

D. Receiver-initiated Protocol (N1)

D.1 Mean Waiting Times at a Sender

Analogous to protocol (A1), three packet flows are to be processed at the sender. Packet flow λ_t^S for the first transmission with rate λ and processing time X. Then the NAK flow λ_r^S with rate $\lambda(E(M^{N1})-1)$ that triggers a retransmission and therefore has processing time $X+Y$. Finally, λ_n^S with rate $\lambda(\vartheta_1(1-p_N) - (E(M^{N1})-1))$ of additional NAKs, which are processed but trigger no retransmission and therefore have the processing time Y.

ϑ_1 is the total number of NAKs sent in all rounds and is determined in the processing requirements analysis [7] as follows:

$$\vartheta_1 = \sum_{k=1}^{E(M^{N1})} N_k \frac{1}{1-p_k}. \tag{16}$$

N_k, the number of NAKs sent in round k is given by:

$$N_k = {q_D}^k R, \tag{17}$$

where ${q_D}^k$ is the probability for a single receiver that until round k all data packets are lost. p_k is the probability that all sent NAKs in round k are lost:

$$p_k = {p_N}^{N_k} = {p_N}^{{q_D}^k R}. \tag{18}$$

The number of transmissions is (see Eq. 1):

$$E(M^{N1}) = \sum_{i=1}^{R} \binom{R}{i} (-1)^{i+1} \frac{1}{1-{q_D}^i}. \tag{19}$$

Given these flows, the load on the sender is:

$$\varrho_S^{N1} = \lambda E(X) + \lambda\Big(E(M^{N1})-1\Big)E(X+Y) + \lambda\Big[\vartheta_1(1-p_N) - \Big(E(M^{N1})-1\Big)\Big]E(Y) \tag{20}$$

$$= \lambda E(M^{N1})E(X) + \lambda\vartheta_1(1-p_N)E(Y). \tag{21}$$

The mean waiting time for a packet at the sender until it is processed is:

$$E(W_S^{N1}) = \frac{\lambda E(S^2)}{2(1-\varrho_S^{N1})} \tag{22}$$

$$= \frac{\lambda_t^S E(X^2) + \lambda_r^S\Big(E(X^2) + E(Y^2) + 2E(X)E(Y)\Big) + \lambda_n^S E(Y^2)}{2(1-\varrho_S^{N1})}. \tag{23}$$

D.2 Mean Waiting Times at a Receiver

A receiver detects a missing packet by a gap in the sequence numbers, i.e. when it receives packet $i+1$ without having received i. As a consequence of this error detection mechanism, the delay for detecting an error is influenced by the data packet sending rate λ. Since NAKs are also subject to packet losses, one or more NAKs must be sent to trigger a retransmission. After sending a NAK a timer is started at the receiver. If the timer expires, a new NAK is sent and the timer is restarted until the data packet is correctly transmitted.

At the receiver we have the following two packet flows. Flow λ_t^R of data packets received from the sender with rate $\lambda E(M^{N1})(1-q_D)$ and processing time X. And the flow λ_n^R of generated NAKs with rate $\lambda\Big(E(O_r^{N1})-1\Big)$ and processing time Y. O_r^{N1} is the number of necessary transmissions for a single receiver plus the number of empty rounds in which no retransmission is triggered due to NAK loss (see [7]):

$$E(O_r^{N1}) = E(M_r^{N1}) + E(O_{r,e}^{N1}) \tag{24}$$

$$E(O_{r,e}^{N1}) = \sum_{k=1}^{E(M_r^{N1})-1}\Big(\frac{1}{1-p_k} - 1\Big) \tag{25}$$

$$E(M_r^{N1}) = \frac{1}{1-q_D}. \tag{26}$$

Given these flows, the load on a receiver is:

$$\varrho_R^{N1} = \lambda E(M^{N1})(1-q_D)E(X) + \lambda P(O_r^{N1}>1)\Big(E(O_r^{N1}|O_r^{N1}>1) - 1\Big)E(Y). \tag{27}$$

Now, the mean waiting time for a packet at a receiver is:

$$E(W_R^{N1}) = \frac{\lambda_t^R E(X^2) + \lambda_n^R E(Y^2)}{2(1-\varrho_R^{N1})}. \tag{28}$$

D.3 Overall Delay of Protocol (N1)

After the sender has transmitted a data packet it collects all NAKs within a certain timeout period and sends a retransmission. We assume the timeout period to be long enough for all NAKs to be received.

To obtain the overall delay, Yamamoto et al. [6] distinguish between the following phases:

1. Loss Detection Phase. This phase encompasses the time between the initial arrival of a packet i at the sender and the triggering of a NAK at one of the receivers, which have lost the first data packet. The loss is detected with the arrival of a packet j, where $j > i$.
2. Loss Recovery Phase. This phase encompasses the time between the end of the first phase and the correct reception of a packet at the considered receiver. As data packets and NAKs can be lost, this phase includes the periodical sending of NAKs until the data packet is received correctly.

D.4 Loss Detection Phase

Here we must consider the time to unsuccessfully send data packets, the time to send and receive the first successful data packet and the time to send an initial NAK for the first lost data packet. The random variable L is the number of consecutive lost packets at the $k+1$ unsuccessful receivers. Given that $K=k$, the conditional probability distribution of L is:

$$P(L=l|K=k) = q_D^{l(k+1)}(1-q_D^{k+1}), \quad l = 0,1,\ldots \text{ and } k = 0,1,\ldots,R-1. \tag{29}$$

The number of subsequent lost packets at $k+1$ receiver follows from the expectation of the geometric distribution:

$$E(L|K=k) = \frac{q_D^{k+1}}{1-q_D^{k+1}}. \tag{30}$$

To obtain the mean k among the possible ones between 0 and $R-1$ we have:

$$E(L|K) = \sum_{k=0}^{R-1} \binom{R-1}{k} q_D^k (1-q_D)^{R-1-k} \frac{q_D^{k+1}}{1-q_D^{k+1}}. \quad (31)$$

Now we multiply the mean number of subsequent lost packets with the time $\frac{1}{\lambda}$, to process a packet. The delay of the $L+1$st packet is:

$$\underbrace{E(W_S^{N1}) + E(X)}_{\text{sender processing delay}} + \underbrace{\tau}_{\text{propagation delay}} + \underbrace{E(W_R^{N1}) + E(X).}_{\text{receiver processing delay}} \quad (32)$$

Finally, the first phase can be expressed as follows:

$$E(D^{N1}) = \sum_{k=0}^{R-1} \binom{R-1}{k} q_D^k (1-q_D)^{R-1-k} \frac{q_D^{k+1}}{1-q_D^{k+1}} \frac{1}{\lambda} + E(W_S^{N1}) + E(W_R^{N1}) + \tau + 2E(X) + E(Y). \quad (33)$$

D.5 Loss Recovery Phase

From the viewpoint of a random receiver, this phase encompasses a number of timeout rounds. This means, the initial sent NAK in $E(D^{N1})$ was unsuccessful. The following receiver timeouts have the length $T_R + E(W_R^{N1}) + E(Y)$, where T_R is the timeout period, W_R^{N1} the waiting time and Y the processing time for a NAK. After a number of unsuccessful sent NAKs, this round ends with a final successful sent NAK to the sender. This includes the propagation delay to the sender, the sending of the data packet, the propagation delay to the receiver and receiver processing of the received data packet. The mean loss recovery delay under the assumption that NAK packets are not lost is:

$$\begin{aligned} E(R^{N1}) &= \sum_{j=1}^{\infty} q_D^{j-1}(1-q_D)(j-1)\Big(T_R + E(W_R^{N1}) + E(Y)\Big) \\ &\quad + 2\Big(E(X)+\tau\Big) + E(W_S^{N1}) + E(W_R^{N1}) + E(Y) \quad (34) \\ &= \frac{q_D}{1-q_D}\Big(T_R + E(W_R^{N1}) + E(Y)\Big) + 2\Big(E(X)+\tau\Big) \\ &\quad + E(W_S^{N1}) + E(W_R^{N1}) + E(Y) \quad (35) \\ &= \Big(E(M_r^{N1}) - 1\Big)\Big(T_R + E(W_R^{N1}) + E(Y)\Big) + 2\Big(E(X)+\tau\Big) \\ &\quad + E(W_S^{N1}) + E(W_R^{N1}) + E(Y). \quad (36) \end{aligned}$$

Note that j is the number of timeout rounds due to data packet or NAK loss. If we consider the loss of NAK packets, only one case, the loss of all sent NAK packets increases the recovery delay. The total retransmission rounds due to data packet loss and loss of all NAK packets is denoted as O_r^{N1} (see Eq. 24). Therefore, we change the above equation to:

$$\begin{aligned} E(R^{N1}) &= \Big(E(O_r^{N1}) - 1\Big)\Big(T_R + E(W_R^{N1}) + E(Y)\Big) + 2\Big(E(X)+\tau\Big) \\ &\quad + E(W_S^{N1}) + E(W_R^{N1}) + E(Y). \quad (37) \end{aligned}$$

The overall delay of protocol (N1) consists of both phases:

$$\begin{aligned} E(S_\phi^{N1}) &= (1-q_D)\Big(E(W_S^{N1}) + E(W_R^{N1}) + 2E(X) + \tau\Big) \\ &\quad + q_D\Big(E(D^{N1}) + E(R^{N1})\Big). \quad (38) \end{aligned}$$

The delay to reliably deliver a certain percentage γ of data packets is denoted by $E(S_\gamma^{N1})$. It can be obtained with a modified loss recovery phase as follows:

$$\begin{aligned} E(R^{N1}) &= \Big(E(M_\gamma^{N1}) + E(O_{e,r}^{N1}) - 1\Big) * \Big(T_R + E(W_R^{N1}) + E(Y)\Big) \\ &\quad + \tau + E(W_S^{N1}) + E(Y) \\ &\quad + E(X) + \tau + E(W_R^{N1}) + E(X) \quad (39) \\ E(S_\gamma^{N1}) &= E(D^{N1}) + E(R^{N1}). \quad (40) \end{aligned}$$

$E(O_{e,r}^{N1})$ is determined similar to Eq. 25 with $E(M_\gamma^{N1})$ instead of $E(M_r^{N1})$. Besides the delivery delay we want to determine the delay for removing the data packet at the sender and to advance the sending window. Although protocols (N1) and (N2) provide no signaling for the correct reception of a data packet at all receivers, we will examine the expected round trip delay if an additional signaling scheme would be applied, which is necessary if a window-based sending scheme is used (see discussion in Section II).

$$\begin{aligned} E(R^{N1}) &= \Big(E(O^{N1}) - 1\Big)\Big(T_R + E(W_R^{N1}) + E(Y)\Big) + 2\Big(E(X)+\tau\Big) \\ &\quad + E(W_S^{N1}) + E(W_R^{N1}) + E(Y) \quad (41) \\ E(S_{RTD}^{N1}) &= (1-q_D)^R\Big(E(W_S^{N1}) + E(W_R^{N1}) + 2E(X) + \tau\Big) \\ &\quad + \Big(1-(1-q_D)^R\Big)\Big(E(D^{N1}) + E(R^{N1})\Big) \\ &\quad + E(Y) + \tau + E(W_S^{N1}) + E(Y). \quad (42) \end{aligned}$$

E. Receiver-initiated Protocol (N2)

Protocol (N2) varies protocol (N1) with a NAK suppression mechanism trying to reduce the number of NAKs. Analogous to protocol (N1), we assume that the sender maintains a timer to collect all NAKs belonging to one round and therefore prevent multiple retransmissions for the same lost data packet.

E.1 Mean Waiting Times at a Sender

At the sender we distinguish among the following three packet flows: First, the flow for the initial data packet transmission λ_t^S with rate λ and processing time X. Second, the NAK flow that trigger a retransmission λ_r^S with rate $\lambda\Big(E(M^{N2})-1\Big)$ and processing time $X+Y$. Finally, the third flow of additional NAKs, which are not necessary to trigger a retransmission λ_n^S with rate $\lambda\vartheta_1(1-q_N) - \lambda_r^S$ and processing time Y.

The number of generated NAKs is:

$$\vartheta_1 = \sum_{k=1}^{E(M^{N2})} N_k \frac{1}{1-p_k}. \quad (43)$$

p_k is the probability that all sent NAKs in round k are lost:

$$p_k = q_N^{N_k}. \quad (44)$$

N_k, the number of NAKs sent in round k, is obtained as follows. The first receiver that did not receive the data packet sends a NAK. The probability for packet loss in round k is q_D^k, which is equal to $N_{k,1}$, the probability for the first receiver to send a NAK. Then a second receiver sends a NAK provided that it has received no data packet and no NAK packet. Either the first receiver has sent no NAK (with probability $1-N_{k,1}$) or the NAK was lost or sent simultaneously (with probability $N_{k,1}(q_N + p_s - q_N p_s)$). As we assume a system model in which local clocks are not synchronized, it is possible that NAKs are sent simultaneously. This probability is given by p_s. Now, N_k can be expressed as follows:

$$\begin{aligned} N_k &= \sum_{i=1}^{R} N_{k,i} \quad (45) \\ N_{k,1} &= {q_D}^k \quad (46) \\ N_{k,2} &= {q_D}^k(1 - N_{k,1} + N_{k,1}(q_N + p_s - q_N p_s)) \\ &= N_{k,1} - N_{k,1}^2 + N_{k,1}^2(q_N + p_s - q_N p_s) \quad (47) \\ N_{k,n} &= N_{k,n-1} - N_{k,n-1}^2 + N_{k,n-1}^2(q_N + p_s - q_N p_s), n > 1. \quad (48) \end{aligned}$$

The number of transmissions is (see Eq. 1):

$$E(M^{N2}) = \sum_{i=1}^{R} \binom{R}{i} (-1)^{i+1} \frac{1}{1-{q_D}^i}. \quad (49)$$

Given these flows, the load on the sender is:

$$\varrho_S^{N2} = \lambda E(M^{N2})E(X) + \lambda\vartheta_1(1-q_N)E(Y). \quad (50)$$

The mean waiting time of a packet at the sender until it is processed is:

$$E(W_S^{N2}) = \frac{\lambda E(S^2)}{2(1-\varrho_S^{N2})} \quad (51)$$

$$= \frac{\lambda_t^S E(X^2) + \lambda_r^S\Big(E(X^2)+E(Y^2)+2E(X)E(Y)\Big) + \lambda_n^S E(Y^2)}{2(1-\varrho_S^{N2})} \quad (52)$$

E.2 Mean Waiting Times at a Receiver

At the receiver we distinguish among three packet flows. λ_t^R is the flow of data packets from the sender with rate $\lambda E(M^{N2})(1-q_D)$ and processing time X. The second flow λ_n^R consists of the submitted NAK packets with rate $\lambda\Big(E(O_r^{N2})-1\Big)\frac{\vartheta_2}{\vartheta_3}$ and processing time Y. The probability for sending a NAK is according to the throughput analysis $\frac{\vartheta_2}{\vartheta_3}$. ϑ_2 is the average number of NAKs sent in each round and ϑ_3 is the mean number of receivers that did not receive a data packet and therefore want to send a NAK. The third flow $\lambda_{n,r}^R$ are the received NAKs from other receivers with rate $\lambda(1-q_N)\Big[\vartheta_2\Big(E(O^{N2})-1\Big) - \frac{\vartheta_2}{\vartheta_3}\Big(E(O_r^{N2})-1\Big)\Big]$ and processing time Y.

The number of rounds O^{N2} is the sum of the number of necessary rounds for sending transmissions M^{N2} and the number of empty rounds in which all NAKs are lost and therefore no retransmission is made:

$$O_r^{N2} = M_r^{N2} + O_{e,r}^{N2} \quad (53)$$

$$O^{N2} = M^{N2} + O_e^{N2} \quad (54)$$

$$E(O_e^{N2}) = \sum_{k=1}^{E(M^{N2})-1}\Big(\tfrac{1}{1-p_k}-1\Big) \quad (55)$$

$$E(O_{e,r}^{N2}) = \sum_{k=1}^{E(M_r^{N2})-1}\Big(\tfrac{1}{1-p_k}-1\Big). \quad (56)$$

O_r^{N2} and M_r^{N2} are the corresponding numbers for a single receiver.

The number of necessary transmissions, M_r^{N2}, for a single receiver r is given by the probability q_D. Analogous to Eq. 26 of protocol (N1) the expectation is:

$$E(M_r^{N2}) = \frac{1}{1-q_D}. \quad (57)$$

ϑ_2 is the average number of NAKs sent in each round and ϑ_3 is the mean number of receivers that did not receive a data packet and therefore want to send a NAK:

$$\vartheta_2 = \frac{1}{E(O^{N2})}\sum_{k=1}^{E(M^{N2})} N_k\tfrac{1}{1-p_k} \quad (58)$$

$$\vartheta_3 = \frac{1}{E(O^{N2})}\sum_{k=1}^{E(M^{N2})} q_D^k R\tfrac{1}{1-p_k}, \quad (59)$$

where $(1/1-p_k)$ is the number of empty rounds plus the last successful NAK sending (see Eq. 56 and 26).

With these flows, the load on the receiver is:

$$\varrho_R^{N2} = \lambda E(M^{N2})(1-q_D)E(X) + (\lambda_n^R + \lambda_{n,r}^R)E(Y). \quad (60)$$

Therefore, the mean waiting time of a packet at a receiver is:

$$E(W_R^{N2}) = \frac{\lambda_t^R E(X^2) + (\lambda_n^R + \lambda_{n,r}^R)E(Y^2)}{2(1-\varrho_R^{N2})}. \quad (61)$$

Analogous to protocol (N1) we distinguish between the loss detection and loss recovery phase to analyze the overall delay.

E.3 Loss Detection Phase

This phase is similar to (N1) but additionally considers a random delay $E(B^{N2})$. This delay starts with the discovery of packet loss at the first receiver. It ends with the expiration of the backoff timer and the transmission of the initial NAK. The loss detection phase is given as follows:

$$E(D^{N2}) = \underbrace{\sum_{k=0}^{R-1}\binom{R-1}{k} q_D^k (1-q_D)^{R-1-k}\frac{q_D^{k+1}}{1-q_D^{k+1}}\frac{1}{\lambda}}_{\text{mean number of subsequent lost packets}}$$

$$+ \underbrace{E(W_S^{N2}) + E(W_R^{N2}) + \tau + 2E(X)}_{\text{delay of the first received packet}} + E(B^{N2}) + E(Y). \quad (62)$$

E.4 Loss Recovery Phase

The loss recovery phase is similar to (N1):

$$E(R^{N2}) = \Big(E(O_r^{N2})-1\Big)\Big(T_R + E(W_R^{N2}) + E(B^{N2}) + E(Y)\Big) + 2\Big(E(X)+\tau\Big) + E(Y) + E(W_S^{N2}) + E(W_R^{N2}). \quad (63)$$

E.5 Overall Delay of Protocol (N2)

The overall delay of protocol (N2) is analogous to protocol (N1):

$$E(S_\phi^{N2}) = (1-q_D)\Big(E(W_S^{N2}) + E(W_R^{N2}) + 2E(X) + \tau\Big) + q_D\Big(E(D^{N2}) + E(R^{N2})\Big). \quad (64)$$

The delay to reliably deliver a certain percentage γ of data packets is denoted by $E(S_\gamma^{H2})$. It can be obtained as follows:

$$E(R^{N2}) = \Big(E(M_\gamma^{N2}) + E(O_{e,r}^{N2}) - 1\Big) * \Big(T_R + E(W_R^{N2}) + E(B^{N2}) + E(Y)\Big) + \tau + E(W_S^{N2}) + E(Y) + E(X) + \tau + E(W_R^{N2}) + E(X) \quad (65)$$

$$E(S_\gamma^{N2}) = E(D^{N2}) + E(R^{N2}). \quad (66)$$

Analogous to protocol (N1) we determine the round trip delay as:

$$E(R^{N2}) = \Big(E(O^{N2})-1\Big)\Big(T_R + E(W_R^{N2}) + E(B^{N2}) + E(Y)\Big) + 2\Big(E(X)+\tau\Big) + E(Y) + E(W_S^{N2}) + E(W_R^{N2}) \quad (67)$$

$$E(S_{RTD}^{N2}) = (1-q_D)^R\Big(E(W_S^{N2}) + E(W_R^{N2}) + 2E(X) + \tau\Big) + \Big(1-(1-q_D)^R\Big)\Big(E(D^{N2}) + E(R^{N2})\Big) + E(Y) + \tau + E(W_S^{N2}) + E(Y). \quad (68)$$

IV. Numerical Results

We examine the expected delays of the analyzed protocols by means of some numerical examples. The chosen delays are according to measurements in [16] $X = 500\mu s$ for data packets and $Y = 100\mu s$ for control packets. Analogous to [6], the packet processing times are assumed as constant with no variability, i.e. according to $Var(X) = E(X^2) - (E(X))^2 = 0$, the second moments are determined as $E(X^2) = (E(X))^2$. The propagation delay is chosen as $\tau = 10ms$. The timeouts are chosen as the doubled propagation delay, i.e. $T_S = T_R = 20ms$. For the NAK suppression time we have assumed $B = 30ms$. A discussion of reasonable values for the NAK suppression time can be found in [6].

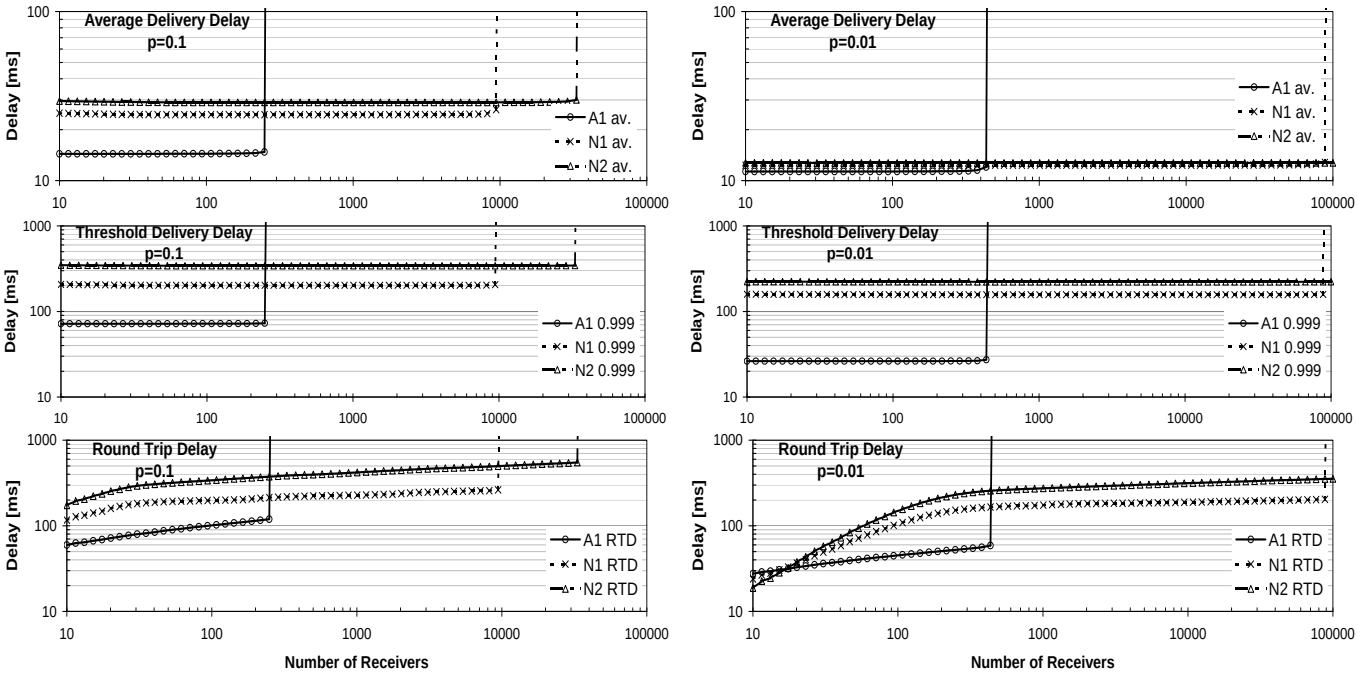

Fig. 1. Delays with respect to the number of receivers

Figure 1 plots the expected average delay, the threshold delay to reliably deliver all packets with probability 0.999 and the round trip delay for all protocol classes with varying number of receivers. Data and control packet loss probability is 0.1 (left side) and 0.01 (right side). The data rate is $\lambda = 0.01 \frac{1}{ms}$. Figure 2 plots the delays for 1000 receivers with varying sending rate λ.

As it can be seen, protocol (A1) provides poor scalability. If the number of receivers or the sending rate exceeds a certain limit, protocol (A1) becomes quickly overwhelmed with feedback control messages. For example, with sending rate $\lambda = 0.01 \frac{1}{ms}$ and loss probability 0.1, protocol (A1) can support only up to 250 receivers. Protocol (N1) as well as (N2) provide significantly better scalability.

Although (N1) and (N2) provide better scalability, their average delivery delays as well as threshold and round trip delays are higher within their scalability range. This results from the receiver-initiated loss detection. Recall that receiver-initiated protocols detect packet loss by a gap in the sequence number, i.e. not before a subsequent packet is correctly received. For low sending rates this loss detection delay is the dominant delay. The varying sending rate in Figure 2 shows this in more detail. For example, with sending rate $\lambda = 0.0001 \frac{1}{ms}$ it takes about 10s to detect packet loss. You can see in Figure 2 that the threshold delay and the round trip delay are about 10s. With respect to the average delivery delay, only nodes that have lost a packet are affected by the loss detection delay. Therefore, for loss probability 0.1 only 10% of all nodes need to wait 10s for detecting a packet loss; all other nodes receive the packet from the initial transmission. Therefore, the average delay is about 1s. With packet loss probability 0.01, the average delay is also decreased by about the factor 10.

The average delivery delay of all protocol classes within their scalability range and with low loss probability is close to the propagation delay of $\tau = 10ms$, since most nodes need no retransmissions. Compared to the average delivery delay, the threshold delay to deliver all messages with probability 0.999 is significantly higher. For applications having a time constraint to deliver all messages, threshold delay may be more important than average delivery delay. Analogous to average delivery delay, protocol (A1) has a significantly lower threshold delay within its scalability range, while protocols (N1) and (N2) provide better scalability for large groups or high transmission rates. With probability 1 for reliably deliver all packets, the threshold delay of our analysis would be infinite for all protocol classes, since there exists a low but non-zero probability that an infinite number of retransmissions is necessary. We will show in Section V that the threshold delay for $\gamma = 0.999$ is a good approximation for the delay to deliver all packets correctly.

The round trip delay for protocol (A1) shows the expected delay until all ACKs are received. After this time, the sender can remove the data packet. Besides freeing buffer space, the round trip delay is important if a window based sending scheme for flow and congestion control is used. In this case the round trip delay may limit the throughput, since throughput is basically given by $\frac{bufferspace}{rtt}$ [2]. As shown in Figure 1 the round trip delay increases with the group size. As protocols (N1) and (N2) provide no signaling for correct reception of data packets, the displayed round trip delay shows the expected delay if for example periodic positive acknowledgments would be used. Such periodic ACKs are used for example by SRM [5]. Otherwise, if no periodic ACKs are used, the expected round trip delay may be at least useful as a clue to configure the timeout delay for removal of data packets.

V. Comparison with Simulation Results

To assess the analytical results we have implemented a SRM like [5] reliable multicast protocol in the NS2 [14] network simulator environment. Recall that SRM is a receiver-initiated protocol with NAK

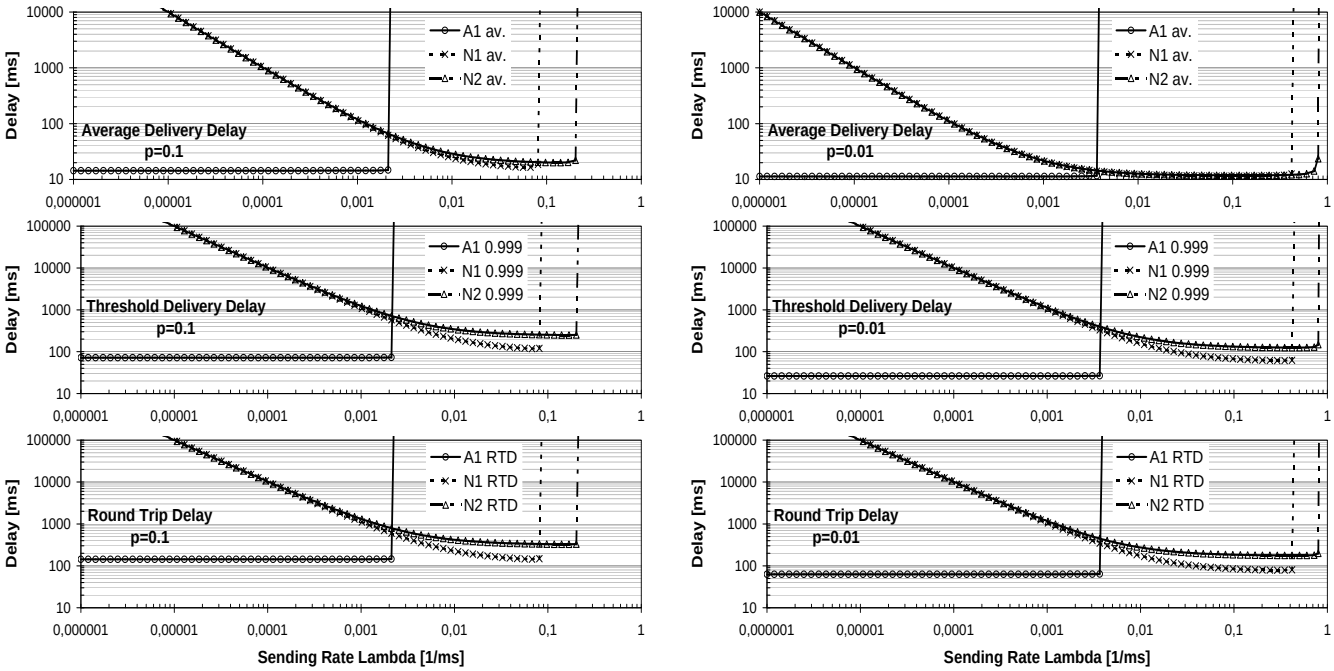

Fig. 2. Delays with respect to the sending rate

suppression, hence it belongs to class (N2). Besides the NAK suppression scheme our protocol use a rate and window based sending scheme for flow and congestion control. This requires the use of additional positive ACKs to advance the sending window. We have used ACKs only for controlling the sending window and not for triggering retransmissions which is completely NAK-based. While receivers in our implementation send an ACK after every correct data packet reception, a real world implementation would do this rather periodically.

While we have varied the sending rate, the window size was always 5. For our simulations we have used two networks generated by Tiers [17] with 250 and 1000 nodes. All nodes in the network use DVMRP [18] routing. To simulate message loss, each link in the network is configured with probability 0.02% or 0.002% respectively for message loss. We have measured an average end-to-end message loss probability for data packets of about 12% or 1% respectively. The average propagation delay was measured to be about 70ms for the 250 node network and 130ms for the 1000 node network. Though it would be very interesting, we were not able to simulate the saturation of SRM and the resulting delay explosion. The reason is that the saturation follows from the overwhelming CPU requirements but NS2 is a network simulator rather than a CPU simulator.

Figure 3 shows the average delay, round trip delay and threshold delay for varying sending rates and varying number of receivers. If the sending rate varies, the number of receivers is 100. If the number of receivers varies, the sending rate is $0.001\frac{1}{ms}$. By comparing the average delay, we can see that the analysis predicts very exactly the simulation results. In most cases, the deviation is less than 10%.

The analytical results for the threshold delay are made with probability 0.999 for correct delivery of all packets. The measured simulation results show the delay to correctly deliver all packets. We can see that the predicted results for threshold probability 0.999 are close to the measured results. In fact, for loss probability 0.01 they are almost identical with a deviation less than 5%. For loss probability 0.12% the measured results are up to $\lambda = 0.0005\frac{1}{ms}$ about two times of the predicted results and between $\lambda = 0.003\frac{1}{ms}$ and $0.1\frac{1}{ms}$ only half of the predicted results. Note that some of this deviation results from the window based sending scheme. If no further data packets can be sent due to missing ACKs of previous sent packets, the loss detection of the last packet sent is also delayed. If the average delay is measured, the results are only moderately affected by this behaviour since most packets are received from the initial transmission. However, for the threshold delay we measure the maximum delay which is affected significantly.

The results for the round trip delay are similar to the threshold delay results. Again, maximum delays are measured here which are more affected by the window based sending scheme or fluctuating message loss probabilities. Thus, the measured and analytical results show a larger deviation compared to the average delays.

The last figure shows the delay results for a varying number of receivers. Here we can see that within the scalability range, the average delivery delay is indeed almost independent of the group size. Although there is again a deviation in the absolute results for the round trip delay, the predicted and measured results show a similar delay increase with increased group size.

We can conclude from the results that the measured average delivery delay from our simulation studies is very appropriately predicted by our analytical model. For the round trip delay and threshold delay there are more significant deviations. They may result from fluctuating message loss probabilities in the simulation as well as from the window based sending scheme which introduces additional delays. However, in all cases the behaviour of the analytical model and the simulation studies with varying number of receivers, varying loss probabilities and varying sending rates are closely correlated.

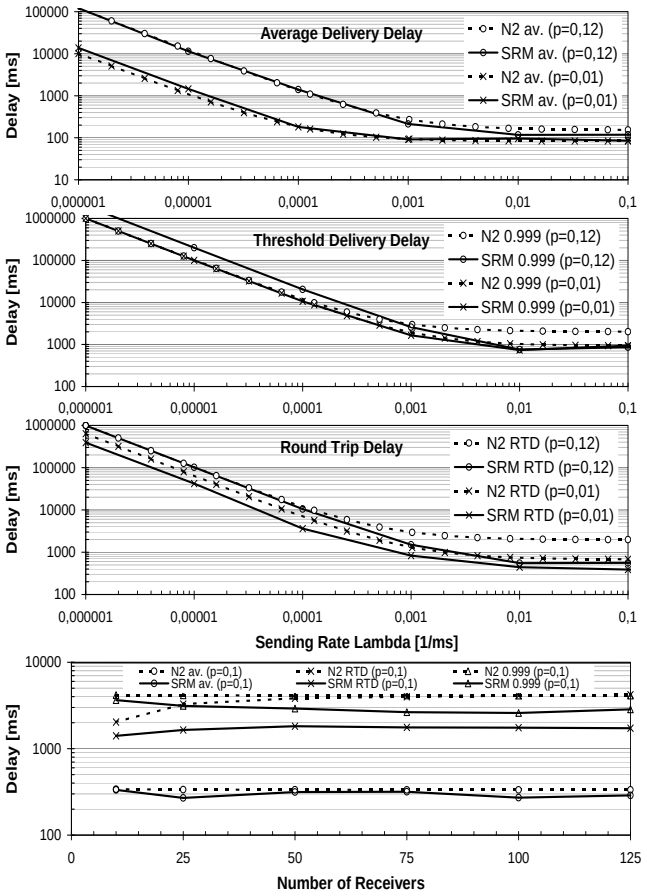

Fig. 3. Analytical versus Simulation Results

VI. SUMMARY

We have presented a delay analysis of a sender-initiated, a receiver-initiated and a receiver-initiated reliable multicast protocol class with NAK suppression. Besides the average delivery delay we have considered the delay to reliably deliver all packets and the round trip delay.

Our numerical results showed that receiver-initiated protocols provide significantly better scalability for large receiver groups and transmission rates compared to the sender-initiated protocol. However, the delay of the sender-initiated protocol within its scalability range is substantially lower compared to the receiver-initiated protocols and therefore possibly more appropriate for delay sensitive applications.

To assess the quality of our analytical model we have compared the analytical results with a SRM-like protocol simulation. Both show identical behaviour with varying number of receivers, transmission rates or loss probabilities. In case of average delivery delay, even the absolute delays of the analytical and simulation results are almost identical which shows that our analytical model is appropriate.

REFERENCES

[1] S. Pejhan, M. Schwartz, and D. Anastassiou, "Error control using retransmission schemes in multicast transport protocols for real-time media," *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 413–427, 1996.

[2] M. Allman, D. Glover, and L. Sanchez, "Enhancing TCP over satellite channels using standard mechanisms," RFC 2488, 1999.

[3] T. W. Strayer, B. J. Dempsey, and A. C. Weaver, *XTP – The Xpress transfer protocol*, Addison-Wesley Publishing Company, 1992.

[4] T. Speakman, D. Farinacci, S. Lin, and A. Tweedly, "PGM reliable transport protocol specification," Internet Engineering Task Force, Network Working Group, Internet Draft <draft-speakman-pgm-spec-04.txt>, work in progress, 2000.

[5] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang, "A reliable multicast framework for light-weight sessions and application level framing," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 784–803, Dec. 1997.

[6] M. Yamamoto, JF. Kurose, DF. Towsley, and H. Ikeda, "A delay analysis of sender-initiated and receiver-initiated reliable multicast protocols," in *Proceedings of IEEE INFOCOM'97*, Los Alamitos, Apr. 1997, pp. 480–488, IEEE.

[7] C. Maihöfer, K. Rothermel, and N. Mantei, "A throughput analysis of reliable multicast transport protocols," in *Proceedings of the Ninth International Conference on Computer Communications and Networks*, Las Vegas, Oct. 2000, pp. 250–257, IEEE.

[8] C. Maihöfer, "A bandwidth analysis of reliable multicast transport protocols," in *Proceedings of the Second International Workshop on Networked Group Communication (NGC 2000)*, Palo Alto, Nov. 2000, pp. 15–26, ACM.

[9] S. Pingali, D. Towsley, and J. F. Kurose, "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, May 1994, pp. 221–230, ACM Press.

[10] B. Levine and J. Garcia-Luna-Aceves, "A comparison of reliable multicast protocols," *Multimedia Systems*, vol. 6, no. 5, pp. 334–348, Sept. 1998.

[11] S. Kasera, J. Kurose, and D. Towsley, "A comparison of server-based and receiver-based local recovery approaches for scalable reliable multicast," in *Proceedings of IEEE INFOCOM'98*, New York, Apr. 1998, pp. 988–995, IEEE.

[12] J. Nonnenmacher, M. Lacher, M. Jung, G. Carl, and E. Biersack, "How bad is reliable multicast without local recovery," in *Proceedings of IEEE INFOCOM'98*, New York, Apr. 1998, pp. 972–979, IEEE.

[13] B. DeCleene, "Delay characteristics of generic reliable multicast protocols," Technical Report TR-08150-3, TASC, 1996.

[14] S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, M. Handley P. Haldar, A. Helmy, J. Heidemann, P. Huang, S. Kumar, S. McCanne, R. Rejaie, P. Sharma, K. Varadhan, Y. Xu, H. Yu, and D. Zappala, "Improving simulation for network research," Technical Report 99-702, University of Southern California, 1999.

[15] Leonard Kleinrock, *Queueing Systems, Volume II: Computer Applications*, Wiley Interscience, New York, 1976.

[16] S. Kasera, J. Kurose, and D. Towsley, "Scalable reliable multicast using multiple multicast groups," in *Proceedings of ACM SIGMETRICS*, Seattle, jun 1997, pp. 64–74, ACM.

[17] K. Calvert, M.B. Doar, and E.W. Zegura, "Modeling internet topology," *IEEE Communications Magazine*, vol. 35, no. 6, pp. 160–163, June 1997.

[18] D. Waitzman, C. Partridge, and S. Deering, "Distance vector multicast routing protocol," RFC 1075, 1988.