A Throughput Analysis of Reliable Multicast Transport Protocols

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

Abstract—**Tree-based reliable multicast protocols are known to provide better scalability than the protocols based on pure sender- and receiver-initiated schemes. However, previous analytical work that has provided these results is based on a system model which assumes reliable control message delivery and synchronized local clocks. These assumptions are questionable simplifications, since they favor protocols using multicasted negative acknowledgments with NAK avoidance scheme. In this paper, we extend previous analysis by taking into account control data loss and asynchronous local clocks.**

We further analyze a new protocol class with particular importance, the tree-based approach with aggregated acknowledgments. In contrast to other approaches, this class provides reliability not only in case of message loss but also in case of node failures. Our results show that the additional overhead to cope with node failures is very low and therefore acceptable for reliable multicast implementations.

I. Introduction

In recent years, a number of reliable multicast transport protocols have been proposed, which are based on the acknowledgment scheme. Reliability is ensured by replying acknowledgment messages from the receivers to the sender, either to confirm correct data packet delivery or to ask for a retransmission. Most reliable multicast protocols can be classified into sender-initiated, receiver-initiated and tree-based ones. Briefly characterized, in sender-initiated approaches receivers reply positive acknowledgments (ACKs) to confirm correct message delivery in contrast to receiver-initiated protocols, which indicate transmission errors or losses by negative acknowledgments (NAKs). Both classes can result in the well-known acknowledgment implosion problem, i.e. the overwhelming of the sender and the network around the sender by a large number of ACK or NAK messages, resulting in scalability problems. Tree-based approaches promise to be scalable even for a large number of members, since they arrange receivers into a hierarchy, called ACK tree. The ACK tree is created by techniques like expanding ring search or the token repository service [1, 2]. During normal operation, each receiver is responsible for collecting ACKs or NAKs only from their direct child nodes in the hierarchy. The maximum number of child nodes can be determined according to the processing performance of a node, its available network bandwidth, its memory equipment, and its reliability. Thus, no node is overwhelmed with messages and scalability for a large receiver group is ensured.

Sender-initiated, receiver-initiated and tree-based protocols have already been compared in previous analytical work. Pingali et al. [3] have shown that receiver-initiated protocols provide better scalability than sender-initiated ones and Levine et al. [4] have shown that a special class of tree-based protocols is the most scalable one. Our paper extends this previous work by considering loss of acknowledgment messages and analyzing a new class of tree-based protocols with aggregated acknowledgments. This class has particular importance, since its mechanisms are necessary to provide reliability even in the presence of node failures. Other classes of tree-based protocols analyzed so far provide reliability only in case of communication failures, since a node failure can lead to the loss of messages for this node's entire subhierarchy. We will explain this in more detail in the following section.

The remainder of this paper is structured as follows. In Section 2 we discuss the background of our throughput analysis and take a look at related work. In Section 3 we briefly classify the analyzed protocols. Our throughput evaluation in Section 4 starts with a definition of the assumed system model before the various protocol classes are analyzed in detail. To illustrate the results, some numerical evaluations are presented in Section 5 before we conclude with a brief summary.

II. Background and Related Work

Reliable multicast protocols were already analyzed in previous work. In contrast to most of this work, we analyze not a specific protocol but general protocol classes. Moreover, in this paper we focus on processing requirements of the protocol classes. The first work in this area was presented by Pingali et al. [3]. They have compared the general class of sender- and receiver-initiated protocols. Levine et al. [4] have extended this analysis to the class of ring- and tree-based approaches. Our paper is based on this previous work and extends it in three ways. First, we consider the loss of control packets rather than assuming reliable delivery. As already pointed out by Levine et al. [4], the assumption of reliable acknowledgment delivery in his own and previous work especially favors protocols that multicast acknowledgments. These protocols are based on a NAK-avoidance scheme which works most efficient if no NAKs are lost at receivers and therefore only one NAK per lost data packet is sufficient. Second, we assume that local clocks are not synchronized. This is also important for the NAK-avoidance scheme, which works less efficient with this more realistic assumption. Since a receiver does not know whether a NAK from another receiver has already been sent and will be received later, additional NAKs will be caused. Third, our work extends previous analysis by a new tree-based protocol class. It is based on aggregated ACKs to cope with node failures and a combined unicast/multicast retransmission scheme. The number of retransmission requests is compared with a threshold parameter that determines whether unicast or multicast is used for retransmissions. In the next section, these protocols are classified and described in more detail.

III. Classification of Reliable Multicast Protocols

In this section we want to briefly classify the reliable multicast protocols analyzed in this paper. A more detailed and more general description can be found in [3] and [4].

A. Sender-Initiated Protocols

The class of sender-initiated protocols is characterized by positive acknowledgments (ACKs) returned by the receivers to the sender. A missing ACK detects either a lost data packet at the corresponding receiver, a lost ACK packet or a crashed receiver, which cannot be distinguished by the sender. Therefore, a missing ACK packet leads to a data packet retransmission from the sender. We assume that such a retransmission is always sent using multicast. This protocol class will be referred to as (A1). Note that the use of negative acknowledgments, for example to speed up retransmissions, does not necessarily mean that a protocol is not of class (A1). Important is that positive acknowledgments are necessary, for example to release data from the sender's buffer space. An example for a sender-initiated protocol is XTP [5].

B. *Receiver-Initiated Protocols*

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. 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 NAK is returned to the sender. Pure receiver-initiated protocols have a non-deterministic characteristic, since the sender is unable to decide when all group members have correctly received a data packet.

Receiver-initiated protocols can either send NAKs using unicast or multicast transmission. The protocol class sending unicast NAKs will be called (N1). The approach using multicast NAKs (N2) is known as NAK-avoidance scheme. A receiver that has detected an error sends a multicast NAK provided that it has not already received a NAK for this data packet from another receiver. Thus, in optimum 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) [6].

C. *Tree-Based Protocols*

Tree-based approaches organize the receivers into a tree structure called ACK tree, which is responsible for collecting acknowledgments and sending retransmissions. We assume that the sender is the root of the tree. If a receiver needs a retransmission, the parent node in the ACK tree is informed rather than the sender. A parent node is called group leader for its children which form a local group. Note that a group leader may also be a child of another local group. A child that is only a receiver rather than a group leader is called leaf node.

The first considered scheme of this class (H1) is similar to sender-initiated protocols since it uses ACKs sent by the receivers to their group leaders to indicate correctly received packets. Each group leader that is not the root node also sends an ACK to its parent group leader until the root node is reached. If a timeout for an ACK occurs at a group leader or the root, a multicast retransmission is invoked. An example of a protocol similar to our definition of (H1) is RMTP [7]. The second scheme (H2) is based on NAKs with NAK suppression similar to (N2) and selective ACKs (SAKs), which are sent periodically for deciding deterministically when packets can be removed from memory. A SAK is sent after a certain number of packets are received or after a certain time period has expired, to propagate the state of a receiver to its group leader. TMTP [8] is an example for class (H2).

Before the next scheme will be introduced, it is necessary to understand that (H1) and (H2) can guarantee reliable delivery only if no group member fails in the system. Assume for example that a group leader G_1 fails after it has acknowledged correct reception of a packet to its group leader G_0 which is the root node. If a receiver of G_1's local group needs a retransmission, neither G_1 nor G_0 can resend the data packet since G_1 has failed and G_0 has removed the packet from memory. This problem is solved by aggregated hierarchical ACKs (AAKs) of the third scheme (H3). A group leader sends an AAK to its parent group leader after all children have acknowledged correct reception. After a group leader or the root node has received an AAK, it can remove the corresponding data from memory because all members in this subhierarchy have already received it correctly. Lorax is an example for such a protocol [9]. Our definition of its generic behavior is as follows:

1. Group leaders send an ACK to its parent after the data packet was received correctly.
2. Leaf node receivers send an AAK to its parent after the data packet was received correctly.
3. Group leaders wait a certain time to receive ACKs from their children. If a timeout occurs, the packet is retransmitted to all children or selective to those whose ACK is missing. Since leaf node receivers send only AAKs rather than ACKs, a received AAK from a receiver is also allowed to prevent the retransmission.
4. Group leaders wait to receive AAKs from their children. Upon reception of all AAKs, the corresponding packet can be removed from memory and an AAK is sent to the parent. If a timeout occurs while waiting for AAKs, a unicast AAK query is sent to the affected nodes.
5. If a group leader or leaf node receiver receives further data packets after an AAK has been sent or the prerequisites for sending an AAK are met, these data packets are acknowledged by AAKs rather than ACKs. The same applies for receiving an AAK query that is replied with an AAK if the prerequisites are met.

Besides the AAK scheme, we consider in our analysis of (H3) a threshold scheme to decide whether a retransmission is performed using unicast or multicast. The group leader compares the number of missing ACKs with a threshold parameter. If the number of missing ACKs is smaller than this threshold, the data packets are retransmitted using unicast. Otherwise, the overall network and node load is assumed to be lower using multicast retransmission.

IV. Maximimum Throughput Analysis

A. *Model*

Our model is similar to the one used by Pingali et al. [3] and Levine et al. [4]. This means, that our analysis is based on the per packet processing requirements. A single sender X is assumed, multicasting to R identical receivers Y. In case of tree-based protocols, the sender is the root of the ACK tree. We assume that nodes do not fail, i.e. that retransmissions are finally successful. In contrast to previous work, packet loss can occur on both, data packets *and* control packets. Multicast packet loss probability is given by q and unicast packet loss probability by p for any node. All parameters are summarized in Table 1, 2 and 3. Table 1 shows the definitions by Levine et al. [4] and Pingali et al. [3]. Table 2 summarizes the additional notations for considering control message loss and Table 3 the additional notations for the protocol class (H3).

We assume that losses at different nodes are independent events. In fact, since receivers share parts of the multicast routing tree, this assumption does not hold in real networks. However, no protocol class is privileged relative to another one by this assumption.

Table 1
Notation used by Pingali et al. [3] and Levine et al. [4]

R	Size of the receiver set.
B	Branching factor of a tree.
X_f, Y_f	Time to feed in a new packet from the higher protocol layer or to deliver a packet to a higher layer, respectively.
X_d, X_t	Time to process the transmission respectively time to process a timeout at the sender.
X_a, X_n, X_Φ	Time to receive and process an ACK, NAK, or periodic ACK, respectively.
Y_d, Y_t	Time to receive and process a data packet, or to process a timeout at the receiver.
Y_a, Y_n, Y_Φ	Time to process and transmit an ACK, NAK, or periodic ACK.
Y'_n	Time required to receive and process a NAK at the receiver.
X^w, Y^w, H^w	Processing time per packet at sender, receiver, or group leader respectively. $w \in \{A1, N1, N2, H1, H2, H3\}$
$\Lambda_s^w, \Lambda_r^w, \Lambda_h^w, \Lambda^w$	Throughput for protocols w at the sender, receiver, group leader and overall system throughput.

Table 2
Additional notation for the analysis of control message loss

S	Number of periodical SAKs received by the sender.
p_D, q_D	Probability for unicast or multicast data loss.
p_A, p_N	Probability for unicast ACK or NAK loss.
q_N	Probability for multicast NAK loss.
$\tilde{p}, \bar{p}$	Probability that a retransmission is necessary for protocol (A1) or (N2), respectively.
p_s	Probability for simultaneous and therefore unnecessary NAK sending in (N2) and (H2).
$\widetilde{L}_r^w, \widetilde{L}^w$	Number of ACKs or NAKs per data packet from receiver r or from all receivers that reach the sender.
$\widetilde{N}_r^w$	Total number of transmissions per data packet received by receiver r.
M_r^w, M^w	Number of necessary transmissions for receiver r or for all receivers, to receive a data packet correctly 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, respectively.
$O_e^w, O_{e,r}^w$	Total number of empty rounds or empty rounds for receiver r, respectively.
N_k	Number of NAKs sent in round k.

B. *Sender-Initiated Protocol (A1)*

To determine the maximum throughput we analyze the processing times at the sender X^{A1} and receivers Y^{A1}, where Y^{A1} denotes a single receiver. The throughput is then limited by the highest processing requirement to successfully send a multicast packet or successfully receive a packet, respectively. We assume that the sender waits until all ACKs are received and then sends a retransmission if necessary. At the sender we have:

$$X^{A1} = \text{(higher protocol layer)} + \text{(initial transmission)}$$
$$+ \text{(timeouts)} + \text{(retransmissions)} + \text{(receiving ACKs)}$$

$$X^{A1} = X_f + X_d(1) + \sum_{m=2}^{M^{A1}} \Big(X_t(m) + X_d(m) \Big) + \sum_{i=1}^{\widetilde{L}^{A1}} X_a(i) \quad (1)$$

$$E(X^{A1}) = E(X_f) + E(M^{A1})E(X_d) + \Big(E(M^{A1}) - 1 \Big) E(X_t) + E(\widetilde{L}^{A1})E(X_a). \quad (2)$$

$X_a(i)$ denotes the processing requirement to receive an ACK packet for the i-th transmission. Analogous, $X_t(m)$ and $X_d(m)$ are the processing requirements for a timeout or data packet transmission for the m-th transmission. M^{A1} is the total number of transmissions necessary to transmit a packet correctly to all receivers in the presence of data packet and ACK loss and $\widetilde{L}^{A1}$ is the total number of ACKs received for this packet. $E(X^{A1})$ is the expectation of the processing requirement at the sender. The only unknowns are $E(M^{A1})$ and $E(\widetilde{L}^{A1})$:

$$E(\widetilde{L}^{A1}) = RE(M^{A1})(1 - q_D)(1 - p_A). \quad (3)$$

This means, the sender gets one ACK per data packet transmission $E(M^{A1})$ from every receiver R, provided that the data packet is not lost with probability $(1 - q_D)$ and the ACK is not lost with probability $(1 - p_A)$.

The probability for a retransmission is:

$$\tilde{p} = q_D + (1 - q_D)p_A, \quad (4)$$

i. e. either a data packet is lost (q_D) or the data packet is received correctly and the ACK is lost $((1 - q_D)p_A)$. So, the probability that the number of necessary transmissions M_r^{A1} for receiver r is smaller or equal to m is:

$$P(M_r^{A1} \le m) = 1 - \tilde{p}^m, m = 1, 2, \ldots \quad (5)$$

As the packet losses at different receivers are independent from each other:

$$P(M^{A1} \le m) = \prod_{r=1}^{R} P(M_r^{A1} \le m) = (1 - \tilde{p}^m)^R \quad (6)$$

$$P(M^{A1} = m) = P(M^{A1} \le m) - P(M^{A1} \le m - 1), \quad m = 1, 2, \ldots \quad (7)$$

It follows from [3] that $E(M^{A1})$ is:

$$E(M^{A1}) = \sum_{i=1}^{R} \binom{R}{i} (-1)^{i+1} \frac{1}{1 - \tilde{p}^i}. \quad (8)$$

$E(M^{A1})$ is the expected total number of necessary transmissions to receive the data packet correctly at all receivers.

Now $E(X^{A1})$ is entirely determined. The maximum rate for the sender Λ_s^{A1}, to send data packets successfully to the receivers is:

$$\Lambda_s^{A1} = \frac{1}{E(X^{A1})}. \quad (9)$$

Accordingly, the processing requirement for a packet at the receiver is:

$$Y^{A1} = \text{(receiving packets)} + \text{(sending ACKs)} + \text{(higher layer)} \quad (10)$$

$$E(Y^{A1}) = E(M^{A1})(1 - q_D)\Big(E(Y_d) + E(Y_a) \Big) + E(Y_f). \quad (11)$$

And the maximum packet processing rate Λ_r^{A1} of a receiver is:

$$\Lambda_r^{A1} = \frac{1}{E(Y^{A1})}. \quad (12)$$

Overall system throughput Λ^{A1} is determined by the minimum of the processing rates at the sender and receivers:

$$\Lambda^{A1} = min\{\Lambda_s^{A1}, \Lambda_r^{A1}\}. \quad (13)$$

C. *Receiver-Initiated Protocol (N1)*

Now we want to analyze the first receiver-initiated approach. Data packets are always transmitted using multicast and reliability is ensured by unicast NAKs. The sender collects all NAKs received within a certain timeout period and sends only one retransmission independent of the number of losses during that round. We will therefore analyze the number of necessary rounds O^{N1} to correctly deliver a data packet. A round starts with the sending of a data packet and ends with the expiration of the sender's timeout. Normally, there will be one data transmission in each round. However, if the sender receives no NAKs due to NAK loss, no retransmission is made and new NAKs must be sent by the receivers in the next round. O_r^{N1} is the number of necessary rounds for receiver r.

The processing requirement at the sender is:

$$X^{N1} = \text{(higher layer)} + \text{(transmissions)} + \text{(receiving NAKs)} + \text{(timer)}$$

$$X^{N1} = X_f + \sum_{m=1}^{M^{N1}} X_d(m) + \sum_{i=1}^{\widetilde{L}^{N1}} X_n(i) + \sum_{m=1}^{O^{N1}} X_t(m) \tag{14}$$

$$E(X^{N1}) = E(X_f) + E(M^{N1})E(X_d) + E(\widetilde{L}^{N1})E(X_n) + E(O^{N1})E(X_t). \tag{15}$$

The only unknowns are $E(M^{N1})$, $E(\widetilde{L}^{N1})$ and $E(O^{N1})$. The number of transmissions, M^{N1}, until all receivers correctly receive a packet does not change in the presence of NAK loss. M^{N1} is determined analogous to Equation 8 of (A1) with q_D instead of $\tilde{p}$. However, receivers must process more NAKs and NAK timeouts to invoke those retransmissions.

The number of rounds is the sum of the number of necessary rounds for sending transmissions M^{N1} and the number of empty rounds O_e^{N1} in which all NAKs are lost and therefore no retransmission is made:

$$O^{N1} = M^{N1} + O_e^{N1} \tag{16}$$

$$O_r^{N1} = M_r^{N1} + O_{e,r}^{N1}. \tag{17}$$

O_r^{N1}, M_r^{N1} and $O_{e,r}^{N1}$ are the corresponding numbers for a single receiver.

The number of transmissions, M_r^{N1}, for a single receiver is given by the probability q_D. This means, M_r^{N1} counts the number of trials until the first success occurs. The probability for the first success in a Bernoulli experiment at trial k with probability for success $(1 - q_D)$ is:

$$P(X = k) = (1 - q_D){q_D}^{k-1}. \tag{18}$$

The necessary number of transmissions for a single receiver M_r^{N1} follows from the Bernoulli distribution and [3]:

$$E(M_r^{N1}) = \frac{1}{1 - q_D} \tag{19}$$

$$E(M_r^{N1} | M_r^{N1} > 1) = \frac{2 - q_D}{1 - q_D} \tag{20}$$

$$E(M_r^{N1} | M_r^{N1} > 2) = \frac{3 - 2q_D}{1 - q_D} \tag{21}$$

$$P(M_r^{N1} > 1)[E(M_r^{N1} | M_r^{N1} > 1) - 1] = E(M_r^{N1}) - 1. \tag{22}$$

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

$$N_k = {q_D}^k R, \tag{23}$$

where ${q_D}^k$ is the probability for a single receiver that until round k all data packets are lost. The number of empty rounds after transmission k is determined by the failure probability:

$$p_k = {p_N}^{N_k} = {p_N}^{{q_D}^k R}. \tag{24}$$

p_k is the probability that all sent NAKs in round k are lost. The number of sent NAKs is equal to the number of receivers ${q_D}^k R$ that need a retransmission in round k (see Equation 23). The expected number of empty rounds $E(O_e^{N1})$ is the expected number of empty rounds after the first transmission plus the expected number of empty rounds after the second transmission and so on. Now, $E(O_e^{N1})$ and $E(O_{e,r}^{N1})$ can be determined analogous to M_r^{N1}:

$$E(O_e^{N1}) = \sum_{k=1}^{E(M^{N1})-1} \left(\frac{1}{1-p_k} - 1\right) \tag{25}$$

$$E(O_{e,r}^{N1}) = \sum_{k=1}^{E(M_r^{N1})-1} \left(\frac{1}{1-p_k} - 1\right). \tag{26}$$

$(1/1 - p_k)$ is the expectation for the number of empty rounds plus the last successful NAK reception at the sender which is subtracted.

$\widetilde{L}^{N1}$ is the number of NAKs received by the sender and ϑ_1 is the total number of NAKs sent in all rounds:

$$E(\widetilde{L}^{N1}) = \vartheta_1(1 - p_N) \tag{27}$$

$$\vartheta_1 = \sum_{k=1}^{E(M^{N1})} N_k \frac{1}{1-p_k}. \tag{28}$$

Finally, at the receiver we have:

$$E(Y^{N1}) = E(Y_f) + E(M^{N1})[1 - q_D]E(Y_d) + P(O_r^{N1} > 1)[E(O_r^{N1} | O_r^{N1} > 1) - 1]E(Y_n) + P(O_r^{N1} > 2)[E(O_r^{N1} | O_r^{N1} > 2) - 2]E(Y_t). \tag{29}$$

Note that the last, successful transmission is not replied with a NAK and that a NAK timeout occurs not for the first and the last transmission.

The processing rates and throughput can be obtained analogous to protocol (A1).

D. Receiver-Initiated Protocol (N2)

In contrast to (N1), this protocol class sends NAKs to all group members using multicast. Ideally, NAK suppression ensures that only one NAK is received by the sender. As in the previous protocol, the sender collects all NAKs belonging to one round and then starts a retransmission.

$$X^{N2} = \text{(higher layer)} + \text{(transmissions)} + \text{(receiving NAKs)} + \text{(NAK timer)}$$

$$X^{N2} = X_f + \sum_{m=1}^{M^{N2}} X_d(m) + \sum_{i=1}^{\widetilde{L}^{N2}} X_n(i) + \sum_{j=1}^{O^{N2}} X_t(j) \tag{30}$$

$$E(X^{N2}) = E(X_f) + E(M^{N2})E(X_d) + E(\widetilde{L}^{N2})E(X_n) + E(O^{N2})E(X_t) \tag{31}$$

$E(M^{N2})$ is determined analogous to (A1) and (N1) with loss probability q_D. $\widetilde{L}^{N2}$ contains the number of necessary and additional NAKs received at the sender:

$$E(\widetilde{L}^{N2}) = \vartheta_1(1 - q_N) \tag{32}$$

$$\vartheta_1 = \sum_{k=1}^{E(M^{N2})} N_k \frac{1}{1-p_k}. \tag{33}$$

N_k, the number of NAKs sent in round k, is the sum of: NAK of the first receiver that did not receive the data packet plus NAK of another unsuccessful receiver that did not receive the first NAK packet and sends a second NAK and so on:

$$N_k = \sum_{i=1}^{R} N_{k,i} \tag{34}$$

$$N_{k,1} = {q_D}^k \tag{35}$$

$$\begin{aligned} N_{k,2} &= {q_D}^k(1 - N_{k,1} + N_{k,1}q_N) \\ &= N_{k,1}(1 - N_{k,1} + N_{k,1}q_N) \\ &= N_{k,1} - N_{k,1}^2 + N_{k,1}^2 q_N \end{aligned} \tag{36}$$

$$N_{k,n} = N_{k,n-1} - N_{k,n-1}^2 + N_{k,n-1}^2 q_N \quad , n > 1. \tag{37}$$

The first receiver sends a NAK provided that the data packet was lost with probability ${q_D}^k$. The second receiver sends a NAK provided that the data packet was lost and the NAK of the first receiver was lost ($N_{k,1}q_N$) or the first receiver sends no NAK ($1 - N_{k,1}$), and so on.

In Equation 37, a perfect system model is assumed in which additional NAKs are only sent due to NAK loss at receivers. This means, receivers must have synchronized local clocks and a defined sending order for NAKs. However, since receivers are usually not synchronized in real systems it can occur that NAKs are sent simultaneously. Therefore, we extend Equations 35-37 with the probability for simultaneous NAK sending (p_s) to:

$$N_{k,1} = {q_D}^k \tag{38}$$

$$N_{k,n} = N_{k,n-1} - N_{k,n-1}^2 + N_{k,n-1}^2(q_N + p_s - q_N p_s) \quad , n > 1. \tag{39}$$

The number of rounds O^{N2} is obtained analogous to protocol (N1). It is the sum of the number of necessary rounds for sending transmissions M^{N2} and the number of empty rounds O_e^{N2} in which all NAKs are lost and therefore no retransmission is made:

$$O^{N2} = M^{N2} + O_e^{N2} \tag{40}$$

$$O_r^{N2} = M_r^{N2} + O_{e,r}^{N2}. \tag{41}$$

O_r^{N2}, M_r^{N2} and $O_{e,r}^{N2}$ are the corresponding numbers for a single receiver.

The number of necessary transmissions, M_r^{N2}, for a single receiver is given by the probability q_D. Analogous to Equation 19 of protocol (N1) the expectation is:

$$E(M_r^{N2}) = \frac{1}{1-q_D}. \tag{42}$$

The number of empty rounds after transmission k is determined by the failure probability:

$$p_k = q_N^{N_k}. \tag{43}$$

p_k is the probability that all sent NAKs in round k are lost. The expected number of empty rounds $E(O_e^{N2})$ is equal to the expected number of empty rounds after the first transmission plus the expected number of empty rounds after the second transmission and so on. Now, $E(O_e^{N2})$ and $E(O_{e,r}^{N2})$ can be determined analogous to M_r^{N2} (see Equation 19):

$$E(O_e^{N2}) = \sum_{k=1}^{E(M^{N2})-1} \left(\frac{1}{1-p_k} - 1\right) \tag{44}$$

$$E(O_{e,r}^{N2}) = \sum_{k=1}^{E(M_r^{N2})-1} \left(\frac{1}{1-p_k} - 1\right). \tag{45}$$

$(1/1-p_k)$ is the expectation for the number of empty rounds plus the last successful NAK reception at the sender, which is subtracted.

At the receiver we have:

$$\begin{aligned} E(Y^{N2}) &= E(Y_f) + E(M^{N2})(1-q_D)E(Y_d) \\ &+ P(O_r^{N2} > 1)[E(O_r^{N2}|O_r^{N2} > 1) - 1]\frac{\vartheta_2}{\vartheta_3}E(Y_n) \\ &+ \Big[P(O^{N2} > 1)[E(O^{N2}|O^{N2} > 1) - 1]\vartheta_2 \\ &- P(O_r^{N2} > 1)[E(O_r^{N2}|O_r^{N2} > 1) - 1]\frac{\vartheta_2}{\vartheta_3}\Big](1-q_N)E(Y_n') \\ &+ P(O_r^{N2} > 2)[E(O_r^{N2}|O_r^{N2} > 2) - 2]E(Y_t). \end{aligned} \tag{46}$$

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:

$$\vartheta_2 = \frac{1}{E(O^{N2})} \sum_{k=1}^{E(M^{N2})} N_k \frac{1}{1-p_k} \tag{47}$$

$$\vartheta_3 = \frac{1}{E(O^{N2})} \sum_{k=1}^{E(M^{N2})} {q_D}^k R \frac{1}{1-p_k}, \tag{48}$$

where $(1/1-p_k)$ is the number of empty rounds plus the last successful NAK sending (see Equations 19, 44 and 45).

The third term in $E(Y^{N2})$ is the processing requirement to send NAKs, where the considered receiver r is only with probability ϑ_2/ϑ_3 the one that sends a NAK. In the forth term the number of sent NAKs is subtracted from the number of total NAKs to get the number of received NAKs.

The sending rate and throughput can be obtained analogous to protocol (A1).

E. *Tree-Based Protocol (H1)*

Protocol (H1) uses solely unicast ACKs for controlling the reliable message delivery. Our analysis distinguishes between the three different kinds of nodes in the ACK tree, the sender at the root of the tree, the receivers that form the leaves of the ACK tree and the receivers that are non-leaf nodes. We will call these non-leaf receivers group leaders. Group leaders are sender and receiver as well.

Our analysis of all tree-based protocols is based on the assumption that each local group consists of exactly B members and one group leader. We assume further, that when a group leader has to sent a retransmission, the group leader has already received this packet correctly. The following subsections analyze the processing requirements at the sender, receivers and group leaders.

1) *Sender (root node)*

$$X^{H1} = X_f + X_d(1) + \sum_{m=2}^{M^{H1}} \Big(X_t(m) + X_d(m)\Big) + \sum_{i=1}^{\widetilde{L}^{H1}} X_a(i) \tag{49}$$

$$\begin{aligned} E(X^{H1}) &= E(X_f) + E(M^{H1})E(X_d) \\ &+ \Big(E(M^{H1}) - 1\Big)E(X_t) + E(\widetilde{L}^{H1})E(X_a) \end{aligned} \tag{50}$$

M^{H1} is the number of necessary transmissions until all members of a local group have received a packet correctly. $E(M^{H1})$ is determined analogous (B instead of R) to Equation 8 of protocol (A1), since every local group is like a sender-based system. Furthermore, the number of ACKs received by group leaders in the presence of possible ACK loss $E(\widetilde{L}^{H1})$ is similar to $E(\widetilde{L}^{A1})$, with B instead of R:

$$E(\widetilde{L}^{H1}) = BE(M^{H1})(1-q_D)(1-p_A) \tag{51}$$

2) *Receiver (leaf node)*

$E(\widetilde{N}_r^{H1})$ is the number of received transmissions at receiver r:

$$E(\widetilde{N}_r^{H1}) = E(M^{H1})(1-q_D). \tag{52}$$

So, the processing requirement Y^{H1} for a receiver is:

$$Y^{H1} = Y_f + \sum_{i=1}^{\widetilde{N}_r^{H1}} \Big(Y_d(i) + Y_a(i)\Big) \tag{53}$$

$$E(Y^{H1}) = E(\widetilde{N}_r^{H1})\Big(E(Y_d) + E(Y_a)\Big) + E(Y_f) \tag{54}$$

$$= E(M^{H1})(1-q_D)\Big(E(Y_d) + E(Y_a)\Big) + E(Y_f). \tag{55}$$

3) *Group leader (inner node)*

Since a group leader is a sender and receiver as well:

$$\begin{aligned} H^{H1} = &\underbrace{\sum_{m=2}^{M^{H1}} \Big(X_t(m) + X_d(m)\Big) + \sum_{k=1}^{\widetilde{L}^{H1}} X_a(k)}_{\text{as sender}} \\ &+ \underbrace{Y_f + \sum_{i=1}^{\widetilde{N}_r^{H1}} \Big(Y_d(i) + Y_a(i)\Big)}_{\text{as receiver}}. \end{aligned} \tag{56}$$

Please note that the initial transmission $(X_d(1))$ is not considered in the equation since it is sent using the multicast routing tree rather than the ACK tree.

$$\begin{aligned} E(H^{H1}) &= \Big(E(M^{H1}) - 1\Big)\Big(E(X_d) + E(X_t)\Big) + E(\widetilde{L}^{H1})E(X_a) \\ &+ E(M^{H1})(1-q_D)\Big(E(Y_d) + E(Y_a)\Big) + E(Y_f) \end{aligned} \tag{57}$$

$$= E(X^{H1}) + E(Y^{H1}) - E(X_f) - E(X_d(1)) \tag{58}$$

The maximum rates Λ_s^{H1}, Λ_r^{H1}, Λ_h^{H1} for the sender, receiver and group leader, respectively, are:

$$\Lambda_s^{H1} = \frac{1}{E(X^{H1})}, \Lambda_r^{H1} = \frac{1}{E(Y^{H1})}, \Lambda_h^{H1} = \frac{1}{E(H^{H1})} \tag{59}$$

Overall system throughput Λ^{H1} is given by the minimum of the packet processing rates for the sender, receiver and group leader:

$$\Lambda^{H1} = min\{\Lambda_s^{H1}, \Lambda_h^{H1}, \Lambda_r^{H1}\} \tag{60}$$

F. Tree-Based Protocol (H2)

(H2) uses selective periodical ACKs (SAKs) and NAKs with NAK avoidance. Group leaders collect all NAKs belonging to one round and send a retransmission if the waiting time has expired and at least one NAK has been received. We have to distinguish between the number of rounds and the number of transmissions. The number of rounds is equal or greater than the number of transmissions, since if a sender or receiver receives no NAK within one round, no retransmission is invoked.

A SAK is sent by the receiver to announce its state, i.e. its received and missed packets, after a sequence of data packets have been received. We assume that a SAK is sent after a certain period of time. Therefore, when analyzing the processing requirements for a *single* packet, only the proportionate requirements for sending (Y_Φ) and receiving (X_Φ) a SAK is considered. S is assumed to be the number of SAKs received by the sender in the presence of possible SAK loss, where $S = (1 - p_A)B$.

1) Sender (root node)

$$X^{H2} = X_f + \sum_{i=1}^{M^{H2}} X_d(i) + \sum_{j=1}^{\widetilde{L}^{H2}} X_n(j) + \sum_{m=1}^{O^{H2}} X_t(m) + SX_\Phi \quad (61)$$

$$\begin{aligned} E(X^{H2}) &= E(X_f) + E(M^{H2})E(X_d) + E(\widetilde{L}^{H2})E(X_n) \\ &\quad + E(O^{H2})E(X_t) + E(S)E(X_\Phi) \end{aligned} \quad (62)$$

$E(M^{H2})$, $E(\widetilde{L}^{H2})$ and $E(O^{H2})$ are determined analogous to (N2) (B instead of R). We have SAKs as unicast control messages, as well as multicast NAKs with NAK avoidance. Since SAKs are sent only periodically, mainly NAKs are responsible for retransmissions. Ideally, the NAK avoidance scheme ensures that only one NAK is received by the group leaders.

2) Receiver (leaf node)

$$\begin{aligned} E(Y^{H2}) &= E(Y_f) + E(M^{H2})(1 - q_D)E(Y_d) + E(Y_\Phi) \\ &+ P(O_r^{H2} > 1)[E(O_r^{H2}|O_r^{H2} > 1) - 1]\frac{\vartheta_2}{\vartheta_3}E(Y_n) \\ &+ \Big[P(O^{H2} > 1)[E(O^{H2}|O^{H2} > 1) - 1]\vartheta_2 \\ &\quad - P(O_r^{H2} > 1)[E(O_r^{H2}|O_r^{H2} > 1) - 1]\frac{\vartheta_2}{\vartheta_3}\Big](1 - q_N)E(Y_n') \\ &+ P(O_r^{H2} > 2)[E(O_r^{H2}|O_r^{H2} > 2) - 2]E(Y_t) \end{aligned} \quad (63)$$

ϑ_2 and ϑ_3 can be obtained analogous to (N2) with B instead of R.

3) Group leader (inner node)

As the group leader role contains the sender role and the receiver role as well, the processing requirements are:

$$E(H^{H2}) = E(X^{H2}) + E(Y^{H2}) - E(X_f) - E(X_d(1)) \quad (64)$$

The rates for sender, receiver, group leader and overall system throughput for (H2) can be obtained analogous to (H1).

G. Tree-Based Protocol (H3)

We assume that the correct transmission of a data packet consists of two phases. In the first phase, the data is transmitted and ACKs are collected until all ACKs are received, i.e. until all nodes have received the data packet. Then the second phase starts, in which the missing AAKs are collected. Note that most AAKs are already received in phase one, since AAKs are sent as soon as all children have sent their AAKs. In this case, a retransmission is acknowledged with an AAK rather than an ACK. So, only nodes whose AAK is missing must be queried in phase two.

1) Sender (root node)

$$\begin{aligned} X^{H3} &= X_f + \sum_{j=1}^{M_m^{H3}} X_{d,m}(j) + \sum_{k=1}^{M_u^{H3}} N_u X_{d,u}(k) + \sum_{s=1}^{M^{H3}-1} X_t(s) \\ &+ \sum_{i=1}^{\widetilde{L}_a^{H3}} X_a(i) + \sum_{v=1}^{O_q^{H3}} X_t(v) + \sum_{w=1}^{L_{aaq}^{H3}} X_{aaq}(w) + \sum_{z=1}^{\widetilde{L}_{aa}^{H3}} X_{aa}(z) \end{aligned} \quad (65)$$

M_m^{H3} and M_u^{H3} are the number of necessary multicast or unicast transmissions, respectively. M^{H3} is the total number of transmissions. $X_{d,m}$ and $X_{d,u}$ determine the processing requirements for a multicast or unicast packet transmission. X_{aa} is the time for the sender to receive and process an AAK and $\widetilde{L}_{aa}^{H3}$ is the number of received AAKs. The processing of AAKs is similar to the processing of data packets and ACKs. If AAKs are missing after a timeout has occurred, the sender or group leader sends unicast AAK query messages to the corresponding child nodes. Note that this processing is started after all ACKs are received and no further retransmissions due to lost data packets are necessary. O_q^{H3} is the number of necessary query rounds and L_{aaq}^{H3} is the number of necessary unicast AAK queries in the presence of message loss.

With p_t, the probability that unicast is used for retransmissions, the number of unicast and multicast transmissions are:

$$M_u^{H3} = p_t(M^{H3} - 1) \quad (66)$$

$$M_m^{H3} = (1 - p_t)(M^{H3} - 1) + 1. \quad (67)$$

Table 3
Additional notations for the analysis of (H3)

X_{aa}, Y_{aa}	Time to receive and process an AAK at the sender, or process the transmission of an AAK at the receiver, respectively.
X_{aaq}, Y_{aaq}	Time to send an AAK query at the sender, or receive and process an AAK query at the receiver.
p_q	Probability for AAK query loss at the receiver.
p_{AA}	Probability for unicast AAK loss at the sender.
n_k	Current number of receivers that need a retransmission.
ϕ	Threshold for unicast retransmission. If n_k is smaller than ϕ, unicast is used for retransmission and multicast otherwise.
p_t	Probability that n_k is smaller than threshold ϕ and therefore unicast is used for retransmissions.
N_u	Mean number of sent unicast messages per packet retransmission.
τ	Probability that a retransmission is necessary due to data or ACK loss.
$\hat{p}$	Probability that an AAK query fails.
M^{H3}	Total number of necessary transmissions in the presence of data loss. $M^{H3} = M_u^{H3} + M_m^{H3}$
M_u^{H3}, M_m^{H3}	Number of necessary unicast or multicast transmissions in the presence of failures, respectively.
$X_{d,u}, X_{d,m}$	Time to send a data packet per unicast or multicast, respectively.
O_q^{H3}	Number of necessary AAK query rounds.
L_a^{H3}, L_{aa}^{H3}	Number of ACKs or AAKs sent by a receiver.
$\widetilde{L}_a^{H3}$, $\widetilde{L}_{aa}^{H3}$	Number of ACKs or AAKs received by the sender.
L_{aaq}^{H3}, $\widetilde{L}_{aaq}^{H3}$	Number of AAK queries sent by the sender or received by a receiver, respectively.
B_{aa}	Number of receivers from which the AAK is missing when phase two starts.
p_c	Probability that no AAK can be sent due to missing AAKs of child nodes.

Please note that the first transmission is always sent with multicast. The probability for a retransmission due to data or ACK loss is given by:

$$\tau = \underbrace{p_t p_D + (1-p_t)q_D}_{\text{data loss}} + \underbrace{\Big[1 - \Big(p_t p_D + (1-p_t)q_D\Big)\Big]p_A}_{\text{no data loss but ACK loss}}. \quad (68)$$

$E(M^{H3})$ is determined by τ instead of $\tilde{p}$ and B instead of R analogous to Equation 8 of protocol (A1).

ϕ is the threshold for unicast or multicast retransmissions. If the current number of nodes n_k, which need a retransmission is smaller than the threshold ϕ, then unicast is used for retransmission. p_t is the probability that the current number of nodes n_k is smaller than the threshold ϕ:

$$p_t = \frac{1}{M^{H3}} \sum_{k=1}^{M^{H3}} \begin{cases} 1, n_k < \phi \\ 0, n_k \geq \phi \end{cases} \quad (69)$$

Since p_t is used to obtain M^{H3}, p_t can only be determined if $q_D = p_D$. In this case, parameter p_t is unnecessary to determine M^{H3}.

N_u is the mean number of receivers per round for which a unicast retransmission is invoked:

$$N_u = \frac{1}{M_u^{H3}} \sum_{k=1}^{M^{H3}} \begin{cases} n_k, n_k < \phi \\ 0, n_k \geq \phi \end{cases} \quad (70)$$

$E(\tilde{N}_r^{H3})$ is the total number of transmissions that reach receiver r with unicast and multicast:

$$E(\tilde{N}_r^{H3}) = \frac{N_u}{B} E(M_u^{H3})(1-p_D) + E(M_m^{H3})(1-q_D). \quad (71)$$

The number of ACKs that reach the sender or group leader in the presence of ACK loss is given by:

$$E(\tilde{L}_a^{H3}) = BE(\tilde{N}_r^{H3})(1-p_A)p_c. \quad (72)$$

p_c is the probability that no AAK can be sent due to missing AAKs of child nodes.

The number of AAK query rounds O_q^{H3}, is determined by the probability $\hat{p}$ that a query fails:

$$\hat{p} = p_q + (1-p_q)p_{AA}. \quad (73)$$

Now, $E(O_q^{H3})$ can be determined analogous to M^{A1} of protocol (A1) (see Equation 8) with B_{aa} instead of R and $\hat{p}$ instead of $\tilde{p}$. B_{aa} is the number of receivers, the sender has to query when the first AAK timeout occurs, which is equal to the number of receivers that have not already successfully sent an AAK in the first phase:

$$E(O_q^{H3}) = \sum_{i=1}^{B_{aa}} \binom{B_{aa}}{i} (-1)^{i+1} \frac{1}{1-\hat{p}^i} \quad (74)$$

$$B_{aa} = B\Big(p_c + (1-p_c)p_{AA}\Big)^{E(\tilde{N}_r^{H3})}. \quad (75)$$

$p_c + (1-p_c)p_{AA}$ is the probability that no AAK can be sent in a round or that the AAK is lost.

Queries are sent with unicast to the nodes whose AAK is missing. The total number of queries in all rounds are:

$$E(L_{aaq}^{H3}) = \sum_{k=1}^{E(O_q^{H3})} B_{aa}\hat{p}^{(k-1)}. \quad (76)$$

The number of AAKs received at the sender is the number of AAKs in the retransmission phase plus the number of AAKs in the AAK query phase, which is exactly one AAK from every receiver in B_{aa} (see Equation 72).

$$E(\tilde{L}_{aa}^{H3}) = BE(\tilde{N}_r^{H3})(1-p_{AA})(1-p_c) + B_{aa}. \quad (77)$$

Now, $E(X^{H3})$ is entirely determined by:

$$\begin{aligned} E(X^{H3}) &= E(X_f) + E(M_u^{H3})N_u E(X_{d,u}) + E(M_m^{H3})E(X_{d,m}) \\ &+ E(M^{H3}-1)E(X_t) + E(\tilde{L}_a^{H3})E(X_a) + E(O_q^{H3})E(X_t) \\ &+ E(L_{aaq}^{H3})E(X_{aaq}) + E(\tilde{L}_{aa}^{H3})E(X_{aa}). \end{aligned} \quad (78)$$

2) Receiver (leaf node)

Y_{aa} is the required time at the receiver to send an AAK and Y_{aaq} is the processing requirement to receive an AAK query. So, the processing requirement at the receiver is given by:

$$\begin{aligned} Y^{H3} &= Y_f + \sum_{i=1}^{\tilde{N}_r^{H3}} Y_d(i) + \sum_{j=1}^{L_a^{H3}} Y_a(j) \\ &+ \sum_{k=1}^{L_{aa}^{H3}} Y_{aa}(k) + \sum_{l=1}^{\tilde{L}_{aaq}^{H3}} \Big(Y_{aa}(l) + Y_{aaq}(l)\Big). \end{aligned} \quad (79)$$

The number of transmissions that are acknowledged with an ACK, L_a^{H3}, or with an AAK, L_{aa}^{H3} are:

$$L_a^{H3} = p_c E(\tilde{N}_r^{H3}) \quad (80)$$

$$L_{aa}^{H3} = (1-p_c)E(\tilde{N}_r^{H3}). \quad (81)$$

$\tilde{L}_{aaq}^{H3}$, the number of AAK queries received by an receiver are:

$$\tilde{L}_{aaq}^{H3} = \frac{1}{B_{aa}} E(L_{aaq}^{H3})(1-p_q), \quad (82)$$

where $\frac{1}{B_{aa}}$ is the probability to be a receiver that gets an AAK query.

$$\begin{aligned} E(Y^{H3}) &= E(Y_f) + E(\tilde{N}_r^{H3})E(Y_d) + E(L_a^{H3})E(Y_a) + E(L_{aa}^{H3})E(Y_{aa}) \\ &+ E(\tilde{L}_{aaq}^{H3})\Big(E(Y_{aa}) + E(Y_{aaq})\Big) \end{aligned} \quad (83)$$

3) Group leader (inner node)

The processing requirement at a group leader consists of the sender and receiver processing requirements:

$$E(H^{H3}) = E(X^{H3}) + E(Y^{H3}) - E(X_f) - E(X_{d,m}(1)). \quad (84)$$

The rate for sender, receiver and group leader as well as the overall system throughput for (H3) can be obtained analogous to (H1).

V. Numerical Results

We examine the relative performance of the analyzed protocols to investigate the influence of control packet loss and the performance of the new class (H3). All mean processing costs are set equal to 1, except for the periodic costs $X\phi$ and $Y\phi$ which are set to 0.1. The following graphs show the throughput of the various protocol classes relative to the normalized maximum throughput of 1.

Figure 1 shows the throughput of the sender-initiated protocol (A1) and the receiver-initiated protocols (N1) and (N2). The data packet loss probability is 0.1. The dotted curve is the throughput without considering control message loss while in the solid curve control message loss probability is set equal to the data loss probability. For (N2), the probability for simultaneous NAK sending is 0 for the dotted curve and 0.2 for the solid one. The results in Figure 1 show, that (A1) performs poorly and that the throughput is further decreased by ACK packet loss. (N1) is very robust against control message loss. In fact, the throughput even increases with NAK loss. (N1)'s throughput is limited by an overwhelming of the sender with NAK messages. Since one NAK packet is sufficient to start a retransmission, NAK loss of that scale decreases the sender's load and therefore increases throughput. (N2)'s throughput decreases significantly in the presence of NAK loss and asynchronous local clocks. However, (N2) continues to provide the best relative throughput.

In Figure 2, the results for the hierarchical protocol classes (H1), (H2) and (H3) are shown. The number of child nodes is set equal to 10 for all classes. (H3) is shown with $\phi = 0$ which corresponds with (H1) except for the additional aggregated ACKs of (H3). $\phi = 0$ means that all retransmission are

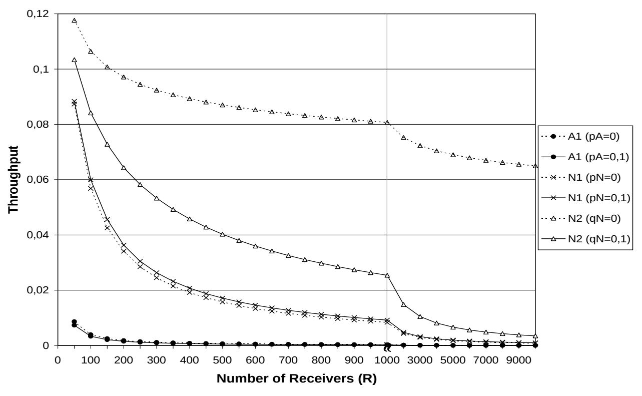

Figure 1: Throughput of sender- and receiver-initiated protocols

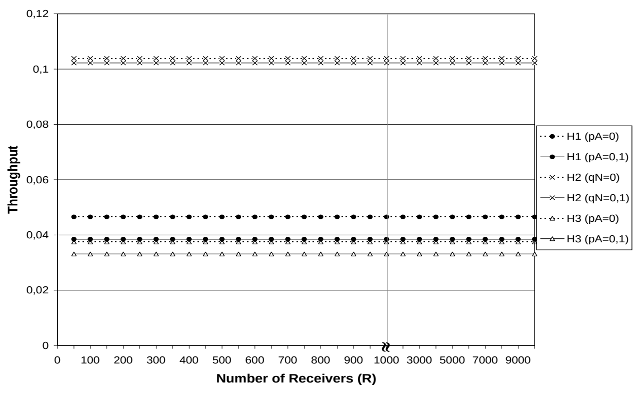

Figure 2: Throughput of tree-based protocols

sent with multicast. It is shown, that all protocol classes experience a throughput degradation. For (H1), (H2) and (H3) the throughput degradation is about 20%, 3% and 16%, respectively. This means, for that small local group size of 10 nodes, the throughput of (H2), which is the multicast NAK protocol with NAK avoidance, decreases least of all and is again the protocol with the best performance. Of particular importance is the only small degradation of (H3)'s throughput compared to (H1). This means, that the additional costs for providing true reliability even in the presence of node failures are small and therefore acceptable.

Due to readability, the result for (H3) with $\phi = 2$ is not shown in the figure. With $\phi = 2$, only retransmissions for equal or more than 2 nodes are made using multicast and with unicast otherwise. In this case, (H3) provides even slightly better performance than (A1), since the receivers have to receive and acknowledge less retransmissions.

Of course, such a scheme with aggregated ACKs is also applicable for a protocol with multicasted NAKs. In this case, the relative decrease in throughput will be higher than in ACK based protocols, since more AAKs must be sent and processed to provide deterministic behaviour. Nevertheless, we assume that such a protocol will provide a throughput comparable to (N2).

VI. Conclusions

We have analyzed the throughput of sender-initiated, receiver-initiated and tree-based multicast protocols assuming a realistic system model with data packet loss, control packet loss and asynchronous clocks. Our analysis is based on the processing requirements for the sender, receivers and group leaders. Of particular importance is the analyzed protocol class with aggregated acknowledgments. In contrast to other hierarchical approaches this class provides reliability even in the presence of node failures.

Our numerical results show that protocols with multicasted NAKs and NAK avoidance experience a significant throughput decrease in the presence of NAK loss and non-synchronized clocks. However, for small local group sizes, which is the case in hierarchical approaches, the throughput decrease is only 3%. On balance, protocols with multicasted NAKs and NAK avoidance continue to provide the best performance.

The protocol class with aggregated acknowledgments leads to only a small throughput decrease compared to the same class without aggregated acknowledgments. This means, that the additional costs for providing a reliable multicast service even in the presence of node failures are small and therefore acceptable for reliable multicast protocol implementations.

Currently we are working on further throughput analysis, which are based on delay requirements rather than processing requirements. Our objective is not only to analyze throughput assuming a realistic system model to compare the different protocol classes, but also to be able to configure protocols to provide the highest throughput. For example, in tree-based protocols the local group size is a crucial parameter which has to be investigated in more detail by future work.

VII. References

[1] K. Rothermel and C. Maihöfer, "A robust and efficient mechanism for constructing multicast acknowledgment trees," in *Proceedings of the Eight International Conference on Computer Communications and Networks*, New York, Oct. 1999, pp. 139–145, IEEE.

[2] C. Maihöfer, "Scalable and reliable multicast ack tree construction with the token repository service," in *Proceedings of the IEEE International Conference on Networks (ICON)*, New York, Sept. 2000, IEEE.

[3] 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.

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

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

[6] 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.

[7] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharyya, "Reliable multicast transport protocol (RMTP)," *IEEE Journal on Selected Areas in Communications, special issue on Network Support for Multipoint Communication*, vol. 15, no. 3, pp. 407–421, Apr. 1997.

[8] R. Yavatkar, J. Griffioen, and M. Sudan, "A reliable dissemination protocol for interactive collaborative applications," in *The Third ACM International Multimedia Conference and Exhibition (MULTIMEDIA '95)*, New York, Nov. 1996, pp. 333–344, ACM Press.

[9] B. Levine, D. Lavo, and J. Garcia-Luna-Aceves, "The case for reliable concurrent multicasting using shared ack trees," in *Proceedings of the Fourth ACM Multimedia Conference (MULTIMEDIA'96)*, New York, Nov. 1996, pp. 365–376, ACM Press.