A Delay Analysis of Tree-based 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

Abstract—

We present a comparative delay analysis of tree-based reliable multicast protocols and show the influence of varying sending rates, group sizes, packet loss probabilities and branching factors of the control tree. 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 all tree-based protocols provide low delays and good scalability. From the four considered protocol classes, NAK-based protocols achieve the best scalability but ACK-based protocols achieve the lowest delays.

An important aspect of our work is to be of practical relevance rather than being of only theoretical nature. Therefore, we have compared the analytical results with a RMTP and TMTP simulation. Both show similar results which confirms that our analysis can help to choose a suitable protocol and to tune them for improved performance.

I. Introduction

In analysis and simulation studies concerning bandwidth and processing load, tree-based reliable multicast protocols have proven to provide scalability for a large number of receivers. In tree-based protocols, the members of a multicast group are organized in a so-called control tree to overcome the well-known acknowledgment implosion problem of flat approaches, i.e., overwhelming of the sender by a large number of positive (ACKs) or negative acknowledgments (NAKs). A positive acknowledgment returned by a receiver confirms correct message delivery, whereas a negative acknowledgment asks for a message retransmission. Since acknowledgments are propagated along the edges of the control tree in a leaf-to-root direction, the implosion problem can be avoided by limiting the branching factor of a node and thus the number of acknowledgment messages.

In this paper we present a delay analysis of tree-based reliable multicast protocols. The message delivery delay is an important issue for multimedia applications. For example real time applications like interactive distributed simulations, distributed games, or the delivery of MPEG I-frames [1] benefit from guaranteed reliability and low delays. Besides time constraints of some applications, low delays are vital for providing high throughput with a window based sending scheme [2].

In contrast to previous delay analysis we assume a more realistic system model as explained later in Section II. Besides analyzing the delay between sender and receiver we determine the round trip delay between sending a data packet and receiving the last corresponding control packet at the sender. The round trip delay determines the time after a data packet can be removed from memory and influences the sending rate if the sender uses a window based sending scheme. Furthermore, knowledge about this delay is important to adjust the retransmission timeout at the sender.

Our numerical results show that all tree-based protocols provide good scalability and low delays compared to non-hierarchical approaches. To be more precise, NAK-based protocols achieve the best scalability but ACK-based protocols achieve the lowest delays. With respect to the branching factor, the optimal value depends on several parameters like packet loss probability, protocol class and whether average delivery delay, maximum delivery delay or round trip delay is of interest. We can conclude, though, that a tuned branching factor can significantly reduce delay. To assess the analytical results we have implemented the RMTP [3] and TMTP protocol [4] in the network simulator NS-2 environment [5] and compared the analytical results with simulation results. Both show similar results with varying number of receivers, transmission rates, loss probabilities and branching factors, which shows that our analytical approach is adequate.

The remainder of this paper is structured as follows. In the next section related work is discussed. In Section III we discuss the analyzed protocol classes. In Section IV we introduce our assumed system model followed by the

detailed delay analysis. Numerical results are presented in Section V and compared with simulation results in Section VI. 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. [6]. They have compared the processing requirements of flat protocol classes. Levine et al. [7] 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 [8] 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. Analysis of generic reliable multicast protocols were done by Kasera et al. [9], Nonnenmacher et al. [10] and Poo et al. [11]. In [9], local recovery techniques are analyzed and compared. Nonnenmacher et al. [10] studied the performance gain of protocols using parity packets to recover from transmission errors. In contrast to previous work, [11] analyzes go-back-N and selective-repeat error recovery schemes rather than merely a stop-and-wait approach. Recently, end system bandwidth requirements were analyzed rather than total bandwidth consumption within the network [12].

Regarding delay analysis, the first comparative analysis of sender- (ACK) and receiver-initiated (NAK) approaches was presented by Yamamoto et al. [13] and DeCleene [14]. 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 [13] is independent of the network topology, in [14] a delay analysis of generic ACK- and NAK-based protocols operating over star and linear topologies was presented. In [10] 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 and comparatively simple. For example, they do not consider queuing delay in detail and neglect feedback processing. They concluded that local recovery techniques and parity packets outperforms other approaches.

Our paper extends previous work in five significant ways. First, to our knowledge this is the first comparative analysis of generic classes of tree-based reliable multicast protocols, which considers feedback traffic and queuing delays. Second, we consider the loss of control packets rather than assuming reliable delivery. In previous work, control packets are assumed to be reliably delivered, which especially favors protocols with multicast NAK and NAK suppression scheme. NAK suppression works most efficiently if no NAKs are lost at receivers and the sender and therefore, only one NAK per lost data packet is sufficient. Third, we assume that local clocks are not synchronized which again affects the NAK suppression scheme. In related work it was assumed that in case of data packet loss only one NAK is returned to the sender. Our assumption allows that multiple NAKs are sent. Fourth, 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 of the various protocol classes, 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. Finally, the 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 sending window. The fifth significant difference from previous work is that we have implemented the RMTP [3] and TMTP [15] protocol in a simulation environment and compared the results.

III. Classification of Tree-based Multicast Protocols

A. ACK-based Protocol (H1)

The first considered scheme is denoted as (H1). As in all other protocol classes we assume that the initial sender is the root of the control tree and that the initial transmission is multicasted to the global group. Global group denotes the whole multicast group in contrast to a local group, which is described below. (H1) uses ACKs sent by receivers to their parent in the control tree, called group leader, in order to indicate correctly received packets. Each group leader that is not the root node also sends

an ACK to its parent as soon as a data packet has been received. If a timeout for an ACK occurs at a group leader, a multicast retransmission is invoked for this local group. A local group encompasses a group leader and its directly attached children. Such a retransmission can be sent to a separate multicast address for this local group or sent to the global group address and limited in scope by the TTL value. An example of a protocol similar to our definition of (H1) is RMTP [3]. RMTP uses subtree multicasting to limit the retransmission scope.

B. NAK-based Protocol (H2)

The second scheme (H2) is based on NAKs with NAK suppression [16]. NAKs are sent by means of multicast to the group leader and other nodes of this local group. A receiver that misses a data packet sends a NAK provided that it has not already received a NAK from another receiver that also misses the data packet. NAKs alone do not allow a deterministic decision when packets can be removed from memory at the sender. Therefore, selective ACKs (SAKs) are sent after a certain number of packets has been received or after a certain time period has been expired, to propagate the state of a receiver to its group leader. TMTP [4] is an example for class (H2).

C. ACK and AAK-based Protocol (H3)

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 after all children have acknowledged correct reception. After a group leader has received an AAK, it can remove the corresponding data from memory because all members in this subhierarchy (i.e. the transitive closure of the child relation) have already received it correctly. RMTP II is an example for a protocol that uses AAKs [17]. Our definition of its generic behaviour is as follows:

1. A group leaders sends an ACK to its parent after a data packet has been received correctly.
2. A leaf node receiver of the control tree sends an AAK to its parent after a data packet has been received correctly.
3. Group leaders wait a certain time to receive ACKs from all 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 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 a group leader sends an AAK to its parent. If a timeout occurs while waiting, a unicast AAK query is sent to the affected nodes.
5. If retransmissions or AAK queries are received by a node after an AAK has been sent or the prerequisites for sending an AAK are met, an AAK is sent to the parent instead of an ACK.

Besides AAKs, 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 below this threshold, the data packets are retransmitted using unicast. Otherwise, if the number of missing ACKs exceeds the threshold, the overall network and node load is assumed to be lower using multicast retransmission.

D. NAK and AAK-based Protocol (H4)

Our next protocol will be denoted as (H4) and is a combination of the negative acknowledgment with NAK suppression scheme (H2) and aggregated acknowledgments. Similar to (H2), NAKs are used to start a retransmission. Instead of selective periodical ACKs, aggregated ACKs are used to announce the receivers' state and allow group leaders to remove data from memory. Like SAKs, we assume that AAKs are sent periodically. We define the generic behaviour of (H4) as follows:

1. Upon detection of a missing or corrupted data packet, receivers send a NAK to the local group by means of multicast scheduled at a random time in the future and provided that not already a NAK for this data packet is received before the scheduled time. If no retransmission arrives within a certain time period, the NAK sending scheme is repeated.
2. Group leaders retransmit a packet to the local group by means of multicast if a NAK has been received.
3. After a certain number of correctly received data packets, leaf node receivers send an AAK to their group leader in the control tree. A group leader forwards an AAK to its parent as soon as the data packets are correctly received and the corresponding AAKs from all child nodes have been received.
4. Group leaders initiate a timer and wait for all AAKs to be received. If the timer expires, an AAK query is sent to those child nodes whose AAK is missing.
5. If an AAK query is received by a node and the prerequisites for sending an AAK are met, the query is acknowl-

edged with an AAK.

IV. 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 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 protocol 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 [18].

The number of necessary data packet transmissions M is determined by the packet loss probabilities q_D, p_A, and q_N. M has already been determined for the various protocol classes in our processing and bandwidth requirements analysis [8], [12].

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 I.

TABLE I
Frequently used Notations

q_D	Probability for multicast data loss at a receiver.
p_A, q_N	Probability for unicast ACK or multicast NAK loss.
R	Size of the receiver set.
B	Branching factor of a tree or the local group size.
W_S^w, W_R^w, W_G^w	Waiting time for the sender, receiver or group leader. $w \in \{H1, H2, H3, H4\}$
λ	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.
$\lambda_{r,u}^S, \lambda_{r,m}^S$	Flow of unicast or multicast retransmissions.
λ^R	Data packet reception flow at the receiver.
λ_s^S, λ_s^R	SAK flow at the sender or receiver.
$\lambda_{n,g}^R$	Flow of transmitted NAK packets at the receiver.
$\lambda_{n,r}^R$	Flow of received NAK packets at the receiver.
λ_{aa}^S	Flow of received AAK packets at the sender.
λ_q^S, λ_q^R	AAK query flow at the sender or receiver.
$\varrho_S^w, \varrho_R^w, \varrho_G^w$	Total load on the sender, receiver or group leader.
T	Timeout delay.
τ, τ_H	Global or hierarchical network propagation delay.
B^{H2}, B^{H4}	Random NAK suppression delay.
X, Y, Z	Processing time for data packets, control packets or periodical control packets.
$h, \tilde{h}$	Maximum and mean height of the control tree.
M^w, M_r^w	Total number of retransmissions for all receivers or for receiver r, respectively.
I	Delay for the initial transmission.
H	Delay for a hierarchical retransmission.
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.

C. Protocol Independent Methods

If a node in a tree-based protocol has lost a data packet and a retransmission is needed, the retransmission request (either by a NAK packet or a missing ACK packet) is sent to the group leader. If this group leader has lost the data packet as well, the group leader's group leader is queried an so forth. As a prerequisite for the delay analysis we will determine the height of the control tree. We define the root node's height as 1. The height of every other node is the height of the parent node plus 1. With this definition, the height can be obtained as follows, where R is the number of receivers:

$$
\begin{aligned}
R &= \sum_{i=0}^{h-1} B^i = B^0 + B^1 + \ldots + B^{h-2} + B^{h-1} \\
&= \frac{(1-B)B^0}{1-B} + \frac{(1-B)B^1}{1-B} + \ldots + \frac{(1-B)B^{h-2}}{1-B} + \frac{(1-B)B^{h-1}}{1-B} \\
&= \frac{B^0 - B^1 + B^1 - B^2 + \ldots + B^{h-2} - B^{h-1} + B^{h-1} - B^h}{1-B}
\end{aligned}
$$

$$= \frac{1 - B^h}{1 - B}, \tag{1}$$

and the tree height follows to:

$$h = \log_B \Big(R(B-1) + 1 \Big), \tag{2}$$

where B is the number of members in a local group (i.e. the branching factor of the control tree). To obtain the mean delay, we obtain the average tree height $\tilde{h}$:

$$\tilde{h} = \frac{\Big(\sum_{i=1}^{h-2} (i+1) * B^i\Big) + \Big(R - \sum_{j=1}^{h-2} B^j\Big) h}{R}. \tag{3}$$

D. ACK-based Protocol (H1)

For a delay analysis of tree-based protocols we distinguish among sender, receivers and group leaders. Although the sender is a group leader as well, here and in the following we will denote only inner nodes as group leaders. All delay components are shown in Figure 1.

D.1 Mean Waiting Times at the Sender (Root Node)

First, we have to determine the mean waiting time for a packet between generation or arrival and completion of processing or sending. The mean waiting time is determined by the load of a node, i.e. the processing of incoming and outgoing packet flows. The sender has to process the following three arriving packet flows:

1. Data packets from the higher protocol layer that are transmitted for the first time. This packet flow is referred to as λ_i^S and has rate λ. The processing time for a data packet is assumed to be X.
2. Data packets that are retransmitted due to packet loss. This packet flow is referred to as λ_r^S and has rate $\lambda(E(M^{H1}) - 1)$, since every packet is $(E(M^{H1}) - 1)$-times retransmitted. $E(M^{H1})$ is the expected total number of transmissions per packet until all multicast group members have received it correctly.
3. Control packets are received by the sender with flow λ_a^S and rate $\lambda B E(M^{H1})(1 - q_D)(1 - p_A)$. B is the branching factor of the ACK-tree, i.e. the number of child nodes per group leader. The processing time for an ACK packet is assumed to be Y.

The expected total number of necessary transmissions $E(M^{H1})$ to receive the data packet correctly at all receivers is given in [8]:

$$E(M^{H1}) = \sum_{i=1}^{B} \binom{B}{i} (-1)^{i+1} \frac{1}{1 - (q_D + (1 - q_D) p_A)^i}. \tag{4}$$

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

$$\varrho = \lambda E(S). \tag{5}$$

The load on the sender (ϱ_S^{H1}, traffic intensity) is then the sum of the packet rates:

$$\varrho_S^{H1} = \lambda E(M^{H1}) E(X) + \lambda B E(M^{H1})(1 - q_D)(1 - p_A) E(Y). \tag{6}$$

As explained in Section IV-B, the system can be modeled as a $M|G|1$ queue. The Pollaczek-Chintchine formula gives the mean number of requests to be processed $E(L)$ [18]:

$$E(L) = \varrho + \frac{\varrho^2 + \lambda^2 Var(S)}{2(1 - \varrho)}. \tag{7}$$

With the formula of Little [18]:

$$E(L) = \lambda E(T), \tag{8}$$

the mean waiting time of a request in the system $E(T)$ is (see Eq. 5, 7 and 8):

$$E(T) = E(S) + \frac{\varrho^2 + \lambda^2 Var(S)}{2\lambda(1 - \varrho)}. \tag{9}$$

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

$$E(W) = E(T) - E(S) = \frac{\varrho^2 + \lambda^2 Var(S)}{2\lambda(1 - \varrho)}. \tag{10}$$

With Eq. 5 and $Var(X) = E(X^2) - (E(X))^2$:

$$E(W) = \frac{\lambda E(S^2)}{2(1 - \varrho)} \tag{11}$$

$$E(W_S^{H1}) = \frac{(\lambda_r^S + \lambda_i^S) E(X^2) + \lambda_a^S E(Y^2)}{2(1 - \varrho_S^{H1})}. \tag{12}$$

D.2 Mean Waiting Times at a Receiver (Leaf Node)

The only packet flow at a receiver is the reception of data packets which are acknowledged by an ACK, λ^R, with rate $\lambda E(M^{H1})(1 - q_D)$. The processing time is $X + Y$ since the arrival of a data 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:

$$\varrho_R^{H1} = \lambda E(M^{H1})(1 - q_D) \Big(E(X) + E(Y) \Big). \tag{13}$$

The mean waiting time of a packet at the receiver until processing starts is (see Eq. 11):

$$E(W_R^{H1}) = \frac{\lambda^R E\Big((X + Y)^2\Big)}{2(1 - \varrho_R^{H1})}. \tag{14}$$

With X and Y are independent random variables, $E(X_1 + X_2) = E(X_1) + E(X_2)$, $Var(X_1) = E(X_1^2) - (E(X_1))^2$ and $Var(X_1 + X_2) = Var(X_1) + Var(X_2)$:

$$E(W_R^{H1}) = \frac{\lambda^R \Big(E(X^2) + E(Y^2) + 2E(X)E(Y) \Big)}{2(1 - \varrho_R^{H1})}. \tag{15}$$

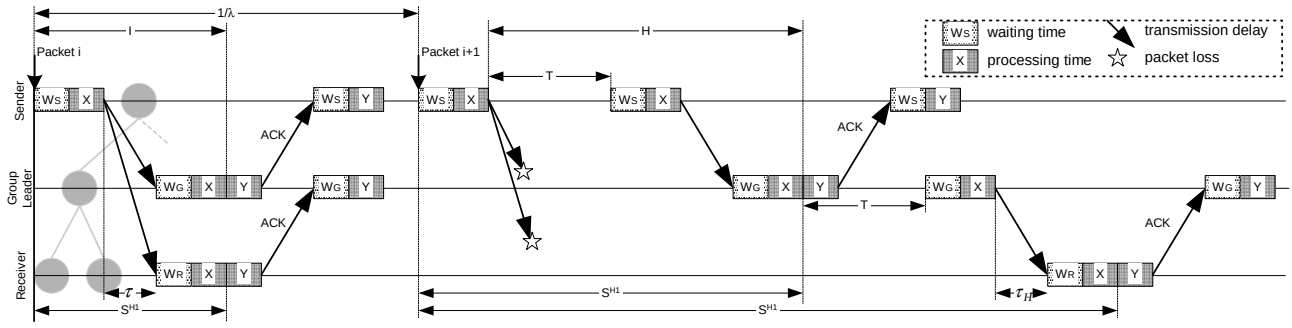


Fig. 1. Packet delivery delay for protocol (H1)

D.3 Mean Waiting Times at a Group Leader (Inner Node)

The load on an inner node is the sender load without the initial transmission and the receiver load:

$$\varrho_G^{H1} = \underbrace{\lambda\Big(E(M^{H1})-1\Big)}_{\lambda_r^S} E(X) + \underbrace{\lambda B E(M^{H1})(1-q_D)(1-p_A)}_{\lambda_a^S} E(Y) + \underbrace{\lambda E(M^{H1})(1-q_D)}_{\lambda^R} E(X+Y). \quad (16)$$

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

$$E(W_G^{H1}) = \frac{\lambda_r^S E(X^2) + \lambda_a^S E(Y^2)}{2(1-\varrho_G^{H1})} + \frac{\lambda^R\Big(E(X^2)+E(Y^2)+2E(X)E(Y)\Big)}{2(1-\varrho_G^{H1})}. \quad (17)$$

D.4 Overall Delay of Protocol (H1)

T is the group leader timeout delay, τ is the network propagation delay, $\tilde{h}$ is the average number of hierarchy levels of the control tree and B is the branching factor. If no retransmission is necessary, the delay from the initial transmission $E(I)$ is:

$$E(I) = E(W_S^{H1}) + E(X) + \tau + E(W_G^{H1}) + E(X). \quad (18)$$

Note that a simplifying pessimistic assumption we made is that the receiver is always a group leader and therefore take $E(W_G^{H1})$ in the above equation.

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

$$E(H) = \Big(E(M_r^{H1}|M_r^{H1}>1)-1\Big)\Big(T + E(W_G^{H1}) + E(X)\Big) + \tau_H + E(W_G^{H1}) + E(X). \quad (19)$$

M_r^{H1} is the number of transmissions for a single receiver r and τ_H the network propagation delay for a hierarchical retransmission. For obtaining the overall delay, we determine the probabilities that no data loss occurs, that a node misses a packet but the parent node is able to retransmit it, that a node and its parent misses that packet and the next parent retransmits it and so forth and multiply these probabilities with the expected delays. The overall delay is then:

$$E(S_\phi^{H1}) = \Big[\sum_{i=o}^{\tilde{h}-2} q_D^i (1-q_D)\Big(E(I)+iE(H)\Big)\Big] + q_D^{\tilde{h}-1}\Big(E(W_S^{H1})+E(X)+(\tilde{h}-1)E(H)\Big). \quad (20)$$

Besides the delay for delivering data packets to the receivers we want to examine the delay for receiving all ACK packets at the sender:

$$E(S_{RTD}^{H1}) = \underbrace{\Big(E(M^{H1})-1\Big)\Big(T+E(W_S^{H1})+E(X)\Big)}_{\text{sending transmissions}} + \underbrace{E(X)+E(W_S^{H1})+\tau_H+E(W_G^{H1})+E(X)}_{\text{receiving last successful retransmission}} + \underbrace{E(Y)+E(W_G^{H1})+\tau_H+E(W_S^{H1})+E(Y)}_{\text{send and receive last ACK}}. \quad (21)$$

Besides the mean delivery delay we can determine the expected delay to reliably deliver a certain percentage of data packets. We assume that γ is the percentage of data packets that has to be reliably delivered and $E(S_\gamma^{H1})$ is the expected delay, which can be obtained as follows:

$$\gamma = 1 - q_D^{M_\gamma^{H1}} \quad (22)$$

$$M_\gamma^{H1} = \frac{ln(1-\gamma)}{ln(q_D)}, \gamma \geq q_D \quad (23)$$

$$E(H) = \Big(E(M_\gamma^{H1}|M_\gamma^{H1}>1)-1\Big)\Big(T+E(W_G^{H1})+E(X)\Big) + \tau_H + E(W_G^{H1}) + E(X) \quad (24)$$

$$E(S_\gamma^{H1}) = E(W_S^{H1}) + E(X) + (\tilde{h}-1)E(H). \quad (25)$$

In Eq. 25 retransmissions encompass all nodes on the path from the sender to a random receiver r. Our assumption here is that all parent nodes first have to receive a message with the desired probability γ before retransmissions can be sent to r.

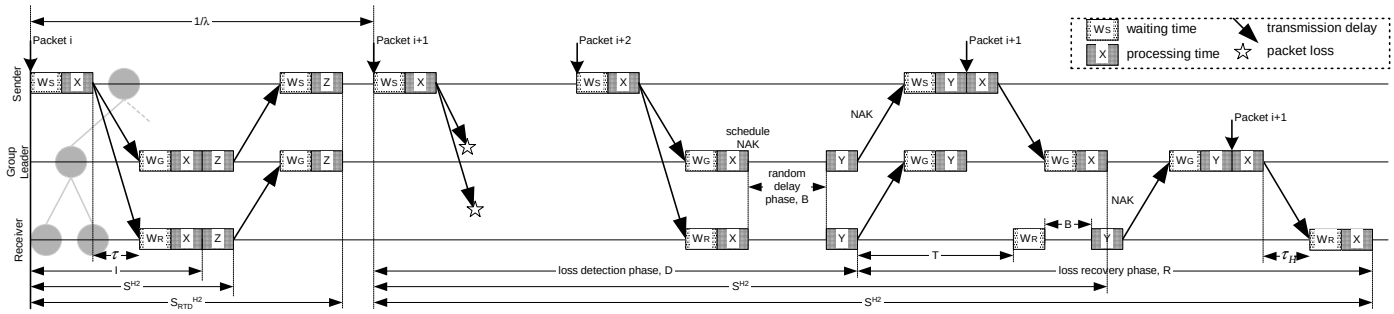


Fig. 2. Packet delivery delay for protocol (H2)

E. NAK-based Protocol (H2)

E.1 Mean Waiting Times at the Sender (Root Node)

At the sender we distinguish among the following four 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 $\lambda\Big(E(M^{H2})-1\Big)$ and processing time $X+Y$. Third, the flow of additional NAKs, which are not necessary to trigger a retransmission λ_n^S with rate $\lambda E(\widetilde{L}^{H2})-\lambda_r^S$ and processing time Y. And finally, the flow of selective (periodical) acknowledgments (SAKs) λ_s^S with rate $\lambda B(1-p_A)$ and processing time Z. All delay components are shown in Figure 2.

The number of transmissions is (see Eq. 4 and [8]):

$$E(M^{H2}) = \sum_{i=1}^{B} \binom{B}{i} (-1)^{i+1} \frac{1}{1-{q_D}^i}. \tag{26}$$

The number of received NAKs $E(\widetilde{L}^{H2})$ is given in the processing requirements analysis [8].

Given these flows, the load on the sender is:

$$\varrho_S^{H2} = \lambda E(M^{H2})E(X) + \lambda E(\widetilde{L}^{H2})E(Y) + \lambda B(1-p_A)E(Z). \tag{27}$$

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

$$E(W_S^{H2}) = \frac{\lambda_t^S E(X^2) + \lambda_r^S \Big(E(X^2)+E(Y^2)+2E(X)E(Y)\Big)}{2(1-\varrho_S^{H2})} + \frac{\lambda_s^S E(Z^2)}{2(1-\varrho_S^{H2})}. \tag{28}$$

E.2 Mean Waiting Times at a Receiver (Leaf Node)

At the receiver we distinguish among four packet flows. λ^R is the flow of data packets from the sender with rate $\lambda E(M^{H2})(1-q_D)$ and processing time X. The second flow $\lambda_{n,g}^R$ consists of the submitted NAK packets with rate $\lambda\Big(E(O_r)-1\Big)\frac{\vartheta_2}{\vartheta_3}$ and processing time Y. 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 $\lambda(1-q_N)\Big[\vartheta_2\Big(E(O)-1\Big)-\frac{\vartheta_2}{\vartheta_3}\Big(E(O_r)-1\Big)\Big]$ and processing time Y. The last flow λ_s^R are the sent SAKs with rate λB and processing time Z.

The total number of rounds O, the number of rounds for a single receiver O_r, as well as ϑ_2 and ϑ_3 are given in the processing requirements analysis [8].

With these flows, the load on the receiver is:

$$\varrho_R^{H2} = \lambda E(M^{H2})(1-q_D)E(X) + (\lambda_{n,g}^R + \lambda_{n,r}^R)E(Y) + \lambda_s^R E(Z). \tag{29}$$

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

$$E(W_R^{H2}) = \frac{\lambda^R E(X^2) + (\lambda_{n,g}^R + \lambda_{n,r}^R)E(Y^2) + \lambda_s^R E(Z^2)}{2(1-\varrho_R^{H2})}. \tag{30}$$

E.3 Mean Waiting Times at a Group Leader (Inner Node)

The load on an inner node is the sender load without the initial transmission and the receiver load:

$$\begin{aligned}\varrho_G^{H2} = {} & \Big(E(M^{H2})-1\Big)E(X) + \lambda\widetilde{L}^{H2}E(Y) + \lambda B(1-p_A)E(Z) \\ & + \lambda E(M^{H2})(1-q_D)E(X) + (\lambda_{n,g}^R + \lambda_{n,r}^R)E(Y) \\ & + \lambda_s^R E(Z).\end{aligned} \tag{31}$$

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

$$\begin{aligned}E(W_G^{H2}) = {} & \frac{\lambda_r^S \Big(E(X^2)+E(Y^2)+2E(X)E(Y)\Big)}{2(1-\varrho_G^{H2})} \\ & + \frac{\lambda_n^S E(Y^2) + \lambda_s^S E(Z^2)}{2(1-\varrho_G^{H2})} \\ & + \frac{\lambda^R E(X^2) + (\lambda_{n,g}^R + \lambda_{n,r}^R)E(Y^2) + \lambda_s^R E(Z^2)}{2(1-\varrho_G^{H2})}.\end{aligned} \tag{32}$$

E.4 Loss Detection Phase

To obtain the overall delay, we distinguish between the following phases (taken from [13]):

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 NAKs can be lost, this phase includes the periodical sending of NAKs until the data packet is received correctly.

For the loss detection phase 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:

$$P(L = l|K = k) = q_D^{l(k+1)}(1 - q_D^{k+1}), \quad l = 0, 1, \ldots \text{ and } k = 0, 1, \ldots, R-1. \tag{33}$$

The number of subsequent lost packets at $k+1$ receiver is:

$$E(L|K = k) = \frac{q_D^{k+1}}{1 - q_D^{k+1}}. \tag{34}$$

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

$$E(L|K) = \sum_{k=0}^{B-1} \binom{B-1}{k} q_D^k (1-q_D)^{B-1-k} \frac{q_D^{k+1}}{1 - q_D^{k+1}}. \tag{35}$$

Note that Eq. 35 differs from the result of Yamamoto et al. [13]. 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:

$$\underbrace{E(W_S^{H2}) + E(X)}_{\text{sender processing delay}} + \underbrace{\tau}_{\text{propagation delay}} + \underbrace{E(W_G^{H2}) + E(X)}_{\text{receiver processing delay}} \tag{36}$$

Finally, the first phase can be expressed as follows:

$$E(D^{H2}) = \sum_{k=0}^{B-1} \binom{B-1}{k} q_D^k (1-q_D)^{B-1-k} \frac{q_D^{k+1}}{1 - q_D^{k+1}} \frac{1}{\lambda} \tag{37}$$

$$+ E(W_S^{H2}) + E(X) + \tau + E(W_G^{H2}) + E(X) + E(B^{H2}) + E(Y). \tag{38}$$

Note that here and in the following we make a pessimistic simplification in assuming that the receiver is always a group leader and therefore take $E(W_G^{H2})$.

E.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 $E(D^{H2})$ was unsuccessful. The following receiver or group leader timeouts have the length $T + E(W_G^{H2}) + E(B^{H2}) + E(Y)$, where T is the timeout period and W_G^{H2} the waiting time before processing starts at a group leader. B^{H2} is the random delay a receiver or group leader waits before a NAK is sent. This delay starts with the discovery of packet loss at the first receiver. In case of no NAK suppression it ends with the expiration of the backoff timer and the transmission of the initial 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 is:

$$\begin{aligned} E(R^{H2}) = &\Big(E(M_r^{H2}) + E(O_{e,r}) - 1\Big) \\ &* \Big(T + E(W_G^{H2}) + E(B^{H2}) + E(Y)\Big) \\ &+ \tau_H + E(W_G^{H2}) + E(Y) \\ &+ E(X) + \tau_H + E(W_G^{H2}) + E(X). \end{aligned} \tag{39}$$

The number of necessary transmissions for a single receiver r, $E(M_r^{H2})$, as well as the number of empty rounds, in which no retransmission is sent due to NAK loss, $E(O_{e,r})$, are given in [12].

E.6 Overall Delay of Protocol (H2)

If no retransmission is necessary, the delay from the initial transmission $E(I)$ is:

$$E(I) = E(W_S^{H2}) + E(X) + \tau + E(W_G^{H2}) + E(X). \tag{40}$$

For obtaining the overall delay, we determine the probabilities that no data loss occurs, that a node misses a packet but the parent node is able to retransmit it, that a node and its parent misses that packet and the next parent retransmits it and so forth and multiply these probabilities with the expected delays. The overall delay is then:

$$\begin{aligned} E(S_\phi^{H2}) = &(1-q_D)E(I) \\ &+ \Big[\sum_{i=1}^{\tilde{h}-2} q_D^i (1-q_D)\Big(E(I) + E(D^{H2}) + iE(R^{H2})\Big)\Big] \\ &+ q_D^{\tilde{h}-1}\Big(E(D^{H2}) + (\tilde{h}-1)E(R^{H2})\Big). \end{aligned} \tag{41}$$

Now we obtain the delay of a SAK. Although SAKs are sent periodically, we want to determine here the delay starting from the sending of a data packet and the reception of the last SAK packet belonging to this data packet. As there is no retransmission mechanism for lost SAKs in (H2) we assume that they are reliably delivered.

$$E(S_{RTD}^{H2}) = (1-q_D)^B E(I) \underbrace{+\Big(1-(1-q_D)^B\Big)\Big(E(D^{H2}) + E(R^{H2})\Big)}_{\text{sending transmissions}}$$

$$+ \underbrace{E(Y) + E(W_G^{H2}) + \tau_H + E(W_S^{H2}) + E(Y)}_{\text{sending and receiving the last SAK}} \quad (42)$$

We have to use the processing delay of control packets $E(Y)$ instead of periodic control packets since we obtain the delay on condition that a SAK is actually being sent.

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

$$\begin{aligned} E(R^{H2}) &= \Big(E(M_\gamma^{H2}) + E(O_{e,r}) - 1\Big) \\ &\quad * \Big(T + E(W_G^{H2}) + E(B^{H2}) + E(Y)\Big) \\ &\quad + \tau_H + E(W_G^{H2}) + E(Y) \\ &\quad + E(X) + \tau_H + E(W_G^{H2}) + E(X) \quad (43) \\ E(S_\gamma^{H2}) &= E(W_S^{H2}) + E(X) + E(D^{H2}) + (\tilde{h} - 1)E(R^{H2}). \quad (44) \end{aligned}$$

M_γ^{H2} is obtained analogous to Eq. 23 of protocol (H1).

F. ACK and AAK-based Protocol (H3)

Protocol (H3) is similar to (H1) but uses besides normal hierarchical ACKs, aggregated ACKs, so-called AAKs. Additionally, we analyze a threshold scheme to send the retransmission per unicast or multicast dependent on the number of lost data packets. All delay components are shown in Figure 3.

F.1 Mean Waiting Times at the Sender (Root Node)

The sender has to process the following six packet flows. The initial data packet flow per multicast λ_t^S with rate λ and processing time X. The retransmissions sent with unicast $\lambda_{r,u}^S$ with rate $\lambda E(N_u)E(M_u^{H3})$ and processing time X, where $E(N_u)$ is the average number of unicast messages per retransmission round. Retransmissions sent with multicast $\lambda_{r,m}^S$ with rate $\lambda(E(M_m^{H3}) - 1)$ and processing time X. The ACK flow λ_a^S with rate $\lambda E(\widetilde{L}_a^{H3})$ and the AAK flow λ_{aa}^S with rate $\lambda E(\widetilde{L}_{aa}^{H3})$, both with processing time Y. Finally, the AAK query flow λ_q^S with rate $\lambda E(L_{aaq}^{H3})$ and processing time Y. The number of unicast, $E(M_u^{H3})$, or multicast retransmissions, $E(M_m^{H3})$, the number of unicast messages per retransmission round $E(N_u)$, the number of ACK, $E(\widetilde{L}_a^{H3})$, and AAK packets, $E(\widetilde{L}_{aa}^{H3})$, and finally the number of AAK query messages, $E(L_{aaq}^{H3})$, are given in [12]. The total load on the sender is:

$$\begin{aligned} \varrho_S^{H3} &= \lambda\Big(E(N_u)E(M_u^{H3}) + E(M_m^{H3})\Big)E(X) \\ &\quad + \lambda\Big(E(\widetilde{L}_a^{H3}) + E(\widetilde{L}_{aa}^{H3}) + E(L_{aaq}^{H3})\Big)E(Y). \quad (45) \end{aligned}$$

The expected waiting time at the sender is then:

$$E(W_S^{H3}) = \frac{(\lambda_t^S + \lambda_{r,u}^S + \lambda_{r,m}^S)E(X^2) + (\lambda_a^S + \lambda_{aa}^S + \lambda_q^S)E(Y^2)}{2(1 - \varrho_S^{H3})}. \quad (46)$$

F.2 Mean Waiting Times at a Receiver (Leaf Node)

A receiver has to process the following flows. There is the data flow λ^R with rate $\lambda E(N_r^{H3})$ which automatically triggers an ACK or AAK flow and therefore results in processing time of $X + Y$. The flow of AAK queries λ_q^R with rate $\lambda E(\widetilde{L}_{aaq}^{H3})$ triggers the replying of AAKs, which results in total processing time $Y + Y$. $E(N_r^{H3})$ and $E(\widetilde{L}_{aaq}^{H3})$ are given in [12]. The load on a receiver is then:

$$\varrho_R^{H3} = \lambda E(N_r^{H3})E(X + Y) + \lambda E(\widetilde{L}_{aaq}^{H3})E(Y + Y). \quad (47)$$

The expectation of the waiting time at the receiver is:

$$E(W_R^{H3}) = \frac{\lambda^R E((X + Y)^2) + \lambda_q^R E((Y + Y)^2)}{2(1 - \varrho_R^{H3})}. \quad (48)$$

F.3 Mean Waiting Times at a Group Leader (Inner Node)

The load on an inner node is the sum of the sender's load without the initial transmission and a receiver's load:

$$\begin{aligned} \varrho_G^{H3} &= \lambda\Big(N_u E(M_u^{H3}) + E(M_m^{H3}) - 1\Big)E(X) \\ &\quad + \lambda\Big(E(\widetilde{L}_a^{H3}) + E(\widetilde{L}_{aa}^{H3}) + E(L_{aaq}^{H3})\Big)E(Y) \\ &\quad + \lambda E(N_r^{H3})E(X + Y) + \lambda E(\widetilde{L}_{aaq}^{H3})E(Y + Y). \quad (49) \end{aligned}$$

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

$$\begin{aligned} E(W_G^{H3}) &= \frac{\Big(\lambda_{r,u}^S + \lambda_{r,m}^S\Big)E(X^2) + \Big(\lambda_a^S + \lambda_{aa}^S + \lambda_q^S\Big)E(Y^2)}{2(1 - \varrho_G^{H3})} \\ &\quad + \frac{\lambda^R E((X + Y)^2) + \lambda_q^R E((Y + Y)^2)}{2(1 - \varrho_G^{H3})}. \quad (50) \end{aligned}$$

F.4 Overall Delay of Protocol (H3)

If no retransmission is necessary, the delay from the initial transmission $E(I)$ is:

$$E(I) = E(W_S^{H3}) + E(X) + \tau + E(W_G^{H3}) + E(X). \quad (51)$$

The time for a hierarchical retransmission $E(H)$ on condition that the parent has received the packet correctly is:

$$\begin{aligned} E(H) &= \Big(E(M_r^{H3} | M_r^{H3} > 1) - 1\Big)\Big(T + E(W_G^{H3}) + E(X)\Big) \\ &\quad + \tau_H + E(W_G^{H3}) + E(X). \quad (52) \end{aligned}$$

The overall delay is then analogous to protocol (H1):

$$\begin{aligned} E(S_\phi^{H3}) &= \Big[\sum_{i=o}^{\tilde{h}-2} q_D^i (1 - q_D)\Big(E(I) + iE(H)\Big)\Big] \\ &\quad + q_D^{\tilde{h}-1}\Big(E(W_S^{H1}) + E(X) + (\tilde{h} - 1)E(H)\Big). \quad (53) \end{aligned}$$

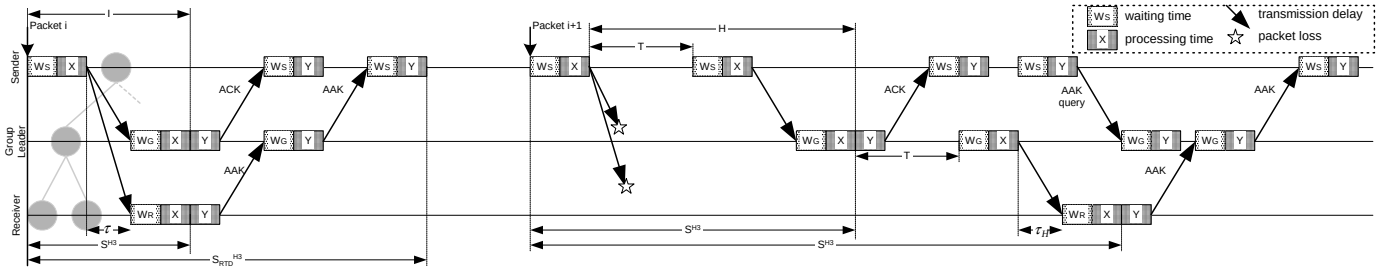


Fig. 3. Packet delivery delay for protocol (H3)

We want to determine the round trip delay of AAKs, since if AAKs are provided they are used to manage the sending window and free buffer space. Before an AAK could be sent from a receiver node, it must receive the data packet before. The mean waiting time between sending a packet and receiving the last AAK at the sender is given by:

$$E(S_{RTD}^{H3}) = \underbrace{\Big(E(M^{H3})-1\Big)\Big(T+E(W_G^{H3})+E(X)\Big)}_{\text{sending transmissions}} + \underbrace{E(X)+E(W_G^{H3})+\tau_H+E(W_G^{H3})+E(X)}_{\text{receiving last successful retransmission}} + (h-1)\Big[\underbrace{E(\widetilde{L}_{aaq}^{H3})\Big(T+E(W_G^{H3})+E(Y)\Big)}_{\text{send AAK queries}} + \underbrace{E(Y)+\tau_H+E(W_G^{H3})+E(Y)}_{\text{send and receive successful AAK}}\Big]. \quad (54)$$

The delay to reliably deliver a certain percentage of data packets $E(S_\gamma^{H3})$ is obtained analogous to protocol (H1).

G. NAK and AAK-based Protocol (H4)

Protocol (H4) is basically (H2) with additional AAKs. In a NAK-based protocol, such AAKs are only reasonable if they are sent periodically rather than after every data packet transmission. Therefore, we assume in protocol (H4) that AAKs have a processing time of Z, which can be set to a proportionate value of Y. All delay components are shown in Figure 4.

G.1 Mean Waiting Times at the Sender (Root Node)

At the sender we have the following packet flows. The data packet flow λ_t^S, NAK and retransmission flow λ_r^S and additional NAK flow λ_n^S are analogous to protocol (H2). The AAK query flow λ_q^S has rate λL_{aaq}^{H4} (see [12] for L_{aaq}^{H4}). The AAK flow λ_{aa}^S has rate λB, since the sender receives from every child node exactly one AAK. Missing AAKs are queried with unicast from the nodes concerned. The load on the sender is:

$$\varrho_S^{H4} = (\lambda_t^S+\lambda_r^S)E(X)+(\lambda_r^S+\lambda_n^S)E(Y)+(\lambda_q^S+\lambda_{aa}^S)E(Z). \quad (55)$$

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

$$E(W_S^{H4}) = \frac{\lambda_t^S E(X^2)+\lambda_r^S\Big(E(X^2)+E(Y^2)+2E(X)E(Y)\Big)}{2(1-\varrho_S^{H4})} + \frac{\lambda_n^S E(Y^2)(\lambda_{aa}^S+\lambda_q^S)E(Z^2)}{2(1-\varrho_S^{H4})}. \quad (56)$$

G.2 Mean Waiting Times at a Receiver (Leaf Node)

The data packet flow λ^R and NAK flows $\lambda_{n,g}^R$, $\lambda_{n,r}^R$ are analogous to protocol (H2). The AAK query flow λ_q^R has rate $\lambda\widetilde{L}_{aaq}^{H4}$ (see [12] for $\widetilde{L}_{aaq}^{H4}$). The load on a receiver is:

$$\varrho_R^{H4} = \lambda^R E(X)+(\lambda_{n,g}^R+\lambda_{n,r}^R)E(Y)+\lambda_q^R E(Z+Z). \quad (57)$$

The mean waiting time of a packet at a receiver is:

$$E(W_R^{H4}) = \frac{\lambda^R E(X^2)+(\lambda_{n,g}^R+\lambda_{n,r}^R)E(Y^2)}{2(1-\varrho_R^{H4})} + \frac{\lambda_q^{H4}E((Z+Z)^2)}{2(1-\varrho_R^{H4})}. \quad (58)$$

G.3 Mean Waiting Times at a Group Leader (Inner Node)

The load on an inner node is the sender's load without the initial transmission and receiver's load.

$$\varrho_G^{H4} = \lambda_r^S E(X+Y)+\lambda_n^S E(Y)+(\lambda_q^S+\lambda_{aa}^S)E(Z) + \lambda^R E(X)+(\lambda_{n,g}^R+\lambda_{n,r}^R)E(Y)+\lambda_q^R E(Z+Z). \quad (59)$$

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

$$E(W_G^{H4}) = \frac{\lambda_r^S E((X+Y)^2)+\lambda_n^S E(Y^2)+(\lambda_q^S+\lambda_{aa}^S)E(Z^2)}{2(1-\varrho_G^{H4})} + \frac{\lambda^R E(X^2)+(\lambda_{n,g}^R+\lambda_{n,r}^R)E(Y^2)+\lambda_q^R E((Z+Z)^2)}{2(1-\varrho_G^{H4})}. \quad (60)$$

G.4 Overall Delay Analysis of Protocol (H4)

The loss detection and loss recovery phase is analogous to protocol (H2). If no retransmission is necessary, the delay from the initial transmission $E(I)$ is:

$$E(I) = E(W_S^{H4})+E(X)+\tau+E(W_G^{H4})+E(X). \quad (61)$$

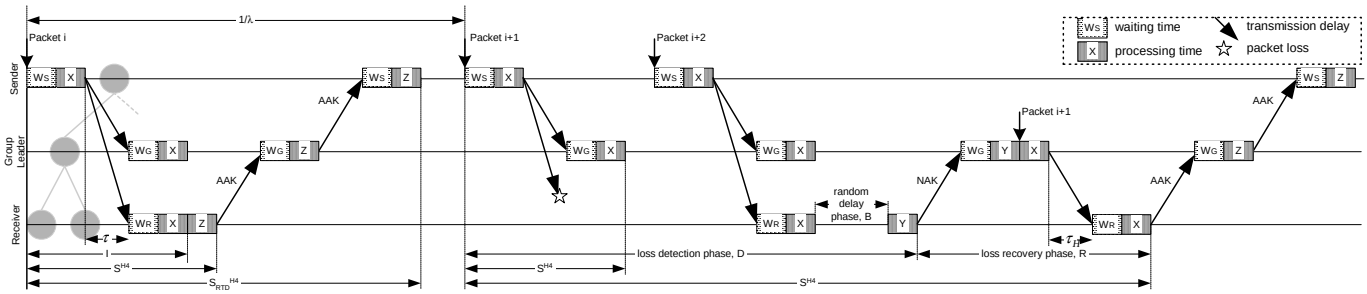


Fig. 4. Packet delivery delay for protocol (H4)

The overall delay is:

$$E(S_{\phi}^{H4}) = (1-q_D)E(I) + \Big[\sum_{i=1}^{\tilde{h}-2} q_D^i(1-q_D)\Big(E(I)+E(D^{H4})+iE(R^{H4})\Big)\Big] + q_D^{\tilde{h}-1}\Big(E(D^{H4})+(\tilde{h}-1)E(R^{H4})\Big). \quad (62)$$

Since protocol (H4) is a NAK-based protocol we have assumed that the AAKs are sent periodically rather than after every data packet transmission. However, for obtaining the round trip delay we assume a scenario in which a data packet is actually acknowledged by an AAK. The mean waiting time between sending a packet and receiving the last AAK at the sender is then given by:

$$E(S_{RTD}^{H4}) = (1-q_D)^R E(I) + \underbrace{\Big(1-(1-q_D)^R\Big)\Big(E(D^{H4})+E(R^{H4})\Big)}_{\text{sending transmissions}} + (h-1)\Big[\underbrace{E(\tilde{L}_{aaq}^{H4})\Big(T+E(W_G^{H4})+E(Y)\Big)}_{\text{AAK queries}} + \underbrace{E(Y)+\tau_H+E(W_G^{H4})+E(Y)}_{\text{send and receive successful AAK}}\Big]. \quad (63)$$

The delay to reliably deliver a certain percentage of data packets $E(S_{\gamma}^{H4})$ is obtained analogous to protocol (H2).

H. An Extension for Spatially Dependent Losses

In this section we will discuss our assumed system model of independent losses and extend it to spatially correlated losses. So far, we have assumed in the analysis that losses at different nodes are temporarily and spatially independent events. In fact, since receivers share parts of the multicast routing tree, this does not hold in real networks. In [19] and [20] the temporal and spatial loss correlation in the Internet and MBone is studied in detail. They concluded from measurements that the timescale for temporal loss correlation is 1 second or less. Beyond this timescale, what happens to a packet is not connected to the behaviour of a former sent packet. Even within the correlation timescale, most losses were solitary losses. With respect to spatial losses, they found only small correlation among the multicast sites except for the loss due to the link next to the source. The backbone loss was found to be very low. We can conclude from these observations that our assumption of temporal independent losses introduces only a negligible inaccuracy into our model. With respect to spatial losses, though, an inaccuracy we introduce is the spatial correlation due to loss on the first link from the sender to the backbone.

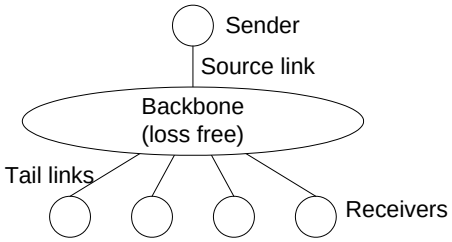


Fig. 5. System Model

Now we present a modified system model to consider these spatial correlation. Figure 5 shows our assumed system model. The sender is connected with an error-prone link to the backbone. An error on this link will be seen by all receivers. The backbone is considered as error free, according to the observations discussed above. Finally, each receiver is connected to the backbone with an error-prone link. Errors on this tail links are assumed to be mutually independent. Our model is similar to [9] and [10].

As the end-to-end loss probability perceived by a random receiver continues to be q_D, we assume that this probability is equally split between the source link loss $q_D\prime$ and tail link loss $q_D\prime$, that are:

$$q_D\prime = 1-\sqrt{1-q_D}. \quad (64)$$

The expected total number of necessary transmissions $E(M^{H1})$ for protocol (H1) to receive the data packet correctly at all receivers is now the sum of the retransmits due to loss on the source link and retransmits due to loss on the tail link or ACK loss and the initial transmission:

$$E(M^{H1}) = E(M_S^{H1}) + E(M_T^{H1}) + 1. \quad (65)$$

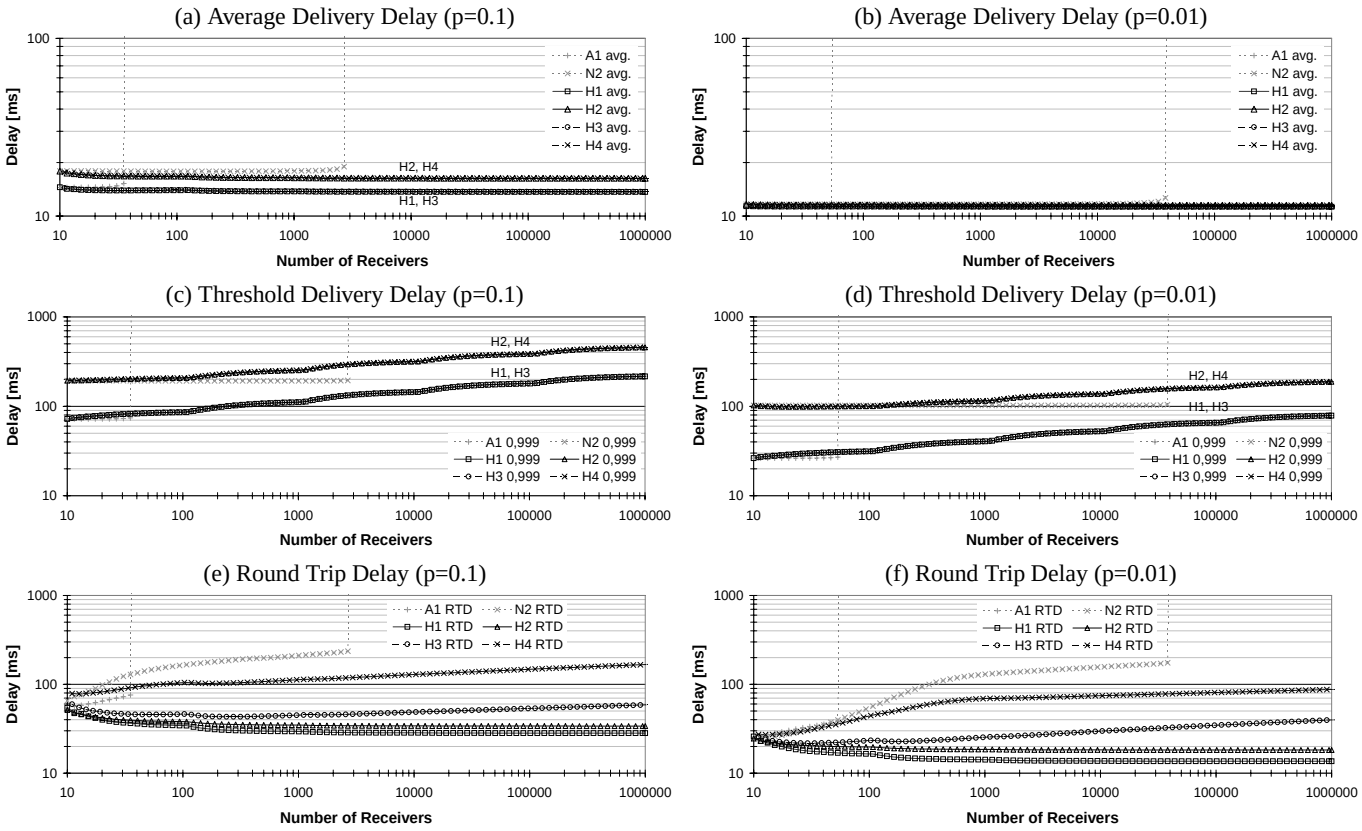


Fig. 6. Delays with respect to the number of receivers

The number of retransmissions due to loss on the source link is:

$$E(M_S^{H1}) = \frac{1}{1-q_D\prime} - 1, \tag{66}$$

and the number of retransmissions due to loss on the tail link or ACK loss:

$$E(M_T^{H1}) = \sum_{i=1}^{B} \binom{B}{i} (-1)^{i+1} \frac{1}{1-(q_D\prime + (1-q_D\prime)p_A)^i} - 1. \tag{67}$$

The number of retransmissions for the other protocol classes can be changed accordingly. Besides a modified number of retransmissions we must change the round trip delay equations for protocols (H2) and (H4) to:

$$\begin{aligned} E(S_{RTD}^{H2}) &= \left[1 - \left(q_D\prime + (1-q_D\prime)(1-(1-q_D\prime)^B)\right)\right] E(I) \\ &+ \left(q_D\prime + (1-q_D\prime)(1-(1-q_D\prime)^B)\right)\left(E(D^{H2}) + E(R^{H2})\right) \\ &+ E(Y) + E(W_G^{H2}) + \tau_H + E(W_S^{H2}) + E(Y), \end{aligned} \tag{68}$$

and analogous for protocol (H4). The numerical results in the following section are based on this modified system model.

V. Numerical Results

We examine the expected delays of the analyzed protocols by means of some numerical examples. According to measurements in [21] we have chosen the delay $X = 500\mu s$ for data packets and $Y = 100\mu s$ for control packets. Analogous to [13], 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 trebled propagation delay, i.e. $T_S = T_R = 30ms$. For the NAK suppression time we have assumed $B = 30ms$. A discussion of reasonable values for the NAK suppression time can be found in [13]. We have chosen a lower suppression time due to our smaller local group sizes.

Figure 6 shows the expected average delay, the threshold delay to reliably deliver all packets with probability 0.999 and the round trip delay for all considered protocol classes with varying number of receivers. Additionally, two non-hierarchical approaches (A1) and (N2) are included to compare their scalability. Protocol (A1) is an ACK-based protocol similar to (H1) and (N2) is a NAK-based protocol with NAK suppression similar to (H2). Protocol (A1) and (N2) are explained in more detail in [12]. Data and control packet loss probability is 0.1 (left side) and 0.01 (right side). The data rate is $\lambda = 0.1\frac{1}{ms}$. Figure 7 plots the delays for 5000 receivers with varying sending rate λ.

In contrast to non-hierarchical approaches, all tree-

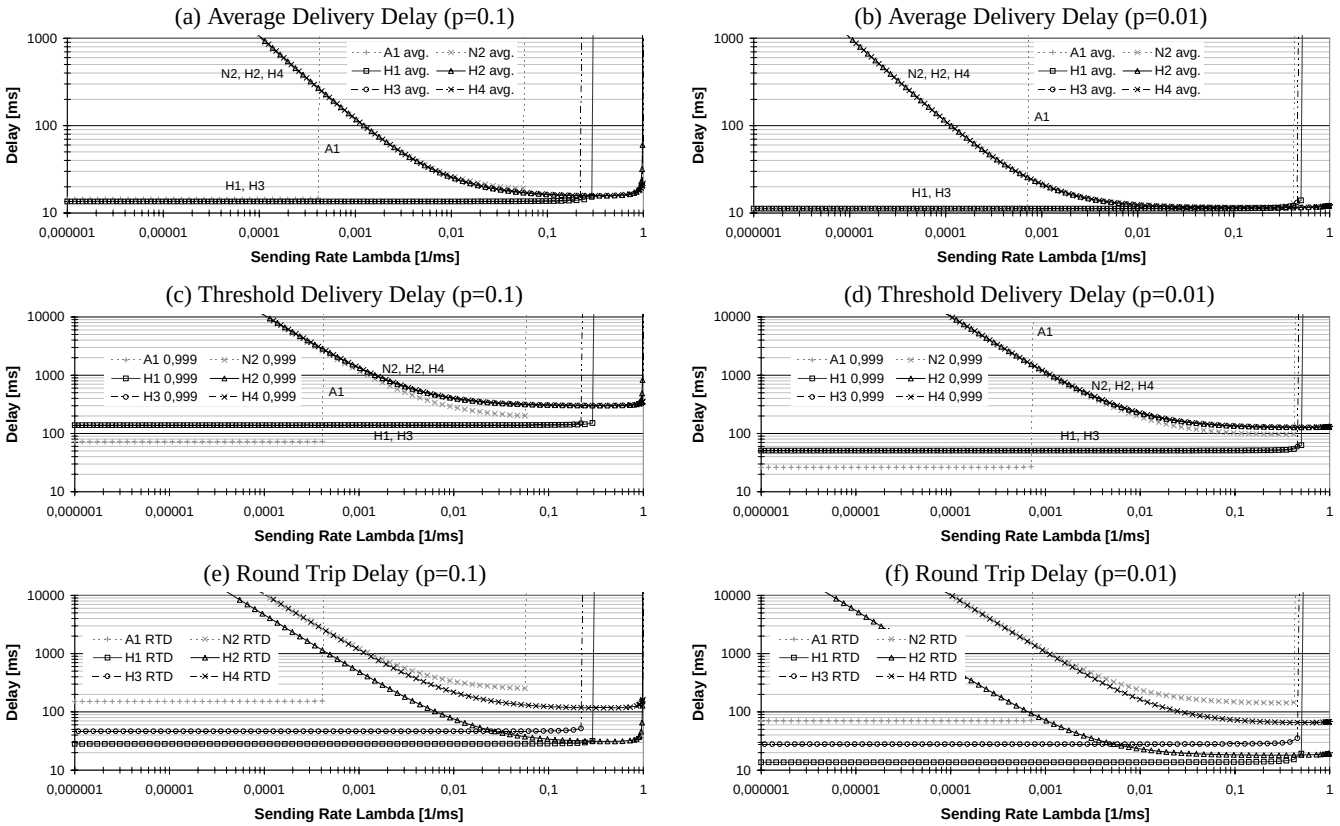


Fig. 7. Delays with respect to the sending rate

based protocols provide scalability for all group sizes and in most cases, they result in lower delays even within the scalability range of (A1) and (N2).

While the low packet loss probability of Figure 6.b results in almost identical average delivery delays close to the propagation delay of the network, Figure 6.a allows a more detailed view. The NAK protocols result in higher average delivery delays as well as threshold and round trip delays. 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, which results in higher delays for retransmissions.

Figure 6.c and 6.d plots the threshold delivery delay to deliver all messages with probability 0.999. While 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, the threshold delivery delay 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, ACK-based protocols have a significantly lower threshold delay. 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 VI that the threshold delay for $\gamma = 0.999$ is a reasonable approximation for the delay to deliver all packets correctly.

Figure 6.e and 6.f shows the expected average round trip delay of the analyzed protocol classes. After this time, the sender can remove the data packet from memory. 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 $\frac{bufferspace}{rtd}$ [2]. Recall that the round trip delay for protocol (H1) and (H2) is assumed to encompass only the direct child nodes of the sender while the round trip delay for protocol (H3) and (H4) encompass the whole control tree. The round trip delay for (H1) and (H2) decreases with the tree height. This is caused by our assumption that an increased tree height results in lower delays between the sender and its direct child nodes, since all members of a local group are nearby. As the round trip delay with AAK protocols considers all local groups from a leaf node to the sender, for (H3) and (H4) it increases with the tree height.

The effect of the receiver-initiated loss detection can be

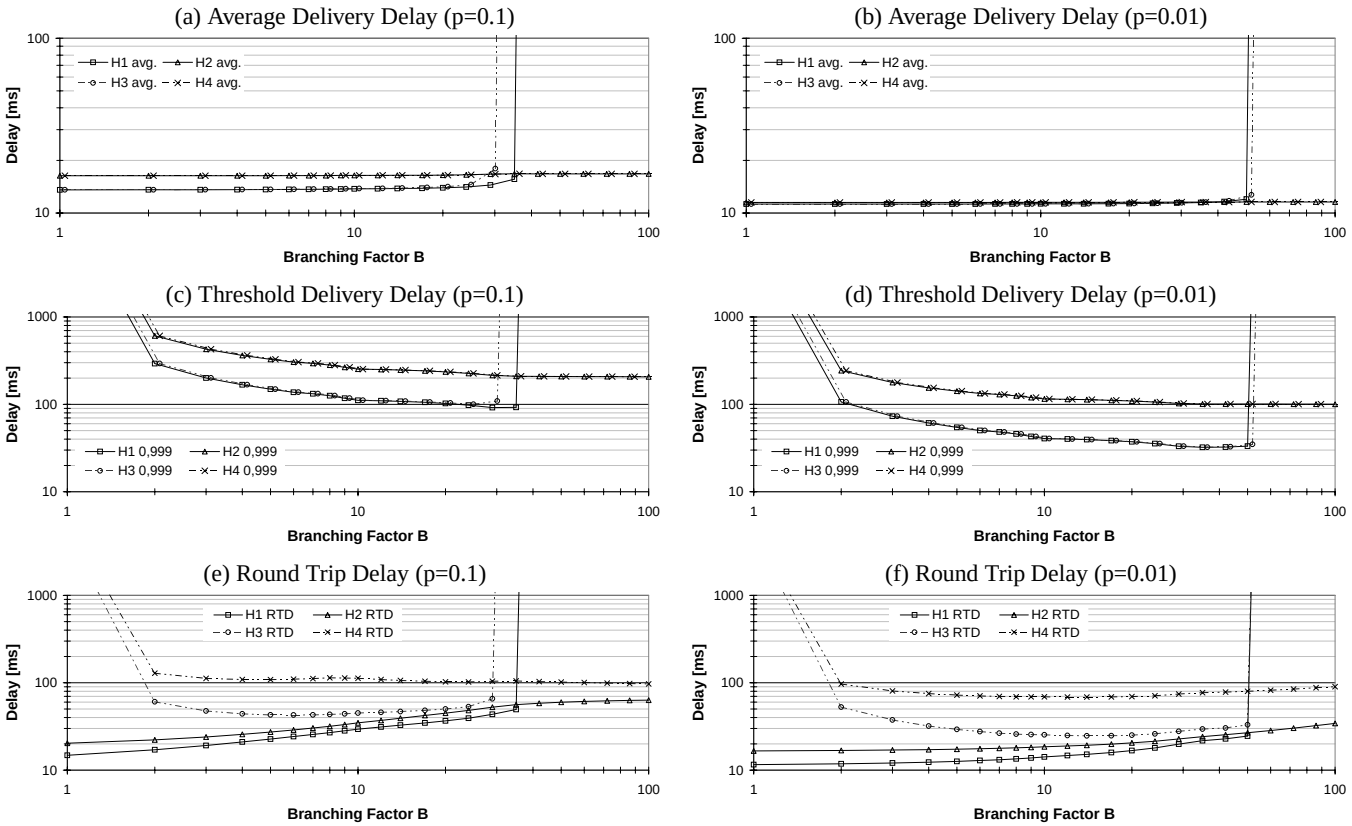


Fig. 8. Delays with respect to the branching factor

studied in detail in Figure 7 with varying sending rates. For low sending rates this loss detection delay is the dominant delay. For example, with sending rate $\lambda = 0.0001\frac{1}{ms}$ it takes about 10s ($1/\lambda$) to detect packet loss. You can see in Figure 7.c and 7.d that the threshold delay is indeed about 10s, almost independent of the loss rate. 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 decreased by about the factor 100. As packet loss is detected by the sender for protocol (H1) and (H3), their delays are independent of the sending rate. If the sending rate exceeds a certain limit ($\lambda = 0.5\frac{1}{ms}$ for packet loss rate 0.01 and $\lambda = 0.2\frac{1}{ms}$ for packet loss rate 0.1), though, the sender is overwhelmed with ACK messages. For protocol (H2) and (H4) this limit is higher due to less control messages ($\lambda = 1\frac{1}{ms}$ for packet loss rate 0.1).

Figure 8 shows the delay results for varying branching factors with 1000 receivers and sending rate $\lambda = 0.1\frac{1}{ms}$. Within the scalability range of a protocol class, the average delivery delay is hardly influenced by the branching factor. Since an ACK-based protocol is only scalable for up to 50 nodes with the given sending rate, a branching factor of more than 50 nodes cannot be supported by (H1) and (H3). The threshold delay and round trip delay provides more interesting results. For all protocol classes, the threshold delay decreases with increasing branching factor until a protocol class is saturated by the feedback implosion. This is caused by the decreased tree height and therefore faster retransmissions in the worst case. With respect to round trip delay, there is a minimum delay at a branching factor of 10 to 20 for protocol (H3) and (H4). Since (H1) and (H2) uses normal ACKs rather than hierarchical ones, the lowest round trip delay is achieved with a small number of child nodes.

VI. Comparison with Simulation Results

To assess the analytical results we have implemented the RMTP [3] and TMTP [4] reliable multicast protocols in the NS2 [5] network simulator environment. Recall that RMTP is a sender-initiated protocol and TMTP is a receiver-initiated protocol with NAK suppression.

In contrast to the specification of RMTP we have implemented no subcast mechanism, as this is not available with general routers. Instead we used TTL-limited multicast to send retransmissions. A further significant difference is

that we send acknowledgments as soon as a data packet is received rather than periodically. Besides normal ACKs we have additionally implemented aggregated ACKs. As a consequence of the aggregated ACKs, this protocol is of class (H3).

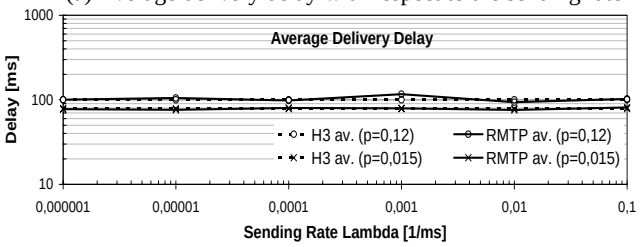
In contrast to the specification of TMTP we have implemented AAKs rather than so-called early ACKs. TMTP uses early ACKs to advance the flow control window. An early ACK is sent after the corresponding data packet has been received. This means, a group leader does not need to wait for ACKs from all its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

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

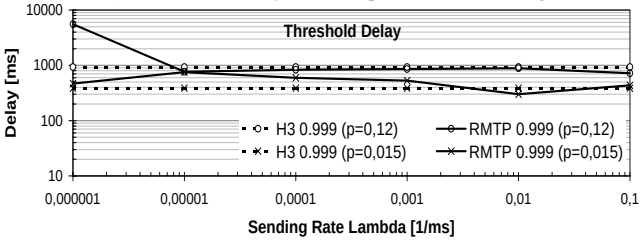
For our simulations we have used two networks generated by Tiers [22] with 250 and 1000 nodes. All nodes in the network use DVMRP [23] routing. To simulate message loss, each link in the network is configured with probability 0.02% or 0.002% respectively for message loss. We have measured an average end-to-end message loss probability for data packets of about 12% or 1.5% respectively. The average propagation delay was measured to be about 70ms for the 250 node network and 130ms for the 1000 node network. While we have varied the sending rate for our simulations, the flow control window size was always 10.

First we take a look at the RMTP results. Figure 9 shows the average delay, round trip delay and threshold delay for varying sending rates, varying number of receivers and varying branching factors. The solid lines display the results for the simulation whereas the dotted lines display the numerical results from our analysis. For Figure 9.a-9.c the number of receivers is 100 in a network consisting of 250 nodes and a branching factor of 10. For Figure 9.d and 9.e the sending rate is $0.001\frac{1}{ms}$ and the network consists of 1000 nodes. For Figure 9.d the branching factor is also 10. The results for the varying branching factor is shown

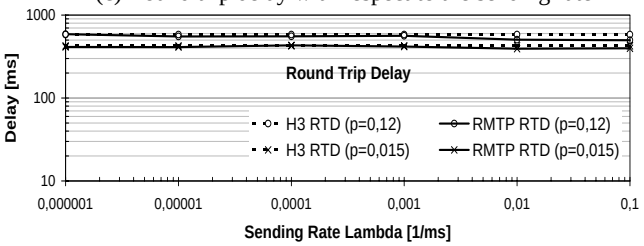
(a) Average delivery delay with respect to the sending rate



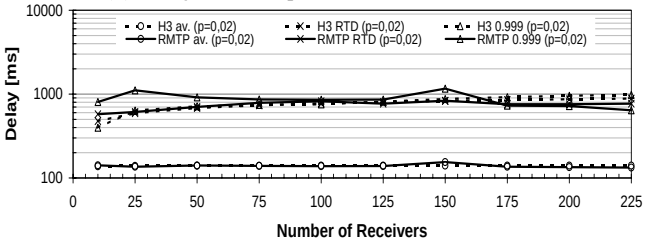
(b) Threshold delay with respect to the sending rate



(c) Round trip delay with respect to the sending rate



(d) Delays with respect to the number of receivers



(e) Delays with respect to the branching factor

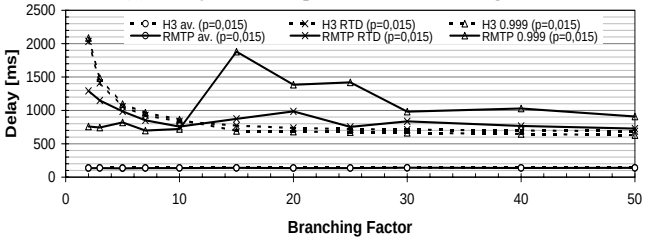


Fig. 9. Analytical vs. simulated delays of RMTP

with 200 receivers.

As the results in figure 9 indicate, the delay of RMTP is mostly independent of the sending rate and group size, for the analytical as well as simulation results. If we take a closer look on the average delay results we can see that even the absolute delays are predicted very precisely by our analysis. In fact, the average delays from the simulation differs from the analytical delays only by less than 5%. The threshold delay results show a significantly higher deviation. The analytical results for the threshold delay in Figure 9.b are made with probability 0.999 for correct

delivery of all packets. The measured simulation results show the delay to correctly deliver all packets. Some of this deviation results from fluctuating message loss probabilities which are only on average 12% or 1.5%, respectively. The results for the round trip delay in Figure 9.c are similar to average delay very precisely predicted by the analysis.

Figure 9.d shows the delay results for a varying number of receivers. Here we can see that the average delay within this group size range is indeed almost independent of the group size, while the round trip delay and threshold delay increases with the tree height.

Our last results for RMTP in Figure 9.e show the influence of a varying branching factor. As predicted by the analytical results, average delay is hardly influenced by the branching factor. For the round trip delay we see that both, the analytical as well as simulation results show a significant decrease with increasing group size. Unfortunately, for the threshold delay the simulation results are not stable, showing a peak that was not predicted by the analytical results, which may result from high message loss at important links, e.g. near the source.

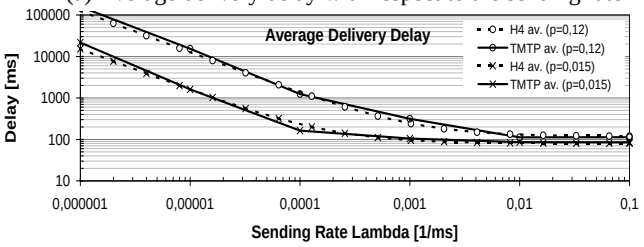
Now we want to present the TMTP simulation results. By comparing the average delay results in Figure 10.a we can again see that the analysis predicts very exactly the simulation results for varying sending rates as well as for different packet loss probabilities.

We can see that the results for threshold probability 0.999 and round trip delay show similar behaviour compared to the measured results, however, deviate in their exact absolute value. Note that some of this deviation results from the window based sending scheme. If no further data packets can be sent due to missing aggregated 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. A second reason are the fluctuating message loss probabilities.

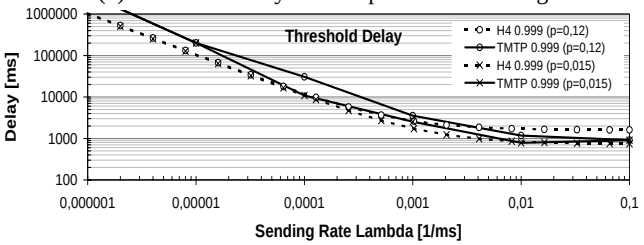
Figure 10.d shows the delay results for a varying number of receivers. Again, the average delivery delay is indeed almost independent of the group size. However, round trip delay is influenced by increased group size, since this results in larger height of the control tree and therefore more forwarding steps in the control tree.

The influence of a varying branching factor on TMTP's results are depicted in Figure 10.e. As predicted by the analytical results, average delay is hardly influenced by the branching factor. With respect to the round trip delay

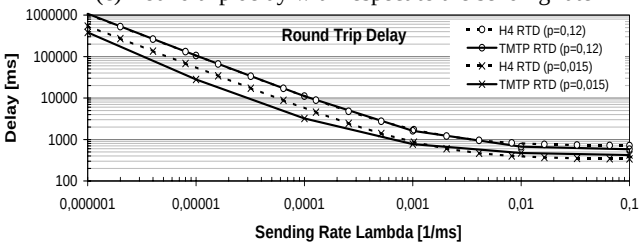
(a) Average delivery delay with respect to the sending rate



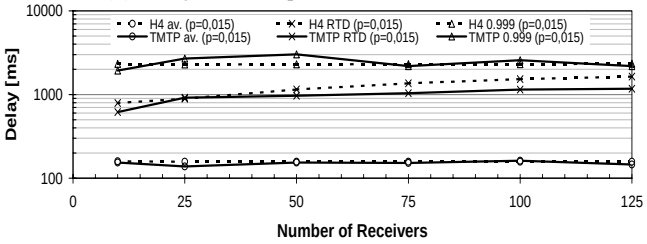
(b) Threshold delay with respect to the sending rate



(c) Round trip delay with respect to the sending rate



(d) Delays with respect to the number of receivers



(e) Delays with respect to the branching factor

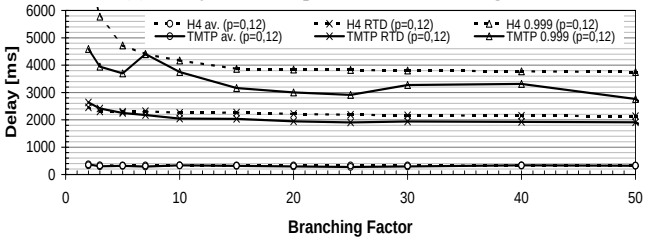


Fig. 10. Analytical vs. simulated delays of TMTP

we can see a moderate decrease between 2 and 10 child nodes per group leader in the analytical as well as simulation results. A significant decrease can be observed for the threshold delay. Although the absolute delays are fluctuating and lower as the analysis predicts, both show similar delay decrease with increased branching factor.

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 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, varying sending rates and varying branching factors are closely correlated.

VII. Summary

We have presented a comparative delay analysis of tree-based reliable multicast protocols. 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 all tree-based protocols provide low delays and good scalability compared to non-hierarchical approaches. From the four considered protocol classes, NAK-based protocols achieve the best scalability but ACK-based protocols achieve the lowest delays. With respect to protocols with aggregated ACKs (AAKs), which provide reliability even in case of node failures, we can conclude that the increase in delay compared to protocols without AAKs is negligible.

It was an important goal of our analysis to be of practical relevance rather than being of only theoretical nature. Therefore, we have compared the analytical results with RMTP and TMTP simulations. Both show identical behaviour with varying number of receivers, transmission rates, loss probabilities and branching factors. 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] 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.

[4] 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.

[5] 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.

[6] 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.

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

[8] 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.

[9] 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.

[10] 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.

[11] G. Poo and A. Goscinski, "Performance comparison of sender-based and receiver-based reliable multicast protocols," *Computer Communications*, vol. 21, no. 7, pp. 597–605, June 1998.

[12] 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.

[13] 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.

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

[15] 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.

[16] S. Ramakrishnan and B. N. Jain, "A negative acknowledgement with periodic polling protocol for multicast over lans," in *Proceedings of IEEE INFOCOM*, Mar. 1987, pp. 502–511.

[17] B. Whetten and G. Taskale, "An overview of the reliable multicast transport protocol II," *IEEE Network*, vol. 14, no. 1, pp. 37–47, Feb. 2000.

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

[19] M. Yajnik, J. Kurose, and D. Towsley, "Packet loss correlation in the mbone multicast network," in *Proceedings of IEEE Global Internet*, London, UK, Nov. 1996, pp. 94–99.

[20] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and modelling of the temporal dependence in packet loss," in *Proceedings of IEEE INFOCOM'99*, New York, 1999, pp. 345–352.

[21] 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.

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

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