

Tight Bounds for FEC-Based Reliable Multicast ^{*†}

Hagit Attiya[‡]

Hadas Shachnai^{§¶}

Department of Computer Science
The Technion, Haifa 32000, Israel

Abstract

We study the problem of reliable multicast of a message to a set of receivers over a network that uses best effort transmission (e.g., ATM), based on *forward error correcting codes (FEC)*. We prove lower bounds on the time and message complexities of *any* algorithm for the problem. We present a randomized algorithm which guarantees the reliable multicast of a packet of size m to n receivers within $O(\log n)$ steps, and whose expected message complexity is $O(\log m)$. Its storage complexity is linear in the size of the original packet. The average message and time complexities of the algorithm are within a constant multiplicative factor of the lower bounds.

1 Introduction

Multicast allows a single *sender* node to send a packet to a set of *receiver* nodes. In *reliable multicast*, the delivery of packets to all receivers is guaranteed. This is a well-known methodology for constructing distributed systems (see e.g., in [4, 7, 14, 18, 22, 23]).

Formally, the reliable multicast problem requires a single node, P_s , to send a packet M to n receivers, P_1, \dots, P_n . The receivers are located in terminal nodes of a network (see Figure 1).

^{*}The extended abstract of this paper appeared in *Proceedings of the 1st International Conference On Principles Of Distributed Systems (OPODIS), 1997*.

[†]This work is supported by grant number 854-6195 from the Israeli Ministry of Science.

[‡]Email: hagit@cs.technion.ac.il.

[§]Corresponding author: Email: hadas@cs.technion.ac.il. Tel. +972-4-829-4359. Fax: +972-4-822-1128.

[¶]Part of this work was done while the author was on a leave in Bell Laboratories, Lucent Technologies, 600 Mountain Ave., Murray Hill, NJ 07974.

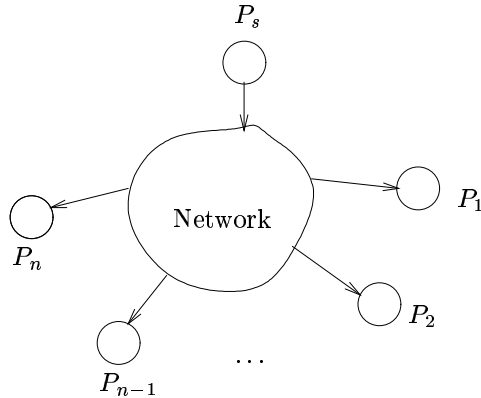


Figure 1: Multicast in a general network.

The packet M is too big to be sent in one piece, so the sender must break it into smaller pieces. The network is unreliable, and some pieces may be lost; different pieces may be lost on the way to different receivers. To re-assemble the packet, a receiver should have all the pieces.

When some pieces are lost and do not arrive at some receiver node, the sender can either retransmit all pieces to all receivers, or have each receiver notify which pieces it misses, and send these particular pieces. Under the first option, it is possible that many pieces are wasted; the second option requires considerable receiver-to-sender traffic.

A different scheme is based on *forward error correcting (FEC)* codes (see, e.g., [13, Ch. 11]). In this scheme, the sender produces $R > m$ pieces from the packet M , such that *any* subset of m pieces suffices to reconstruct M ; pieces are *symmetric* and any subset of size m can be used. Thus, if two receivers P_i and P_j each miss exactly one cell, c_i and c_j , then a single cell c' not already held by either of them suffices to reconstruct M , even if $c_i \neq c_j$.

The benefits of this scheme are twofold. First, the same pieces can be used to compensate for the cell deficits of different receivers. Second, the sender can initially send a surplus of pieces, depending on the estimated reliability of the network (the probability of pieces being lost).

Under this scheme, all the sender has to know is the maximal number of cells missing for some receiver; to achieve this, receivers send negative acknowledgments (*NACK*), with the number of pieces they miss, to the sender.

We study in this paper the time and message complexities of algorithms for managing this scheme. Note that while it is easy to optimize on one of these measures, it is hard to optimize on both. Consider, for example, an algorithm in which, at time t , any receiver node i sends

a NACK message to the sender, containing the number of cells it is missing. The sender then multicasts a set of cells which covers the maximum deficit among the receivers. This algorithm will terminate after $O(1)$ steps. However, the number of NACK messages sent is $\Omega(n)$. At the other extreme, consider an algorithm in which every receiver node still missing cells sends a NACK message, specifying the number of cells it needs, with some fixed probability p , $0 < p \leq 1/n$. The algorithm optimizes on message complexity, i.e. the expected number of messages sent is $O(\log m)$.¹ However, it terminates within $\Omega(n)$ steps, the expected number of trials until a receiver with missing cells sends a NACK message.

Our objective is to develop algorithms that require a minimal number of steps and the smallest number of NACK messages, before each of the receivers has all necessary m cells.

1.1 Main Results

Our study encompasses general unreliable networks. We present the first lower bounds on the time and message complexity of *any* FEC-based scheme for reliable multicast, together with matching upper bounds. In particular,

- We show (in Section 3) that if an algorithm starts with maximal cell deficit r , retransmits at most μr cells, and guarantees that all receivers obtain the original packet with probability at least $1 - \epsilon$, for some small $\epsilon > 0$, then the expected number of NACK messages is at least $(1 - \epsilon) \log_{\mu} r$. Moreover, if the algorithm is message optimal, then its expected time complexity is $\Omega(\log n)$.²
- We present a randomized algorithm which terminates within $O(\log n)$ steps, and whose expected message complexity is $O(\log m)$. The algorithm is first analyzed for the case where NACKs and retransmissions are reliable (Section 4.1) and then for the general case where *any* transmission, of a cell or a NACK message, is unreliable (in Section 4.2).

The proofs of our results do not rely on any properties of the network topology. Thus, all the results in this paper hold for a general network. The multicast model that we adopt here is primarily motivated from the rules set for multicast in ATM networks (see Appendix A); however, our results apply to any network that provides best-effort unreliable transport (e.g., UDP in IP networks).

¹This follows from a general theorem that we prove below (Theorem 6).

²Unless specified otherwise, the logarithms in this paper are to base 2.

In developing the algorithm in Section 4, we use two assumptions on the knowledge of the network parameters by the sender and receivers: (i) each of the receivers knows n , the overall size of the set of receivers, and (ii) the sender knows that maximum loss probability of a cell/NACK message. This knowledge can be acquired by preceding our algorithm with a preprocessing step, that will be used to update the nodes on network parameters. We elaborate on that in Section 4.

1.2 Related Work

There has been extensive work on the use of FEC techniques for reliable multicast in networks with unreliable transmissions. Some reliable multicast schemes, e.g., [3, 5, 10], rely solely on redundancy in the FEC encoding, while others, e.g., [6, 8, 12, 16, 20, 21] combine it with NACKs from receivers to sender. The performance of most of the proposed schemes was evaluated through simulation study [5, 8, 12, 20, 21]. Several works use probabilistic assumptions on parameters of the network to derive analytic results [12, 15].

Nonnenmacher and Biersack [15] deal with mechanisms for avoiding feedback implosion in large networks where receivers send (NACK or other) feedback messages to a single sender. The proposed mechanisms are based on probabilistic feedbacks (using several distributions). The paper gives expressions for the expected feedback delay and the expected number of feedback messages received by the sender; these measures are then evaluated for large number of (up to 10^6) receivers using simulation. In TCP Boston [2], Rabin's *information dispersal algorithm* [17] was used to implement reliable *point to point* communication. Since the communication is point-to-point, there is no concern for minimizing receiver-to-sender traffic. The multicast situation poses non-trivial algorithmic concerns, which are the focus of our work.

Some of the previous papers (e.g. [17, 19]) addressed theoretical and practical issues related to the selection of FEC schemes to be used for reliable multicast. Our results hold with any choice of the FEC scheme, and thus, provide a more general view of this problem.

2 Preliminaries

We assume a one-to-many communication situation, where a single sender, P_s , has to transmit packets over a network to n receivers, P_1, \dots, P_n . Packets are broken into *cells*. In the reliable multicast problem, we refer to the retransmission of a single packet by the sender to a set of receiver nodes, until all receivers can reconstruct the packet from the cells they obtained.

Any multicast of a cell over the network may result in a *loss*, meaning that the cell is not received by some of the receivers. Let $q_L^c(i)$ denote the loss probability of a cell transmitted to receiver node i , $1 \leq i \leq n$. We define the *sender to receiver reliability parameter* of the network as $q_L^c = \max_{1 \leq i \leq n} q_L^c(i)$. In addition, a NACK message sent from receiver node i may be lost with probability $q_L^N(i)$. We denote the *receiver to sender reliability parameter* of the network by $q_L^N = \max_{1 \leq i \leq n} q_L^N(i)$.

Using FEC with parameters $R, m > 1$, each packet is broken to R pieces, such that the complete packet can be reconstructed from any subset of m pieces [13]. Thus, each receiver needs to obtain at least m different pieces, but it does not matter which. For any $t \geq 1$, let $m_i(t)$ denote the number of *missing cells* in receiver node i at time t . Let m_i be the initial *cell deficit* of receiver node i ; $m_i(0) = m_i$, for any i , $1 \leq i \leq n$.

The sender retransmits cells over the network in response to NACK messages sent by receivers. We assume communication is synchronous, and each phase is either a retransmission phase (from sender to receivers) or a NACK phase (from receivers to sender). That is, in phase 0 the sender transmits the packet, and thereafter, for any $t \geq 1$, the t -th receiver-to-sender phase is followed by the t -th sender-to-receiver phase. Receivers do not communicate with each other, and receiver node i has no information on the values of m_j for $j \neq i$. Receiver node i terminates at the first time t in which $m_i(t) = 0$; the algorithm terminates when all receiver nodes terminate.

The *running time* of an algorithm, denoted by T , is the number of phases (or rounds) until the algorithm terminates. Each phase may consist of a predetermined number of time units, during which the communication is done. vice versa.

We also measure the *message complexity*—the number of NACK messages sent during the execution of the algorithm, denoted by COM_N . Since randomization is employed, our goal is to minimize $E[T]$ and $E[COM_N]$, where averaging is done over the coin flips of the algorithm.

Finally, we measure the *storage overhead* of the algorithm. When FEC is used for breaking the original packet into R cells, extra cells are stored to be used by the sender during retransmission phases. The algorithm presented in this paper is space efficient, as it generates at most $3/2m$ cells.

3 Lower Bounds

We first prove lower bounds for randomized algorithms for reliable multicast in FEC-based schemes. The lower bounds are proved under the assumption that NACK transmission and cell retransmission are reliable. Clearly, they also hold when this assumption is removed.

3.1 Message Complexity

In this section, we derive a lower bound on the number of NACK messages sent by any distributed randomized algorithm for reliable multicast. Let $0 \leq r \leq m$ denote the maximal cell deficit after the original transmission. Then, $0 \leq r \leq m$.

For convenience, we associate the t -th phase with *time* t (where the type of a phase can be understood from the context). Let $N_Mes(t)$ denote the set of NACK messages arriving to the sender at time t . The sender responds to the messages in $N_Mes(t)$ by sending a certain amount of cells in the next phase (i.e., the t -th sender-to-receivers phase).

Definition 1 For $t \geq 1$, let $R_{\mathcal{A}}(\max(t))$ denote the number of cells retransmitted by the sender in response to the messages in $N_Mes(t)$, in which the maximal deficit is $\max(t)$. Then the redundancy ratio of \mathcal{A} is given by

$$\mu_{\mathcal{A}} \equiv \max_{t \geq 1, 1 \leq \max(t) \leq m} \frac{R_{\mathcal{A}}(\max(t))}{\max(t)} .$$

Note that for obtaining the lower bounds we may assume that $\mu_{\mathcal{A}} > 1$. Indeed, by this we only decrease the number of rounds and NACK messages required until the sender has completely covered the maximal cell deficit.

Suppose that, during the execution of an algorithm \mathcal{A} , the sender gets k sets of messages $N_Mes(t)$, $1 \leq t \leq k$. If the redundancy ratio of \mathcal{A} is $\mu_{\mathcal{A}}$, then the sender responds to the messages in $N_Mes(t)$ by retransmission of at most $\mu_{\mathcal{A}} \cdot \max(t)$ cells in the following phase.

The next theorem shows a lower bound of $\Omega(\log m)$ on the number of NACK messages sent by any randomized algorithm \mathcal{A} for reliable multicast.

Theorem 1 Fix r , $1 \leq r \leq m$, and ϵ , $0 < \epsilon < 1$, and let \mathcal{A} be a randomized distributed algorithm with redundancy ratio $\mu_{\mathcal{A}} > 1$. If \mathcal{A} guarantees that all receivers obtain the original

packet with probability at least $1 - \epsilon$, then the expected number of NACK messages sent by \mathcal{A} satisfies

$$E[COM_N] \geq (1 - \epsilon) \log_{\mu_{\mathcal{A}}} r . \quad (1)$$

Proof: Observe that in order to guarantee the retransmission of r cells by the sender, at least one receiver i with $m_i \geq r/\mu_{\mathcal{A}}$ has to send a NACK message with probability $1 - \epsilon$.

We partition the receiver nodes into $H = \lceil \log_{\mu_{\mathcal{A}}} r \rceil$ subsets, S_1, \dots, S_H , such that receiver node i belongs to S_j if $\mu_{\mathcal{A}}^{j-1} \leq m_i < \mu_{\mathcal{A}}^j$, for some j , $1 \leq j < H$.

It suffices to have a single member of S_H send a NACK message. We argue however, that at least one NACK message should be sent by some node in S_j with probability higher than $1 - \epsilon$, for *any* j , $1 \leq j \leq H$; otherwise, consider the execution in which $H = j$. Since r is unknown and the receiver nodes do not communicate with each other, the nodes in S_j obtain their missing cells with probability smaller than $1 - \epsilon$.

Let $E[COM_N(j)]$ be the expected number of NACK messages sent by nodes in S_j . The total number of NACK messages sent by \mathcal{A} satisfies

$$\begin{aligned} E[COM_N] &= \sum_{j=1}^H E[COM_N(j)] \\ &\geq \sum_{j=1}^H \text{Prob}(\text{a node in } S_j \text{ sends NACK}) \quad (\text{by Markov inequality [9]}) \\ &\geq \sum_{j=1}^H (1 - \epsilon) \end{aligned}$$

which yields inequality (1). □

3.2 Time Complexity

We now derive a lower bound on the expected number of phases required for the reliable multicast of a packet. Let $E[T^{\mathcal{A}}]$ denote the expected number of phases until a receiver with the largest deficit sends a NACK message for the first time. Since NACK transmission and cell retransmission are reliable, this would terminate the execution of the algorithm.

3.2.1 Uniform Algorithms

We start with the simpler case where the algorithm is uniform, i.e., all receiver nodes follow the same algorithm. In deriving the next result, we use the proof technique of Kushilevitz and Mansour [11], who showed a lower bound on the expected time for broadcast in radio networks.

Theorem 2 *Fix r , $1 \leq r \leq m$ and let \mathcal{A} be a uniform randomized distributed algorithm with redundancy ratio $\mu_{\mathcal{A}} > 1$. If \mathcal{A} sends $O(\log_{\mu_{\mathcal{A}}} r)$ NACK messages, then*

$$E[T^{\mathcal{A}}] = \Omega(\log n). \quad (2)$$

Proof: Let $H = \lceil \log_{\mu_{\mathcal{A}}} r \rceil$. As in the proof of Theorem 1, we partition the receiver nodes into H subsets, S_1, \dots, S_H , such that receiver node i belongs to S_j if $\mu_{\mathcal{A}}^{j-1} \leq m_i < \mu_{\mathcal{A}}^j$, for some j , $1 \leq j < H$. We prove that the expected number of phases until S_H sends the first NACK message is $\Omega(\log n)$.

To compute $E[T^{\mathcal{A}}]$, we average over all possible sizes of S_H . In particular, assume that $|S_H| = 2^l$, where l is chosen uniformly in the range $\{0, \dots, \log n\}$. Let $A_{s,l}$ be the event “The first success of S_H is in phase s ”, where $|S_H| = 2^l$. We denote by $q(s)$ the probability that a node sends NACK in phase s , then

$$\bar{q}(s-1) = 1 - (1 - q(1)) \cdot (1 - q(2)) \cdots (1 - q(s-1))$$

is the probability that node i in S_H sends at least one NACK in the first $(s-1)$ phases.

Claim 3 *If*

$$E[COM_N] = O(\log r) , \quad (3)$$

then there exists a constant $c \geq 1$, such that for any $s > 1$,

$$q(s) < c \cdot \bar{q}(s-1) . \quad (4)$$

Proof: For any subset S_j , $1 \leq j \leq H$, $E[COM_N(j)] \geq 1 - \epsilon$. By Equation (3), and since $H = \lceil \log_{\mu_{\mathcal{A}}} r \rceil$, this implies that each subset S_j sends $O(1)$ messages during the execution of the algorithm. Therefore, there exists $\delta > 1$ such that for any j , $1 \leq j \leq H$,

$$E[COM_N(j)] \leq \delta . \quad (5)$$

Let $E[COM_N(j, s)]$ be the expected number of NACK messages sent by S_j in phase s . Observe that for S_H

$$E[COM_N(H, s)] \geq (1 - \bar{q}(s-1))^{|S_H|} q(s) \cdot |S_H| . \quad (6)$$

Let $c = \delta e^2$, and assume, that there exists $s > 1$, such that $q(s) > c\bar{q}(s-1)$, then we show that the expected number of NACK messages sent by some subset S_H is larger than δ .

Let $\bar{q}(s-1) = \frac{1}{2^k}$, for some $k > 1$. (If $k \leq 1$ than $q(s) \leq 2\bar{q}(s-1)$ and we are done.) For $|S_H| = 2^k$ we have:

$$E[COM_N(H, s)] \geq (1 - \frac{1}{2^k})^{2^k} \frac{1}{2^k} \delta e^2 2^k > \delta .$$

which contradicts inequality (5). □

Given $s > 1$ and $0 \leq l \leq \log n$ let $p_{s,l} = \text{Prob}(A_{s,l})$.

Claim 4 For any $s > 1$, there exists $c' \geq 1$, such that $\sum_{l=0}^{\log n} p_{s,l} < c'$.

Proof: For any l , $0 \leq l \leq \log n$ and $s > 1$, $p_{s,l} \leq (1 - \bar{q}(s-1))^{2^l} 2^l q(s)$. Let c_{min} be the minimal $c \geq 1$ satisfying (4), and

$$c' = 2c_{min} , \quad (7)$$

then

$$\begin{aligned} \sum_{l=0}^{\log n} p_{s,l} &< \sum_{l=0}^{\log n} (1 - \bar{q}(s-1))^{2^l} 2^l c_{min} \bar{q}(s-1) && \text{(by Claim 3)} \\ &\leq 2c_{min} \bar{q}(s-1) \sum_{t=0}^n (1 - \bar{q}(s-1))^t \\ &= c' \bar{q}(s-1) \frac{1 - (1 - \bar{q}(s-1))^{n+1}}{\bar{q}(s-1)} < c' . \end{aligned}$$

□

The expected number of phases until S_H sends a NACK for the first time is given by

$$E[T^A] = \sum_{l=0}^{\log n} \frac{1}{\log n + 1} \sum_{s \geq 1} p_{s,l} \cdot s . \quad (8)$$

Let $\omega > 1$ be an integer (to be determined), then

$$E[T^{\mathcal{A}}] \geq \frac{\omega}{\log n + 1} \sum_{l=0}^{\log n} \sum_{s \geq \omega} p_{s,l} . \quad (9)$$

In addition, from the definition of $p_{s,l}$ and from Claim 4

$$\sum_{l=0}^{\log n} \sum_{s=1}^{\omega} p_{s,l} \leq c' \omega , \quad (10)$$

and since

$$\sum_{s \geq r} p_{s,l} = 1 - \sum_{s=1}^{\omega-1} p_{s,l}$$

it follows from (9) that

$$\begin{aligned} E[T^{\mathcal{A}}] &\geq \frac{\omega}{\log n + 1} \sum_{l=0}^{\log n} \left(1 - \sum_{s=1}^{\omega-1} p_{s,l}\right) \\ &\geq \frac{\omega}{\log n + 1} (\log n + 1 - c' \omega) . \end{aligned}$$

Taking $\omega = \frac{\log n + 1}{2c'}$ we have

$$\begin{aligned} E[T^{\mathcal{A}}] &\geq \frac{1}{2c'} (\log n + 1 - c' \left(\frac{\log n + 1}{2c'}\right)) \\ &= \frac{\log n + 1}{4c'} \end{aligned}$$

which completes the proof. □

3.2.2 Non-Uniform Algorithms

We now extend the lower bound of Theorem 2 to non-uniform algorithms, where different receiver nodes may follow different algorithms.

Theorem 5 *For any r , $1 \leq r \leq m$ and a randomized algorithm \mathcal{A} , if \mathcal{A} sends $O(\log r)$ NACK messages, then*

$$E[T^{\mathcal{A}}] = \Omega(\log n) . \quad (11)$$

Proof: Denote by $q_i(s)$ the probability that node i sends NACK in phase s , and by

$$\bar{q}_i(s-1) = 1 - (1 - q_i(1)) \cdot (1 - q_i(2)) \cdots (1 - q_i(s-1))$$

the probability that node i sends at least one NACK in the first $(s-1)$ phases. Since the algorithm is non-uniform, these probabilities may now be different for different nodes. Let

$$q(s) = \frac{1}{|S_H|} \sum_{i=1}^{|S_H|} q_i(s)$$

be the average success probability of a node in S_H in phase s , and

$$\bar{q}(s-1) = \frac{1}{|S_H|} \sum_{i=1}^{|S_H|} \bar{q}_i(s-1)$$

be the average success probability of a node in S_H in the first $(s-1)$ phases.

In the Appendix we prove that Claims 3 and 4 hold also when \mathcal{A} is non-uniform, which suffices to prove the theorem. \square

4 An Algorithm for Reliable Multicast

In this section, we present a randomized distributed algorithm that guarantees the transmission of at least m distinct cells to each receiver. In Section 4.1 we consider a simplified version of the problem, in which the initial transmission of cells is unreliable, but all later retransmissions are. Section 4.2 addresses the more realistic case, where retransmissions are also unreliable.

Our algorithm assumes that each receiver node knows the number of receivers, n , and uses it in its coin tosses. While this assumption may not hold in a common network setting, counting the receivers and multicasting the value of n to the receivers can be done as a preprocessing step of the algorithm. For our purposes, such a preprocessing step needs to be done only periodically, namely, when the number of receivers increases/decreases by at least factor of 2. Such changes do not occur very often; after this preprocessing step, the sender can transmit a large number of packets to the receivers, before further updates are required.

4.1 Reliable Retransmission

Assume that m_i cells are missing in receiver node i after the initial transmission, $0 < m_i \leq m$. Since retransmission is reliable, it suffices to guarantee that the sender knows $\max_i m_i$, so it can

retransmit additional cells. Thus, if a receiver node i sends $\text{NACK}(i, m_i)$ to the sender, then it will get at least m_i cells in the next phase, satisfying its cell deficit.

The algorithm partitions the set of receivers into $\log m$ subsets $S_1, \dots, S_{\log m}$; receiver node i is in S_j if $2^{j-1} < m_i \leq 2^j$; receiver i knows the subset S_j it belongs to.

Let $H = \max_{1 \leq i \leq n} \lceil \log m_i \rceil$. Since nodes in S_H have the highest cell deficit, it suffices that some node in S_H will send a NACK message. Since H is unknown, receivers in every subset S_j must assume that $H = j$ and attempt to have a member of S_j send a NACK message.

Randomization is used by receivers in S_j to minimize the number of NACKs. At time $t \geq 1$, a receiver in S_j tosses a coin and sends a NACK with some probability $\rho(t)$.

If each receiver node i knows the size of S_l , where $l = \lceil \log m_i \rceil$, then we can employ an iterative algorithm. In each round, node i sends a NACK message with fixed probability $\rho = \frac{1}{|S_l|}$. It can be shown that, using this algorithm, each receiver node obtains m cells with probability $1 - 1/m$ within $O(\log m)$ rounds, and that $E[\text{COM}_N] = O(\log m)$.

Unfortunately, the receivers do not know the size of their subsets. Instead, each node modifies its selection probability, and keeps tossing the coin until it gets “Heads”, or it receives the required number of cells (due to NACK messages sent by other nodes).

The algorithm operates in phases; initially, $m_i(1) = m_i$. In each phase $t \geq 1$, if $m_i(t) = 0$ then receiver node i stops; otherwise, it sends NACK with probability $\rho := 2^{t-1}n^{-1}$. It waits until it receives N_t cells, updates the number of missing cells and proceeds to the next phase. Note that since the deficit of node i , $m_i(t)$, may be different for different values of t , so does the index of the set to which i belongs in each iteration. We denote this index below as j_t . The receiver’s pseudocode for algorithm *choose with unknown partition (CUP)* appears in Figure 2.

Given a packet of length L , assume that the amount of data that can be transmitted in a single cell is L/m , for some m . The sender initially generates $R = 3m/2$ pieces from a packet, using the FEC algorithm, with R and m as parameters; this implies that a receiver node can reconstruct the original packet from any subset of m cells. The sender initially assumes that the network is reliable and sends exactly m cells; the remaining cells are kept “on the shelf”, in a cyclic list, for retransmissions (this means a storage overhead of 50%).

The cells on the shelf are transmitted in a round-robin fashion, using a pointer, which circulates on the list. This guarantees that any receiver node, i , whose deficit is at most $m/2$ can obtain all of its missing cells from the shelf. In other words, if the sender transmits from the shelf some cell, b , which helps to cover the deficit of i , then this cell will not be transmitted

```

Receiver ( $i, n$ );
   $t := 1$ ;
   $subset(i) = \lceil \log m_i \rceil$ ;
  while  $m_i > 0$  do
     $\rho := \frac{2^{t-1}}{n}$ ;
    toss a coin with Prob(Heads)= $\rho$ ;
    if (Heads) then send NACK( $subset(i), m_i$ ) to the sender;
     $t := t + 1$ ;
    wait for  $N_t$  cells;
     $m_i := \max(m_i - N_t, 0)$ ;

```

Figure 2: Algorithm *Choose with Unknown Partition (CUP)*: Pseudocode for receiver node i .

again before the $m/2 - 1$ other cells on the shelf were transmitted. Since retransmissions are reliable, by the time b is retransmitted, all the deficit of node i has been compensated. When the deficit of i is larger than $m/2$, the sender does not use the shelf, and the original message is retransmitted.

NACK messages received by the sender in phase t are of the form $\text{NACK}(j_t, m(t))$, where $1 \leq j_t \leq \lceil \log m \rceil$ indicates that the receiver belongs to the set S_{j_t} at time t , and $m(t)$ is the cell deficit of this receiver.

The sender responds to the NACK messages received in phase t as follows.

- First, the sender accumulates all the NACK messages in the set $\text{N_Mes}(t)$. Let $j_{max} = \max_{\text{N_Mes}(t)} j_t$; then, among the NACK messages sent from $S_{j_{max}}$, the sender responds to the message in which the deficit is maximal, $\max(t)$.
- The sender responds to the message $\text{NACK}(j_{max}, \max(t))$ by the following rule.
 - If $2^{j_{max}} > m/2$ then it retransmits the m cells originally computed for the packet;
 - otherwise, it transmits the next $\max(t)$ cells on the shelf.

The extra cells are kept only for the duration of the CUP algorithm, i.e., for $O(\log n)$ phases, after which these cells are discarded. The sender's pseudocode for algorithm CUP appears in Figure 3.

```

Sender ( $m, n$ );
  prepare  $R = m \cdot 3/2$  cells;
   $L := R - m$  ;
  put on the shelf  $cell(0), \dots, cell(L - 1)$  ;
   $t := 1$ ;
Start: send  $m$  cells;
   $pointer := 0$ ;
  while ( $t < \log n$ ) do
    receive the set of NACK messages of phase  $t$ ,  $N\_Mes(t)$ ;
    if  $N\_Mes(t) \neq \emptyset$  then
      Let  $NACK(j_{max}, \max(t))$  be the message with maximal deficit;
      if ( $2^{j_{max}} \leq L$ )
        send  $\max(t)$  cells starting at  $cell(pointer)$ 
        to all receivers;
         $pointer := (pointer + \max(t)) \bmod L$ ;
      else goto Start
     $t := t + 1$ ;

```

Figure 3: Algorithm CUP: Pseudocode for the sender.

From the above description of the sender's protocol, we note that $\mu_{cup} = 2$, since at any phase $t \geq 1$, if $\max(t) \geq m/2 + 1$, the sender retransmits m cells. It follows from Theorem 1 that the expected message complexity of CUP is $\Omega(\log_2 m)$. We show below that the message complexity of CUP is $\Theta(\log m)$. By Theorem 2, this implies a lower bound of $\Omega(\log n)$ on the time complexity of the algorithm. We show that CUP achieves this bound.

To simplify the calculations, we assume below that n is a power of 2; however, our results hold for any $n \geq 1$.³

Theorem 6 *Algorithm CUP terminates within $O(\log n)$ phases, and its message complexity is $O(\log m)$.*

Proof: Clearly, for each receiver node i , after $\log n$ phases either i already obtained its missing cells, or i sends a NACK message with probability 1, and terminates. This proves that the algorithm terminates within $O(\log n)$ phases.

Let $E[COM_N(j)]$ be the expected number of NACK messages sent by nodes in subset S_j , $1 \leq j \leq H$. Since

$$E[COM_N] = \sum_{j=1}^H E[COM_N(j)] ,$$

it suffices to show that $E[COM_N(j)]$ is bounded by a small constant, for any j , $1 \leq j \leq H$, in order to bound the message complexity of algorithm CUP.

For any j , $1 \leq j \leq H$, write $|S_j|$ as $\frac{n}{2^k}$ for some k , $0 \leq k \leq \log n$. Let A_s^j be the event "the first round in which a node in S_j sends a NACK message is s ". We can write:

$$\begin{aligned} E[COM_N(j)] &= \sum_{s=1}^{\log n} E[COM_N(j) | A_s^j] \cdot \text{Prob}(A_s^j) \\ &= [COM_N(j) | A_1^j] \cdot \text{Prob}(A_1^j) + \sum_{s=2}^{\log n} E[COM_N(j) | A_s^j] \cdot \text{Prob}(A_s^j) \\ &\leq \frac{1}{2^k} + \frac{n}{2^k} \sum_{s=1}^{\log n-1} \frac{2^s}{n} \prod_{t=1}^s \left(1 - \frac{2^{t-1}}{n}\right)^{n/2^k} \\ &\leq \frac{1}{2^k} + \frac{n}{2^k} \sum_{s=1}^{\log n-1} \frac{2^s}{n} \cdot \exp\left(\sum_{t=0}^{s-1} -2^{t-k}\right) \quad (\text{by a standard approximation}) \end{aligned}$$

³For the case where $2^j < n < 2^{j+1}$ one can choose $n' = 2^{j+1}$ and assume that $(n' - n)$ receiver nodes have zero initial deficit.

$$\begin{aligned}
&\leq \frac{1}{2^k} + \sum_{s=1}^{\log n-1} \frac{2^{s-k}}{\exp(2^{s-k-1})} \\
&= \frac{1}{2^k} + e \sum_{s=1}^{\log n-1} \frac{2^{s-k}}{\exp(2^{s-k})} \\
&\leq \frac{1}{2^k} e \sum_{s=0}^{\lfloor k \rfloor + 1} 2^s + \sum_{s=0}^{\log n} \frac{e 2^s}{\exp(2^s)} < 7e,
\end{aligned}$$

which implies the theorem. \square

4.2 Loss of NACKs and Retransmitted Cells

In this section we apply algorithm CUP to the case where retransmissions of cells and transmissions of NACK messages are *unreliable*. Specifically, for each receiver node i and any cell transmitted by the sender, the cell arrives to i with probability $1 - q_L^c(i)$. In addition, a NACK message sent from receiver node i , may be lost with probability $q_L^N(i)$. Loss probabilities in modern networks are typically small. We assume here that $q_L^N(i), q_L^c(i) \in (0, 1/4]$, for all $1 \leq i \leq n$.

The receivers use the algorithm in Figure 2. The sender uses the CUP algorithm with the following change. For each subset S_j , $1 \leq j \leq \log m$ the sender keeps a counter. When a NACK $(j_t, m(t))$ is received at time t from a member in S_{j_t} , the sender examines the counter of S_{j_t} . The sender accepts as input parameter the maximal loss probability of a cell in the network, given by q_L^c .⁴ The sender then uses in its protocol the value of $d \geq 1$, derived from the input parameter q_L^c . Specifically, d is the minimal integer satisfying $q_L^c \leq 1/n^{1/d}$, i.e., $d = \lceil \log n / \log(1/q_L^c) \rceil$.

While the counter is lower than $2d$, i.e., less than $2d$ NACKs were received from S_j , the sender retransmits 2^{j_t+1} cells; otherwise, it retransmits to the receivers $2m(t)$ cells. The modified protocol for the sender is given in Figure 4.

Theorem 7 *For a network with reliability parameters q_L^c and q_L^N , algorithm CUP terminates within $O(\log n)$ phases. The expected message complexity of the algorithm is $O(\log m)$.*

Proof: For deriving the bounds we may assume equal loss probabilities for all the receivers (The set of loss probabilities $q_L^c(1), \dots, q_L^c(n)$, and $q_L^N(1), \dots, q_L^N(n)$, can be replaced by $\hat{q}_L^c(i) = \max_{1 \leq i \leq n} q_L^c(i)$ and $\hat{q}_L^N(i) = \max_{1 \leq i \leq n} q_L^N(i) \forall 1 \leq i \leq n$).

⁴This parameter can be deduced from monitoring the statistical behavior of the network.

```

Sender( $m, n, q_L^c$ );
  prepare  $R = m \cdot 3/2$  cells;
   $L := R - m$  ;
  put on the shelf  $cell(0), \dots, cell(L - 1)$  ;
   $d = \lceil \log n / \log(1/q_L^c) \rceil$ ;
   $t := 1$ ;
Start: send  $m$  cells;
   $pointer := 0$ ;
  for  $j := 1$  to  $\log m$   $counter(j) := 0$ ;
  repeat
    receive the set of NACK messages of phase  $t$ , N_Mes( $t$ );
    Let NACK( $j_{max}, \max(t)$ ) be the message with maximal deficit;
    if ( $counter(j_{max}) < 2d$ ) then  $m_s := 2^{j_{max}+1}$ ;
      else  $m_s := 2 \max(t)$ ;
     $counter(j_{max}) := counter(j_{max}) + 1$ ;
    if ( $m_s \leq L$ )
      send  $m_s$  cells starting at  $cell(pointer)$ 
      to all receivers;
       $pointer := (pointer + m_s) \bmod L$ ;
    else goto Start
   $t := t + 1$ ;
until false; (loop forever)

```

Figure 4: Algorithm CUP: Pseudocode for the sender in the unreliable case

Note that in our computations we allow arbitrary dependencies between the losses of cells (or NACK messages) sent to/from different receivers in a given round. However, we assume that for specific receiver node, i , losses occurring in *different rounds* are independent.

For time complexity, recall that after $\log n$ phases, each receiver node which has not obtained its missing cells, sends a NACK with probability 1. For computing the bound we may assume that i sends a NACK after $\log n$ phases, and needs to wait until it receives m_i cells from the sender. Note that for each NACK message sent by i , the expected number of phases until a NACK is received by the sender is $\frac{1}{1-q_L^N}$. In response to $\text{NACK}(\text{subset}(i), m_i)$ the sender retransmits at least $2m_i$ cells in the next phase. Denote by Y the random variable which counts the number of cells lost in a retransmission of $2m_i$ cells to receiver i , then from Markov's inequality:

$$\text{Prob}(Y > m_i) < 2q_L^c . \quad (12)$$

We note that since (12) holds for any $1 \leq m_i \leq m$, if exactly l NACK messages sent by i were received by the sender, the probability that i needs to send another NACK is bounded by $(2q_L^c)^l$. Therefore the expected number of phases until i receives m_i cells, after it sends the first NACK message, is bounded by

$$\frac{1}{1-q_L^N} \sum_{l \geq 1} (2q_L^c)^{l-1} = \frac{1}{(1-q_L^N)(1-2q_L^c)} = O(1)$$

and the overall time complexity is $O(\log n)$.

For message complexity, it suffices to show that for any $1 \leq j \leq H$, the expected number of NACK messages sent by S_j is bounded by a constant. We write below the total number of messages sent from S_j as the sum of:

- Messages sent from S_j until the d -th phase in which a NACK of S_j was received by the sender. We call each of these phases a *successful phase*.
- NACKs that were sent from S_j after its d -th successful phase.

Let $|S_j| = \frac{n}{2^k}$, for some $0 \leq k \leq \log n$. Recall that when receiving the first $2d$ NACKs from S_j , the sender retransmits 2^{j+1} cells. Therefore, given that the first NACK was sent from S_j in phase s , by the linearity of expectation, we get that the expected number of NACKs sent from S_j till the d -th successful phase is bounded by

$$\frac{1}{1-q_L^N} \frac{n}{2^k} \cdot \sum_{l=1}^d (2q_L^c)^{l-1} \cdot \frac{2^{s+2d}}{n} \leq \frac{1}{1-q_L^N} \frac{n}{2^k} \cdot \frac{2q_L^c}{1-2q_L^c} \cdot \frac{2^{s+2d}}{n} .$$

Let $f(q_L^c) = 2q_L^c/(1 - 2q_L^c)$; then $f'(q_L^c)$ is the first derivative of the function f w.r.t q_L^c .

The expected number of NACKs sent from S_j after the d th successful phase is clearly bounded by

$$\begin{aligned} \frac{n}{2^k} (2q_L^c)^d \frac{1}{1 - q_L^N} \sum_{l \geq 1} l (2q_L^c)^{l-1} &= \frac{n}{2^k} (2q_L^c)^d \frac{1}{1 - q_L^N} f'(q_L^c) \\ &= \frac{n}{2^k} (2q_L^c)^d \frac{1}{1 - q_L^N} \cdot \frac{1}{(1 - 2q_L^c)^2} \end{aligned}$$

Hence, we can summarize writing

$$\begin{aligned} E[COM_N(j)] &\leq \frac{n}{2^k} \sum_{s=1}^{\log n} \prod_{t=0}^{s-1} \left(1 - \frac{2^t}{n}\right)^{n/2^k} \\ &\quad \cdot \left(\frac{2q_L^c}{(1 - q_L^N)(1 - 2q_L^c)} \cdot \frac{2^{s+2d}}{n} + (2q_L^c)^d \frac{2}{(1 - q_L^N)(1 - 2q_L^c)^2} \right) \\ &\leq \frac{n}{2^k} \sum_{s=1}^{\log n} \prod_{t=0}^{s-1} \left(1 - \frac{2^t}{n}\right)^{n/2^k} \cdot \frac{2^s}{n} \cdot O(1) . \end{aligned}$$

Following the steps of the proof of the message complexity of the CUP for the reliable case (Theorem 6) we get the statement of the theorem. \square

5 Discussion

A scheme for reliable multicast using FEC was considered, together with a randomized algorithm which minimizes the number of rounds and the amount of receiver-to-sender traffic. Lower bounds on the time and the number of negative acknowledgments show that this algorithm is optimal.

Our paper leaves open several interesting avenues for further research, e.g., How do we take into account the fact that cells are typically lost in batches? Can information about transmission of previous packets be used to tune the redundancy factor? Our scheme can be applied also for the transmission of audio/video data, where receivers need to obtain only a certain fraction of the cells, to guarantee a desired quality of service. We expect that allowing different levels of service would enable to derive better performance bounds.

Finally, it would be interesting to see if our algorithm can be applied without knowledge of n , the number of receivers, or whether stronger lower bounds can be proved for this case.

References

- [1] N. Alon, J.H. Spencer, *The Probabilistic Method*, Wiley-Interscience, 1992.
- [2] A. Bestavros and G. Kim, "TCP Boston: A Fragmentation-tolerant TCP Protocol for ATM Networks," in Proceedings of *Infocom'97: The IEEE International Conference on Computer Communication*, (Kobe, Japan), April 1997.
- [3] A. Bestavros, "AIDA-based Real-Time Fault-Tolerant Broadcast Disks," in Proceedings of *Second IEEE Real-Time Technology and Applications Symposium* (Boston, MA), 1996.
- [4] K. Birman and R. van Renesse (eds.), *Reliable distributed programming with the Isis Toolkit*, IEEE Computer Society Press, 1993.
- [5] J. Byers, M. Luby, M. Mitzenmacher and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," In Proceedings of ACM SIGCOMM, September 1998, Vancouver.
- [6] G. Carle, "Towards scalable error control for reliable multipoint services in ATM networks," In Proceedings of 12th International Conference on Computer Communication (ICCC'95), August 1995.
- [7] D. Dolev and D. Malki, "The Transis Approach to High Availability Cluster Communication," *Communications of the ACM*, Vol. 39, No. 4 (April 1996).
- [8] J. Gemmell, E. Schooler, and J. Gray, "Fcast multicast file distribution," *IEEE Networks*, Vol.14, No.1 (Jan/Feb 2000), pp.58–68.
- [9] M. Hofri, *Analysis of Algorithms: Computational Methods and Mathematical Tools*. Oxford University Press, 1995.
- [10] C. Huitema, "The Case for Packet Level FEC," In Proceedings IFIP 5th Int'l Workshop on Protocols for High Speed Networks, October 1996.
- [11] E. Kushilevitz and Y. Mansour, "An $\Omega(D \log(N/D))$ Lower Bound for Broadcast in Radio Networks," *SIAM Journal on Computing*, Vol. 27, No 3 (1998), pp. 702–712.
- [12] M. Lacher, J. Nonnenmacher and E. Biersack, Performance Comparison of Centralized versus Distributed Error Recovery for Reliable Multicast," *IEEE/ACM Transactions on Networking*, Vol. 8, No. 2 (2000), pp. 224–238.

- [13] F.J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, 1977.
- [14] L. E. Moser, P. M. Melliar-Smith, D. A. Agarwal, R. K. Budhia, and C. A. Lingley-Papadopoulos, “Totem: A Fault-Tolerant Multicast Group Communication System,” *Communications of the ACM*, April 1996.
- [15] J. Nonnenmacher and E. Biersack, “Scalable Feedbacks for Large Groups,” *IEEE/ACM Transactions on Networking*, Vol. 7, No. 3 (1999), pp. 375–386.
- [16] J. Nonnenmacher, E. Biersack and D. Towsley, “Parity-Based Loss Recovery for Reliable Multicast Transmission,” *IEEE/ACM Transactions on Networking*, Vol. 6, No. 4 (1998), pp. 349–361.
- [17] M. O. Rabin, “Efficient Dispersal of Information for Security, Load Balancing and Fault Tolerance,” *Journal of the ACM*, Vol. 36, No. 2 (1989), pp. 335–348.
- [18] Robbert van Renesse, Kenneth P. Birman and Silvano Maffei, “Horus, a flexible Group Communication System,” *Communications of the ACM*, April 1996.
- [19] L. Rizzo, “Effective Erasure Codes for Reliable Computer Communication Protocols”, *Computer Communication Review*, Vol. 27, No. 2 (April 1997), pp. 24–36.
- [20] L. Rizzo and L. Vicisano, “RMDP: An FEC-based Reliable Multicast Protocol for Wireless Environments,” *Mobile Computing and Communications Review*, Vol. 2, No. 2, April 1998.
- [21] D. Rubenstein, J. Kurose, D. Towsley, “Real-Time Reliable Multicast Using Proactive Forward Error Correction,” Proceedings of IEEE NOSSDAV '98.
- [22] W.T. Strayer, B.J. Dempsey, A.C. Weaver, *XTP: The Xpress Transfer Protocol*, Addison-Wesley, Reading, Mass 1992.
- [23] B. Whetten, T. Montgomery, and S. Kaplan, “A High Performance Totally Ordered Multicast Protocol,” *Theory and Practice in Distributed Systems*, Lecture Notes in Computer Science #938, Springer Verlag.

A Background – Multicast in ATM Networks

The assumptions used in our multicast model are satisfied, for example, by *point-to-multipoint connection* connections in an *asynchronous Transfer Mode (ATM)* network, which have the

following properties.

1. One ATM link, called the *root link*, serves as the root in a simple tree topology. When the *root node*, adjacent to the root link, sends information, all of the remaining nodes on the connection, called *leaf nodes*, receive copies of the information.
2. Each of the leaf nodes on the connection can send information directly to the root node. The root node cannot distinguish which leaf is sending information without additional (higher layer) information.
3. The leaf nodes cannot communicate directly to each other with this connection type.

ATM networks transmit 53-byte long *cells* from one node to another; a packet longer than 53 bytes should be fragmented into cells, which are sent in sequence over the point-to-multipoint connection. A simple approach to fragmentation is to “chop” the packet into pieces, and send them to the receivers. Delivery of cells in ATM networks is *best-effort*, so cells can be lost, typically due to congestion at intermediate switches along the route from the sender to the receivers. When cells comprising a fragmented packet are lost, the packet cannot be reassembled. Thus, native packet delivery on ATM point-to-multipoint connections is not reliable.

B Some proofs

Proof of Claim 3 for non-uniform algorithms: Take $c = 4\delta$, where δ is defined in (5), and assume that there exists $s > 1$, such that $q(s) > c\bar{q}(s - 1)$. Note, that for any $N > 1$ and p_1, \dots, p_N , $0 \leq p_i \leq 1$,

$$\prod_{i=1}^N (1 - p_i) \geq 1 - \sum_{i=1}^N p_i . \quad (13)$$

This can be shown by induction on N . Hence,

$$\begin{aligned} E[COM_N(H, s)] &\geq \prod_{i=1}^{|S_H|} (1 - \bar{q}_i(s - 1)) \sum_{i=1}^{|S_H|} q_i(s) \\ &> \prod_{i=1}^{|S_H|} (1 - \bar{q}_i(s - 1)) c \sum_{i=1}^{|S_H|} \bar{q}_i(s - 1) \\ &\geq (1 - \sum_{i=1}^{|S_H|} \bar{q}_i(s - 1)) \cdot c\bar{q}(s - 1) \cdot |S_H| \\ &= (1 - \bar{q}(s - 1)|S_H|) \cdot c\bar{q}(s - 1) \cdot |S_H| . \end{aligned}$$

Assume that $\bar{q}(s-1) = \frac{1}{2^k}$ for $k > 1$ (if $k \leq 1$ then $q(s) \leq 2\bar{q}(s-1)$, therefore Claim 3 holds), and let $|S_H| = 2^{k-1}$, then

$$E[COM_N(H, s)] > (1 - \frac{1}{2^k} \cdot 2^{k-1}) \cdot c \cdot \frac{1}{2^k} \cdot 2^{k-1} = \frac{c}{4}$$

which contradicts inequality (5). □

Proof of Claim 4 for non-uniform algorithms: We note, that for any $0 \leq l \leq \log n$ and $s > 1$

$$p_{s,l} \leq \prod_{i=1}^{2^l} (1 - \bar{q}_i(s-1)) \sum_{i=1}^{2^l} q_i(s) ,$$

then

$$\sum_{l=0}^{\log n} p_{s,l} \leq \sum_{l=0}^{\log n} \prod_{i=1}^{2^l} (1 - \bar{q}_i(s-1)) \sum_{i=1}^{2^l} q_i(s) .$$

Define the function

$$f(x) = \ln(1 - x) .$$

Since f is concave, Jansen's inequality [1] implies

$$\sum_{i=1}^{2^l} f(\bar{q}_i(s-1)) \leq 2^l f(\bar{q}(s-1)) .$$

Exponentiating in both sides we have

$$\prod_{i=1}^{2^l} (1 - \bar{q}_i(s-1)) \leq ((1 - \bar{q}(s-1))^{2^l}) .$$

Choose c' as in (7), then

$$\begin{aligned} \sum_{l=0}^{\log n} p_{s,l} &\leq \sum_{l=0}^{\log n} ((1 - \bar{q}(s-1))^{2^l} \cdot 2^l q(s)) \\ &\leq \sum_{l=0}^{\log n} ((1 - \bar{q}(s-1))^{2^l} \cdot 2^l c_{\min} \bar{q}(s-1)) \end{aligned}$$

which implies Claim 4. □