A Linear Time Erasure-Resilient Code With Nearly Optimal Recovery

Noga Alon * Michael Luby †

Abstract

We develop an efficient scheme that produces an encoding of a given message such that the message can be decoded from any portion of the encoding that is approximately equal to the length of the message. More precisely, an (n, c, ℓ, r)-erasure-resilient code consists of an encoding algorithm and a decoding algorithm with the following properties. The encoding algorithm produces a set of ℓ-bit packets of total length cn from an n-bit message. The decoding algorithm is able to recover the message from any set of packets whose total length is r, i.e., from any set of r/ℓ packets. We describe erasure-resilient codes where both the encoding and decoding algorithms run in linear time and where r is only slightly larger than n.

1 Introduction

Most existing and proposed networks are packet based, where a packet is fixed length indivisible unit of information that either arrives intact upon transmission or is completely lost. This model accurately reflects properties of Internet and ATM-based networks, where local error correcting codes can be used (and often are used) on individual packets to protect against possible

*Department of Mathematics, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv, Israel and Institute for Advanced Study, Princeton, NJ 08540, USA. Email: noga@math.tau.ac.il. Research supported in part by the Fund for Basic Research administered by the Israel Academy of Sciences and by a USA-Israeli BSF grant.

†International Computer Science Institute, Berkeley, California, 94704, and Computer Science Division, UC Berkeley. Berkeley, California, 94720. Email: luby@icsi.berkeley.edu. Research supported in part by National Science Foundation operating grant CCR-9304722 and NCR-9416101, United States-Israel Binational Science Foundation grant No. 92-00226, and ESPRIT BR Grant EC-US 030.

errors as the packet traverses the network. However, the timely arrival of individual packets sent long distances over a variety of heterogeneous networks is a global property that seems to be harder to control on a local basis. Thus, it makes sense to protect real-time traffic sent through such networks against losses by adding a moderate level of redundancy using erasure-resilient codes.

Algorithms based on this approach have been developed for applications such as multicasting real-time high-volume video information over lossy packet based networks [3, 2, 9] and other high volume real-time applications [14]. The two most important properties of erasure-resilient codes in these applications are the running times of the encoding and decoding algorithms and the amount of encoding sufficient to recover the message. An erasure-resilient code where any portion of the encoding equal to the length of the message is sufficient to recover the message is called a *maximal distance separable* (MDS) code in the literature. An ideal erasure-resilient code would be an MDS code equipped with linear time encoding and decoding algorithms, but so far no such code is known.

Standard Reed-Solomon codes can be used to implement quadratic time MDS codes. These methods have been customized to run in real-time for medium quality video transmission on existing workstations [3, 2], i.e., at the rate of a few megabits per second, but high quality video sent at the rate of hundreds of megabits per second will require either better algorithms or custom designed hardware. Theoretically more efficient (but not linear time) MDS codes can be constructed based on evaluating and interpolating polynomials over specially chosen finite fields using Discrete Fourier Transform, but these methods are not competitive in practice with the simpler quadratic methods except for extremely large messages, and even then, their time complexity is at least $\Omega(n \log n)$, see, e.g., [10], Chapter 11.7 and [13], p. 369. Thus, the design of highly efficient algorithms for implementing erasure-resilient codes is interesting theoretically and important for practical applications.

Our scheme has the property that the code can be constructed for any message length n and encoded message length cn, where $c > 1$. We call such a code an (n, c)-code. In applications, c is relatively small (it typically varies between 1 and 2, but can be as large as 5). Thus, when stating running times, we ignore the dependence on c.

A natural relaxation of an MDS code is to allow slightly more of the encoding than the optimal amount in order to recover the message: We say an (n, c)-code is $(1+\epsilon)$-MDS if the message can be recovered from any $(1+\epsilon)n$

of the encoding. For example, if $\epsilon = .05$ then only 5% more of the encoding than the optimal amount is sufficient to recover the message. A further relaxation of an MDS code is to allow both the encoding and decoding algorithms to be probabilistic (using the same set of random bits for both encoding and decoding): We say an (n, c)-code is probabilistic $(1+\epsilon)$-MDS if the message can be recovered from any $(1+\epsilon)n$ of the encoding with high probability. For probabilistic codes, the network is assumed to drop packets independent of their contents.

Of secondary importance, but still important, is the length ℓ of the packets. Ideally, the length of the packets should be as short as possible, e.g., $\ell = 1$, because an erasure-resilient code using longer packets can always be constructed by concatenating several packets from an erasure-resilient code using shorter packets, but the reverse is not necessarily true. Internet packets for sending video are typically moderate sized, e.g., 1000 bytes, but ATM cells are rather short, i.e., 48 bytes, and here the payload size is more of a constraint. We try to minimize the value of ℓ as much as possible, but in general ignore this parameter when stating results.

We assume that each packet contains a unique index. From this index, the portion of the encoding carried by each received packet can be determined. We do not count the space for the index as part of the packet size. In practice, the space for this index is small compared to the size of the payload of the packet, and packet based protocols typically include a unique index for each packet within the packet in any case.

This paper describes a new erasure-resilient code obtained by a deterministic scheme. A preliminary description of this scheme, together with descriptions of probabilistic schemes that are more efficient based on this scheme can be found in [4]. The scheme has the property that, on inputs n, c, and ϵ, the run time of the (n, c)-code is $\mathcal{O}(n/\epsilon^4)$, it is $(1+\epsilon)$-MDS, and the packet size is $\mathcal{O}((1/\epsilon^4)\log(1/\epsilon))$. Note that for constant ϵ this scheme runs in linear time. Although this is an interesting theoretical result, it is not clear if it can be made practical for values of ϵ that are reasonable, e.g., $\epsilon = .10$, because the $(1/\epsilon^4)$ factor is rather large both in terms of the running time and the packet size.

The scheme we describe produces a *systematic* code, which means that the message itself is part of the encoding. This property is good especially in the case when only a small number of the packets are lost, because the time to decode the message from the encoding is proportional only to the amount of the message that is missing.

Our scheme is based on the properties of *expanders* which are explicit graphs with pseudo-random properties. The relevance of these graphs to error correcting codes has been observed in [5], and indeed we apply some of the ideas of that paper.

Erasure-resilient codes are related to error correcting codes, and are typically easier to design. For example, an error correcting code with encoding length cn that can correct up to bn bit flips can be used as an erasure-resilient code that can recover the message from any $(c-b)n$ of the encoding: For packets not received, set the missing bits to zero and then use the error correcting code to recover the message. This can be improved to nearly $(c-2b)n$ by setting the missing bits randomly, noting that on average half of them will be correctly set. Moreover, if the corresponding error correcting code has efficient encoding and decoding algorithms, then so does the resulting erasure-resilient code.

The recent breakthrough result of Spielman [16] on error correcting codes is directly relevant to our scheme. Spielman applies the techniques in [15] and [8], and constructs linear time error correcting codes with linear rate and linear minimum distance. This error correcting code stretches an n-bit message to a cn-bit message and can recover the message when up to bn of the encoding bits are flipped. Here, $b << 1$ and $c \approx 4$ are absolute constants. A direct application of [16] to the design of an erasure-resilient code yields a linear time code that at best is $(4-2b)$-MDS. Thus, [16] cannot be used directly to yield an $(1+\epsilon)$-MDS code for an arbitrary value of ϵ. Nevertheless, [16] is a crucial ingredient in our construction.

2 Overview

The input parameters for the encoding and decoding algorithms are (n, c, ϵ), where n is the number of letters in the message to be encoded ($q = \mathcal{O}(\log(1/\epsilon))$ is sufficient for the length of a letter). The encoding algorithm accepts as input an n letter message and produces a cn letter encoding partitioned into packets containing $c'\ell$ letters each, where c' and ℓ are described below. The decoding algorithm is able to reconstruct the entire message from any set of the packets that in total contain $(1+\epsilon)n$ letters, i.e., from any set of $(1+\epsilon)n/c'\ell$ of the packets. Both the encoding and decoding algorithms run in time $\mathcal{O}(n/\epsilon^4)$, where this time is measured in terms of basic operations on letters of length q.

Here is a general overview of the steps of our construction. We first define appropriate parameters γ, γ' which depend linearly on ϵ, and β which depends quadratically on ϵ. Given a message of n letters $M_1, \ldots, M_n$, we first show how to construct additional letters $S_1, \ldots, S_{\gamma' n}$ so that the entire message can be recovered from any $n - \beta n$ portion of $M_1, \ldots, M_n$, given all of the letters $S_1, \ldots, S_{\gamma' n}$. This is done using expander graphs, as described in the next section. Next we apply the construction of [16] to stretch $S_1, \ldots, S_{\gamma' n}$ to $S_1, \ldots, S_{\gamma n}$, in a way that enables one to recover all of $S_1, \ldots, S_{\gamma' n}$ from any $\gamma n - \beta n$ portion of $S_1, \ldots, S_{\gamma n}$. Thus, the overall property of the first two stages is that the message can be recovered when up to a βn portion of $M_1, \ldots, M_n, S_1, \ldots, S_{\gamma n}$ is missing.

In the final step we stretch the encoding produced as described above by a factor of $c' = c/(1+\gamma)$. Let $\ell = \mathcal{O}(1/\epsilon^4)$. This is done by partitioning $M_1, \ldots, M_n, S_1, \ldots, S_{\gamma n}$ into $n(1+\gamma)/\ell$ *blocks* of length ℓ each, and then using a standard MDS code to stretch each block separately into an encoded block containing $c'\ell$ letters each. Then, all of the letters of the encoded blocks are mapped using the edges of an appropriate expander into a total of $n(1+\gamma)/\ell$ packets containing $c'\ell$ letters each. These packets have the property that one is able to decode a $1-\beta$ fraction of the original blocks from any portion of the packets which contain in total at least $(1+\epsilon)n$ letters.

The details require some careful analysis, and are described in the next sections.

3 Recovering All from Almost All

We assume that $\epsilon < 1$. Let $\gamma = \epsilon/3$, $\gamma' = \gamma/4$, $\beta = \gamma'^2/8$, and let $q = 4\log(1/\epsilon) + \log(c) + 16$. Throughout, our basic unit of length is a *letter*, which is a bit string of length q. We assume operations on letters, such as the XOR or AND of two letters, can be performed in constant time. The value of q has been chosen large enough so that all of the MDS codes we use in our constructions can be implemented over the finite field GF$[2^q]$.

Let $M_1, \ldots, M_n$ be the message consisting of n letters. The first step of our scheme constructs an encoding $M_1, \ldots, M_n, S_1, \ldots, S_{\gamma n}$ with the property that the message can be recovered from any fraction $1-\beta$ of this encoding.

The first step proceeds in two stages. The first stage uses expander graphs to construct $S_1, \ldots, S_{\gamma' n}$ from the message. The property of the first stage is that the entire message can be recovered from any $n - \beta n$ portion of the message given all of $S_1, \ldots, S_{\gamma' n}$. The second stage directly uses the constructions of Spielman [16] to stretch $S_1, \ldots, S_{\gamma' n}$ to $S_1, \ldots, S_{\gamma n}$. This stage has the property that all of $S_1, \ldots, S_{\gamma' n}$ can be recovered from any $\gamma n - \beta n$ portion of $S_1, \ldots, S_{\gamma n}$. Thus, the overall property of the first two stages is that the message can be recovered when up to a βn portion of $M_1, \ldots, M_n, S_1, \ldots, S_{\gamma n}$ is missing.

3.1 Stage 1: Restricted erasures

The main result of this subsection is the following.

Lemma 1 *There is a scheme for generating, for any given message of n letters $M_1, \ldots, M_n$, a sequence of $\gamma' n$ additional letters $S_1, \ldots, S_{\gamma' n}$ with the following properties.*

(i) *The encoding time is $\mathcal{O}(n/\epsilon)$.*

(ii) *If $S_1, \ldots, S_{\gamma' n}$ are known, and at most a fraction β of $M_1, \ldots, M_n$ are missing, then all of $M_1, \ldots, M_n$ can be recovered in time $\mathcal{O}(n/\epsilon^2)$.*

The construction used in the proof of the above is similar to the one in [15], and is based on properties of expanders.

Definition (Expanders): A graph is called a (d, λ)-*expander* if it is d-regular and the absolute value of each of its nontrivial eigenvalues is at most λ.

Let us call an infinite increasing sequence of integers *dense* if the ratio between consecutive elements of the sequence tends to 1. The known constructions of expanders supply, for every admissible degree of regularity, infinite families of graphs on sets of nodes whose cardinalities form a dense sequence. To simplify the presentation we assume here that there are sufficiently many expanders in these families whose number of nodes is divisible by any desired constant. It is not difficult to show that this assumption can be omitted.

By [11], [12] the sequence of integers m for which there is a $(d, 2\sqrt{d-1})$-expander on m nodes is a dense sequence. We need the following from [6].

Proposition 1 *[6] The number of edges induced by any set of x nodes in a (d, λ)-graph on m nodes does not exceed*

$$\frac{1}{2}x(d\frac{x}{m} + \lambda(1 - \frac{x}{m})).$$

Proof of Lemma 1: Fix an integer d, where $\frac{64}{\gamma'^2} < d \leq \frac{128}{\gamma'^2}$, and let $\lambda = 2\sqrt{d-1}$. Let $G = (V, E)$ be a (d, λ)-expander on $m = 2n/d$ nodes. Given a sequence $\{M_e : e \in E\}$ of n letters we define the corresponding sequence $S_1, \ldots, S_{\gamma' n}$ by assigning each node v of G a set of $\gamma' d/2$ letters $S_{v,1}, \ldots, S_{v,\gamma' d/2}$ as described below. Note that the total number of letters $S_{v,j}$ is $m\gamma' d/2 = \gamma' n$, as needed. Let v be a node of G and let $e_1, \ldots, e_d$ be the edges incident with it. Use a quadratic time MDS code to map the message $M_{e_1}, \ldots, M_{e_d}$ to an encoding $M_{e_1}, \ldots, M_{e_d}, S_{v,1}, \ldots, S_{v,\gamma' d/2}$. Such a code can be implemented so that the encoding time is proportional to $d \cdot \gamma' d/2 = \mathcal{O}(d/\epsilon)$ and the decoding time is proportional to d^2, plus an additive $\mathcal{O}(d)$ for each missing message letter. Note that the letter length q is sufficient to implement such a code.

We claim that this scheme satisfies the two properties required in the proposition. The validity of (i) is clear, as the total encoding time is $\mathcal{O}(md/\epsilon) = \mathcal{O}(n/\epsilon)$.

Since we assume we are missing at most βn message letters, and since the time for recover of each missing message letter is $\mathcal{O}(d^2)$, and since $\beta d^2 = \mathcal{O}(1/\epsilon^2)$, it follows that the decoding time is at most $\mathcal{O}(n/\epsilon^2)$.

We now prove that the entire message can be recovered if we are given all the letters associated with the nodes and at most $\beta n = \gamma'^2 n/8$ of the original message letters are missing. The decoding algorithm works as follows. Suppose there is some node v in the graph where at most $\gamma' d/2$ of the message letters associated with the edges incident with v are missing. Then, because $S_{v,1}, \ldots, S_{v,\gamma' d/2}$ are also known, we know a total of at least d letters of the MDS code associated with v. Thus, by the properties of the MDS code, we can recover all the missing message letters associated with edges incident to v. Repeating this process as long as there are such nodes v, we either recover the entire message, or we are left with a nonempty set of edges corresponding to the missing message letters that form a subgraph of minimum degree greater than $\gamma' d/2$ in G.

We now show that Proposition 1 implies that if the subgraph is nonempty then it must contain more than βn edges, and from the assumption that we started with at most βn such edges it will follow that the subgraph must be empty, i.e., the entire message is recovered. Let x denote the number of nodes incident with edges of this subgraph. Then, since each such node has degree at least $\gamma' d/2$ in the subgraph, the total number of edges of the subgraph exceeds $x\gamma' d/4$. Thus, by Proposition 1,

$$x\gamma' d/4 \leq \frac{1}{2} x(dx/m + \lambda(1 - x/m)).$$

Therefore,
$$\frac{\gamma' d}{2} \leq \frac{dx}{m} + \lambda(1 - \frac{x}{m}) \leq \frac{dx}{m} + \lambda.$$
Since $\lambda = 2\sqrt{d-1} < 2\sqrt{d}$, and $d > 64/\gamma'^2$, the inequality $\gamma' d/2 - \lambda \geq \gamma' d/4$ holds and hence
$$dx/m \geq \gamma' d/2 - \lambda \geq \gamma' d/4,$$
implying that the number of edges corresponding to missing letters exceeds
$$x\gamma' d/4 \geq \frac{\gamma'^2 md}{16} = \frac{\gamma'^2}{8} n = \beta n,$$
contradicting the assumption. This completes the proof. ■

3.2 Stage 2: A Spielman-like Construction

We need the following result, which is an easy consequence of the main result of Spielman described in Theorem 12 in [16].

Proposition 2 *[16] There is an absolute positive constant b so that for all m there is an explicit construction that maps messages of m letters into $4m$ letters so that:*

(i) *The encoding time is $\mathcal{O}(m)$.*

(ii) *If at most bm letters are missing then the original m letters can be recovered in time $\mathcal{O}(m)$.*

We note that since we are interested here only in erasure-resilient codes, whereas the construction of Spielman supplies error correcting ones, it is possible to improve the constant b that follows from his construction considerably, but since we are not optimizing the constants here we do not include the details. For simplicity hereafter, we assume $b \geq \gamma'/8$, which implies that $b\gamma' \geq \beta$.

The second stage of the construction uses the construction of Proposition 2 to stretch $S_1, \ldots, S_{\gamma' n}$ to $S_1, \ldots, S_{\gamma n}$. The next lemma follows directly from Lemma 1 and Proposition 2.

Lemma 2

(i) *The encoding $M_1, \ldots, M_n, S_1, \ldots, S_{\gamma n}$ can be computed in $\mathcal{O}(n/\epsilon)$ time from the message $M_1, \ldots, M_n$.*

(ii) *The message can be decoded in time $\mathcal{O}(n/\epsilon^2)$ when at most βn letters of the encoding are missing.*

4 Recovering Almost All

Let $c' = c/(1+\gamma)$, $N = (1+\gamma)n$, and $\ell = 8/(\gamma^2\beta)$. The final goal is to stretch the encoding produced by the first step by a factor of c'. The second step partitions $M_1, \ldots, M_n, S_1, \ldots, S_{\gamma n}$ produced from the first step into *blocks* $B_1, \ldots, B_{N/\ell}$ of ℓ letters each and then uses a standard MDS erasure-resilient codes to produce, for each $i \in \{1, \ldots, N/\ell\}$, an encoding E_i based on B_i consisting of $c'\ell$ letters. Note that the letter length q is sufficient to implement such a code. The properties of the second step are the following.

Lemma 3

(i) *If, for at least a fraction $1-\beta$ of the N/ℓ encodings $E_1, \ldots, E_{N/\ell}$, at least ℓ letters of the encoding are recovered, then the entire message $M_1, \ldots, M_n$ can be recovered.*

(ii) *Both the encoding and decoding times are $\mathcal{O}(n\ell)$.*

Proof of Lemma 3: By properties of MDS codes, for each i where at least ℓ letters of E_i are recovered, the corresponding block of B_i can be completely recovered. From Lemma 2 and the conditions of the lemma, it follows that the message can be completely recovered. The time for both encoding and decoding using a quadratic time MDS code is $(N/\ell) \cdot \ell^2 = N\ell$. ■

The third step of the scheme is to use a $c'\ell$-regular expander graph with N/ℓ nodes to deterministically simulate a "random mapping" of the letters of the encodings $E_1, \ldots, E_{N/\ell}$ into N/ℓ packets $P_1, \ldots, P_{N/\ell}$ containing $c'\ell$ letters each.

Lemma 4 *There is a scheme for mapping the letters of $E_1, \ldots, E_{N/\ell}$ into packets $P_1, \ldots, P_{N/\ell}$ containing $c'\ell$ letters each such that:*

(i) *The time for both the mapping and the inverse mapping is $\mathcal{O}(n)$.*

(ii) *Every set $\mathcal{I} \subseteq \{1, \ldots, N/\ell\}$ with $|\mathcal{I}| \geq \frac{(1+\epsilon)n}{c'\ell}$ has the following property: For at least a fraction $1-\beta$ of $i \in \{1, \ldots, N/\ell\}$, at least ℓ letters of E_i are contained in the set of packets indexed by $\mathcal{I}$.*

Proof of Lemma 4: Let $\lambda = 2\sqrt{c'\ell - 1}$. Let $G = (V, A)$ be a $(c'\ell, \lambda)$-expander with $V = \{1, \ldots, N/\ell\}$. The mapping of the letters of $E_1, \ldots, E_{N/\ell}$ into packets $P_1, \ldots, P_{N/\ell}$ is defined as follows: For each $i \in V$, let

$$(i, w_1), \ldots, (i, w_{c'\ell})$$

be the edges incident to i in G. Then, the j^{th} letter of the encoding E_i is placed into packet P_{w_j}.

Let $\mathcal{I}$ be any subset of V with $|\mathcal{I}| = \frac{(1+\epsilon)n}{c'\ell}$. For each $i \in V$, let d_i denote the number of letters of E_i that are in the packets indexed by $\mathcal{I}$. By a lemma in [6] (see also [7], page 122),

$$\sum_{i \in V} (d_i - |\mathcal{I}|c'\ell^2/N)^2 \leq \lambda^2 |\mathcal{I}|(1 - |\mathcal{I}|\ell/N) \leq 4c'\ell|\mathcal{I}| \leq 8n. \tag{1}$$

Note that

$$\frac{|\mathcal{I}|c'\ell^2}{N} = \frac{(1+\epsilon)n\ell}{N} = \frac{(1+\epsilon)\ell}{1+\gamma} \geq (1+\gamma)\ell. \tag{2}$$

Let M be the set of $i \in V$ for which the packets indexed by $\mathcal{I}$ contain less than ℓ letter of E_i. From Equation (2) it follows that, for each $i \in M$,

$$(d_i - |\mathcal{I}|c'\ell^2/N)^2 \geq (\gamma\ell)^2,$$

and thus the left-hand side of Inequality (1) is at least $|M| \cdot (\gamma\ell)^2$. This and Inequality (1) implies that $|M| \leq \frac{8n}{(\gamma\ell)^2}$. Recalling that $\ell = \frac{8}{\gamma^2\beta}$, this implies that $|M| \leq \beta n/\ell \leq \beta N/\ell$ as desired. ■

We now state and prove the main theorem.

Theorem 1 *There is a scheme that, on input n, c and ϵ, has the following properties:*

(i) *A message of n letters is encoded into packets containing a total of cn letters, where each packet contains $\mathcal{O}(1/\epsilon^4)$ letters.*

(ii) *The message can be decoded from any set of packets containing in total at least $(1+\epsilon)n$ letters.*

(iii) *The run time for both the encode and decode algorithms is $\mathcal{O}(n/\epsilon^4)$.*

Proof of Theorem 1: The encoding consists of applying the constructions described in steps 1, 2, and 3, in sequence. The decoding guarantee and the run time follow from combining Lemma 4 with Lemma 3. ■

5 Concluding remarks and open problems

It would be interesting to obtain a construction similar to the one in Theorem 1 in which the packet size is smaller. It is not difficult to prove, using the Plotkin bound, that the minimum possible packet size in any erasure-resilient code with the parameters in the theorem (without any assumption on the efficiency of its encoding and decoding procedures) is at least $\Omega(\log((c-1)/\epsilon))$ for all $\epsilon \geq 1/n$, and the algebraic geometry codes show that this is essentially tight. Our construction supplies much bigger packet sizes, but has the advantage of linear encoding and decoding time.

It is possible to construct a scheme that has a theoretical run time that is polylogarithmic in $1/\epsilon$ and linear in n, using Discrete Fourier Transform methods in place of quadratic time methods for MDS codes. However, it is unlikely that using these methods in places where we use quadratic time MDS codes will be as efficient in practice.

The construction in Section 4 can be improved by using walks in expanders instead of edges, using the methods of [1]. The relevance of this method to the case of expander based error correcting codes has been observed by us (cf. [15]), and a similar remark holds here also.

Combining our technique here with the methods of Spielman in [16] we can obtain explicit, linear time encodable and decodable error correcting codes over a large alphabet, whose rate and minimum distance in the range close to the MDS bound are close to optimal. This is done by essentially the same construction described here, with an appropriate choice of the parameters. Since the last stage increases the size of the alphabet considerably, these methods cannot be useful in the binary case.

References

[1] M. Ajtai, J. Komlós, E. Szemerédi, "Deterministic Simulation in Logspace", *Proc. of the* 19^{th} *ACM Symp. on the Theory of Computing*, 1987, pp. 132-140.

[2] A. Albanese, J. Blömer, J. Edmonds, M. Luby, M. Sudan, "Priority Encoding Transmission", *Proc. of the* 35^{th} *Annual Symp. on Foundations of Computer Science*, 1994, 604-612, final version submitted to this issue.

[3] A. Albanese, J. Blömer, J. Edmonds, M. Luby, "Priority Encoding Transmission", *ICSI Technical Report No. TR-94-039*, August 1994.

[4] N. Alon, J. Edmonds, M. Luby, "Linear Time Erasure Codes With Nearly Optimal Recovery", *Proc. of the* 36^{th} *Annual Symp. on Foundations of Computer Science*, 1995, pp. 512-519.

[5] N. Alon, J. Bruck, J. Naor, M. Naor, R. Roth, "Construction of asymptotically good, low-rate error-correcting codes through pseudo-random graphs", *IEEE Transactions on Information Theory*, Vol. 38, 1992, pp. 509-516.

[6] N. Alon, F. R. K. Chung, "Explicit construction of linear sized tolerant networks", *Discrete Math.*, Vol. 72, 1988, pp. 15-19; (*Proc. of the First Japan Conference on Graph Theory and Applications*, Hakone, Japan, 1986.)

[7] N. Alon, J. H. Spencer, **The Probabilistic Method**, Wiley, 1991.

[8] L. A. Bassalygo, V. V. Zyablov, M. S. Pinsker, "Problems in complexity in the theory of correcting codes", *Problems of information transmission*, 13, Vol. 3, 1977, pp. 166-175.

[9] E. Biersack, "Performance evaluation of forward error correction in ATM networks", *Proc. of SIGCOMM '92*, Baltimore, 1992.

[10] R. E. Blahut, **Theory and Practice of Error Control Codes**, Addison Wesley, Reading, MA, 1983.

[11] A. Lubotzky, R. Phillips, P. Sarnak, "Explicit expanders and the Ramanujan conjectures", *Proc. of the* 18^{th} *ACM Symp. on the Theory of Computing*, 1986, pp. 240-246; (See also: A. Lubotzky, R. Phillips, P. Sarnak, "Ramanujan graphs", *Combinatorica*, Vol. 8, 1988, pp. 261-277).

[12] G. A. Margulis, "Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and superconcentrators" *Problemy Peredachi Informatsii*, Vol. 24, 1988, pp. 51-60 (in Russian). (English translation in *Problems of Information Transmission*, Vol. 24, 1988, pp. 39-46).

[13] F. J. Macwilliams and N. J. A. Sloane, **The Theory of Error-Correcting Codes**, North Holland, Amsterdam, 1977.

[14] M. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance", *J. ACM*, Vol. 36, No. 2, April 1989, pp. 335-348.

[15] M. Sipser and D. Spielman, "Expander codes", *Proc. of the* 35^{th} *Annual Symp. on Foundations of Computer Science*, 1994, 566-576.

[16] D. Spielman, "Linear-Time Encodable and Decodable Error-Correcting Codes" *Proc. of the* 27^{th} *ACM Symp. on the Theory of Computing* 1995, ACM Press, 388-397.