Publication 1

Esa Hyytiä, Tuomas Tirronen, and Jorma Virtamo. 2007. Optimal degree distribution for LT codes with small message length. In: Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM 2007). Anchorage, Alaska, USA. 6-12 May 2007, pages 2576-2580.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2007 proceedings.

Optimal Degree Distribution for LT Codes with Small Message Length

Esa Hyytiä[†]
Centre for Quantifiable Quality of
Service in Communication Systems,
Norwegian University of Science and Technology,
Trondheim, Norway

Tuomas Tirronen[‡]
Networking Laboratory
Helsinki University of Technology
Finland

Jorma Virtamo[‡]
Networking Laboratory
Helsinki University of Technology
Finland

Abstract—**Fountain codes provide an efficient way to transfer information over erasure channels. We give an exact performance analysis of a specific type of fountain codes, called LT codes, when the message length N is small. Two different approaches are developed. In a Markov chain approach the state space explosion, even with reduction based on permutation isomorphism, limits the analysis to very short messages, $N \leq 4$. An alternative combinatorial method allows recursive calculation of the probability of decoding after N received packets. The recursion can be solved symbolically for values of $N \leq 10$ and numerically up to $N \approx 30$. Examples of optimization results give insight into the nature of the problem. In particular, we argue that a few conditions are sufficient to define an almost optimal LT encoding.**

I. Introduction

Digital fountain coding is a relatively new concept for digital content distribution introduced by Byers et al. in 1998 [1]. The concept is based on an analogy to a fountain spraying water drops, which then are collected into a bucket. This translates into servers spraying stochastically generated pieces of data, which receivers then collect. When a sufficient number of packets is collected the file can be decoded. With good fountain codes the total size of the packets needed for decoding (on average) is close to the original size of the file, although some overhead is necessary due to the nature of these codes. An important characteristic of a digital fountain is that it is irrelevant which particular packets are received. As soon as a certain amount of the packets are received the message can be decoded (with high probability). It should be noted that codes enabling such fountain coding scenario have already existed for some time, namely the Reed-Solomon codes [2] and LDPC [3] codes to some extent. The key benefit of recently discovered fountain codes is the low computational complexity for even long message lengths N.

In this paper we derive the optimal degree distributions for the so-called LT codes by using two different approaches. The first approach is based on the observation that the decoding process constitutes a Markov chain, and we are able to write down closed form expressions for the mean number of packets required for decoding the message, as well as for the probability of decoding the message after receiving exactly N packets (the earliest time the decoding is possible). This approach, however, is limited to $N \leq 4$ due to the state space explosion, even after the state space reduction based on the permutation isomorphism. The alternative combinatorial approach, applicable for maximizing the probability of decoding the message in precisely N steps, is based on the observation that for this objective the order of arriving packets is irrelevant. This method allows us to derive the optimal degree distribution by a recursive algorithm for values up to $N \approx 20$.

These two approaches for finding the optimal degree distribution are the main contribution of this paper. The exact results give insight into the nature of the optimization problem. In particular, we find that there are just a few important conditions a good distribution has to satisfy. The described approaches, however, are limited to small values of N. Another new approach, more applicable for larger values of N, based on the simulations and importance sampling is described in [4]. The results from simulations, however, are only approximative.

II. Preliminaries

A. LT codes

LT codes proposed by Luby in [5] are the first codes fully realizing the digital fountain concept presented in [1]. They are rateless, i.e., the rate does not need to be fixed beforehand, and encoded symbols are generated on the fly [6], [7].

1) Encoding of LT code: The encoding process is extremely simple. A key element is so-called *degree distribution* defining the number of blocks in each packet. Algorithm 1 shows the encoding procedure [5]. First the degree distribution is sampled to obtain the degree d. The output packet is then generated by choosing d blocks from the original file uniformly at random and combining these blocks by bitwise XOR operation. Stopping condition for the encoder can be specified, e.g., by agreeing on the number of encoded packets beforehand, or the recipient(s) can send an acknowledgement.

2) Decoding of LT codes: Decoding is done iteratively by using information of which source blocks received packets consist of. This information needs to be included somehow in the procedure; different alternatives are available but are not discussed here. First the possible known blocks are subtracted by taking a XOR between the packet and the known block(s).

[†] "Centre for Quantifiable Quality of Service in Communication Systems, Centre of Excellence" appointed by The Research Council of Norway, funded by the Research Council, NTNU and UNINETT. Currently E. Hyytiä is with the Telecommunications Research Center Vienna (ftw.), Austria.

[‡] The work was done in the project ABI supported by the Finnish Funding Agency for Technology and Innovation (Tekes), Nokia and Ericsson.

Algorithm 1 A general LT encoding algorithm

```
repeat
    choose a degree $d$ from degree distribution $\rho(d)$.
    choose uniformly at random $d$ blocks $m(i_1), \ldots, m(i_d)$.
    send $m(i_1) \oplus m(i_2) \oplus \cdots \oplus m(i_d)$.
until enough output symbols are sent.
```

Algorithm 2 A general LT decoding algorithm

```
repeat
    while no degree-1 packets in buffer $\mathcal{B}$ do
        $\mathcal{B} \leftarrow$ received packet $-$ known blocks.
    end while
    $m(j) \leftarrow$ degree-1 packet from $\mathcal{B}$. $\{j$ discovered$\}$
    for all $c \in \mathcal{B} : c$ includes $m(j)$ do
        $c \leftarrow c \oplus m(j)$
    end for
until original message is recovered.
```

If the degree of the packet is still higher than 1 it consists of several original blocks and it is stored in a buffer. If a degree-1 packet is recovered, it is identical to an original block, i.e., a new block has been discovered. Next the newly discovered block is removed from the other buffered packets including it. If this step reveals new degree-1 packets, the decoding continues iteratively until the original message is fully decoded. Otherwise the decoder has to wait for a new packet. The decoding process is sketched in listing Algorithm 2.

Note that one input block can be just one bit or a larger chunk, the encoding and decoding processes are the same regardless. Moreover, the decoding is suboptimal. An ideal decoding equates to solving a linear system of equations, which is a computationally demanding task for large file sizes.

B. Notation

We denote the number of blocks (or input symbols) in the message by N and the degree distribution by $\rho(d)$. When convenient, we also refer to the point probabilities by p_j, i.e., $p_j = \rho(j)$. Later we will compare the performance of the optimized distributions to the following reference distributions:

Def.1 (Uniform): $p_i = 1/N, \quad i = 1, \ldots, N.$

Def.2 (Degree-1): $p_i = 1(i = 1).$

Def.3 (Binomial): $p_i {=} \frac{1}{2^N-1}\binom{N}{i},\ i{=}1, \ldots, N.$

Def.4 (Soliton): $p_1{=}\frac{1}{N}$, and $p_i{=}\frac{1}{i(i-1)},\ i{=}2, \ldots, N.$

Binomial distribution is the standard $\text{Bin}(N, \frac{1}{2})$ with event 0 excluded. It results from including every source block independently with probability $\frac{1}{2}$, discarding an empty packet.

Let the random variable Z_k denote the number of decoded input symbols after receiving the kth packet. Initially, $Z_0 = 0$ and at the end $Z_k = N$. The random variable T denotes the number of packets needed for decoding the original message,

$$T = \min_k \{k \, : \, Z_k = N\}.$$

The earliest time when the decoding process can finish is when the Nth packet arrives and thus $T \geq N$. Moreover, we let $\mathcal{P}_N$ denote the probability that a message consisting of N blocks is successfully decoded with exactly N received packets,

$$\mathcal{P}_N = \mathrm{P}\{Z_N = N\} = \mathrm{P}\{T = N\}.$$

III. Markov Chain Approach

The decoding process can be studied as a Markov chain [8]. From the receiver's point of view, the set of received and either partially or fully decoded packets denotes a state. State transition probabilities depend on the arrival probabilities of specific packets, which in turn depend on the degree distribution used in the encoding. The process ends when it has reached the absorbing state consisting of the original blocks. For example consider a file consisting of three blocks a, b, and c. When a receiver has already received a packet consisting of block a and another one of blocks b and c, the process is in state $\{a, bc\}$. The state $\{a, b, c\}$ is the absorbing state.

The number of possible distinct packets is $2^N - 1$ (i.e. the number of the subsets of a set with N elements, excluding the empty set). The number of different sets of received distinct packets is then 2^{2^N-1} (including the initial state). We call this the number of raw states. For $N = 3$ this number is 128, for $N = 4$ it is 32768, and the number grows very fast with N.

A. Reduction of the state space

The state space of the Markov chain describing the decoding process, however, needs only include the states that are irreducible in the sense that they cannot be reduced by the decoder. For instance the raw state $\{a, abc\}$ is not included as decoding reduces it to the state $\{a, bc\}$. This decreases the number of states remarkably. Further reduction is possible by observing that if a sample path of the process is modified by permuting the original blocks then the resulting sample path is isomorphic to the original one. The transition probabilities in these two sample paths are identical. This is due to the fact that in the encoding process, after the degree d is drawn, d distinct blocks to be combined by the XOR operation are drawn *randomly* from the set of N blocks. Correspondingly, any two states that can be obtained from each other by a permutation of the original blocks are isomorphic. For instance, the states $\{ad, abd, acd\}$ and $\{bd, abd, bcd\}$ are isomorphic. In contrast, $\{ad, abd, acd\}$ and $\{ad, abd, bcd\}$ are two non-isomorphic states. It is enough to include in the state space a single unique canonical representative from each class of isomorphic states.

Using the above reduction schemes the number of states can be brought down to 12 for $N{=}3$ and to 192 for $N{=}4$. Systems of this size are amenable to numerical analysis. For $N{=}5$, however, even the reduced state space has 612224 states, which is too much to be conveniently handled, and for $N{>}5$ the task is overwhelming. The 12 different states of the case $N{=}3$ are shown in Fig. 1 as the darker blocks. The lighter blocks represent intermediate states that are immediately reduced.

Still further reduction is possible by special tricks. For instance, all states that have the property that an arrival of any degree-1 packet will lead to full decoding can be aggregated to a single macro state. Transitions between states within this macro state signify just a self-transition of the macro state. In Fig. 1, the four states in the box, $\{ab, bc\}$, $\{ab, ac, bc\}$, $\{ab, ac, abc\}$, and $\{ab, ac, bc, abc\}$, constitute such a macro state.The corresponding reduced state space sizes for the cases $N = 3, \ldots, 5$ are 9, 87 and 161065, respectively.

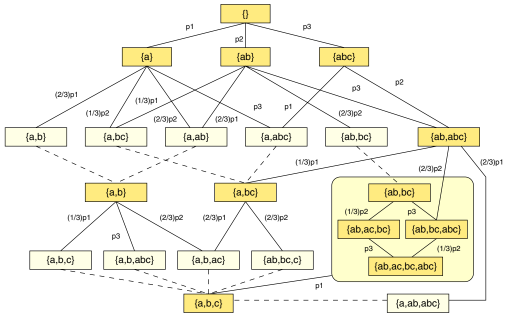

Fig. 1. State transitions in the decoding Markov chain for $n = 3$ blocks.

The state transition probability matrix $\mathbf{P}$ for the Markov process with the reduced state space can be constructed easily, e.g., by using Mathematica [9]. First, one finds the canonical representative of each class of states. Then, for each state s in the reduced state space and each possible packet p, one determines the resulting state s' in the reduced state space, and updates the transition matrix appropriately.

B. Optimizing the degree distribution

We consider two different optimization criteria for the degree distribution. A natural objective is to minimize the mean number of packets needed to successfully decode the message. Alternatively one may wish to maximize the probability of successful decoding after reception of N packets.

Using the state transition matrix $\mathbf{P}$ we can calculate the average number of sent packets needed to recover the whole original file. This Markov chain has now the state where all blocks are decoded as an absorbing state. In the example with three blocks, using the notation presented, this state is $\{a, b, c\}$. As this Markov chain is clearly finite, it can be written in the following canonical form:

$$\mathbf{P} = \begin{pmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{pmatrix},$$

where $\mathbf{Q}$ is the transition matrix between transient states, $\mathbf{R}$ represents transitions from transient states to absorbing ones and $\mathbf{I}$ is identity matrix corresponding to the absorbing states. In our case, there is just one absorbing state and the identity matrix $\mathbf{I}$ reduces to a 1×1 matrix. Now the fundamental matrix $\mathbf{M} = (\mathbf{I} - \mathbf{Q})^{\mathbf{-1}}$ is well-defined with all elements positive and represents all possible transition sequences in the transient states without going to the absorbing one. A specific element m_{ij} in $\mathbf{M}$ tells the mean number of visits in state j before absorption when starting in state i. Using the fundamental matrix, average number of steps can be calculated as follows

$$\mathrm{E}\,[T] = \boldsymbol{\pi}_0\mathbf{M}\mathbf{e}^{\mathrm{T}} = \boldsymbol{\pi}_\mathbf{0}(\mathbf{I} - \mathbf{Q})^{\mathbf{-1}}\mathbf{e}^{\mathrm{T}} = \boldsymbol{\pi}_\mathbf{0}\mathbf{A}^{\mathbf{-1}}\mathbf{e}^{\mathrm{T}}, \qquad (1)$$

where $\boldsymbol{\pi}_0 = (1\ \ 0\ \ \ldots\ \ 0)$ is the initial distribution vector corresponding to an empty system, $\mathbf{e}^{\mathrm{T}} = (1\ \ldots\ 1)^{\mathrm{T}}$, and $\mathbf{A} = \mathbf{I} - \mathbf{Q}$. Similarly we can calculate the probability of success $\mathcal{P}_N$ after receiving N packets. This is given by the probability of the absorbing state after N steps,

$$\mathcal{P}_N = \boldsymbol{\pi}_0\mathbf{P}^N\boldsymbol{\pi}_{\mathrm{abs}}^{\mathrm{T}}, \qquad (2)$$

where $\boldsymbol{\pi}_{\mathrm{abs}} = (0\ \ldots\ 0\ 1)$ represents the absorbing state.

For N=3 the reduced state space of the Markov chain consists of 9 states (with the additional state aggregation). Using Mathematica, or directly by inspection from Fig. 1, we can find the transition probability matrix

$$\mathbf{P} = \begin{pmatrix}
0 & p_1 & p_2 & p_3 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{p_1}{3} & 0 & 0 & \frac{2p_1+2p_2}{3} & \frac{p_2}{3} + p_3 & 0 & 0 & 0 \\
0 & 0 & \frac{p_2}{3} & 0 & \frac{2}{3}p_1 & \frac{p_1}{3} & p_3 & \frac{2}{3}p_2 & 0 \\
0 & 0 & 0 & p_3 & 0 & p_1 & p_2 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{2p_1+p_2}{3} & 0 & 0 & 0 & \frac{p_1+2p_2+3p_3}{3} \\
0 & 0 & 0 & 0 & 0 & \frac{p_1+p_2+3p_3}{3} & 0 & 0 & \frac{2p_1+2p_2}{3} \\
0 & 0 & 0 & 0 & 0 & \frac{p_1}{3} & \frac{p_2}{3}+p_3 & \frac{2}{3}p_2 & \frac{2}{3}p_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & p_2+p_3 & p_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}.$$

Using (1) we now have explicitly

$$\begin{aligned} \mathrm{E}\,[T] &= \boldsymbol{\pi}_0\mathbf{A}^{-1}\mathbf{e}^{\mathrm{T}} \\ &= \frac{1}{p_1} + \frac{6p_1}{p_1-3} + \frac{18p_1}{(3-p_2)(3-2p_1-p_2)} + \frac{9p_1}{2(p_1+p_2)(3p_1+2p_2)}. \end{aligned}$$

From this result it is easy to calculate the optimal weights minimizing the mean number of steps to decode the message. Similarly, using (2) one obtains an expression for $\mathcal{P}_3$, the probability of full decoding after 3 received packets, identical with (5) obtained by the approach of Section IV. Optimized results for these two different objectives are listed in Table I. Objective *MinAvg* means minimization of average number of steps $\mathrm{E}\,[T]$ needed for decoding and *MaxPr* maximizing the probability $\mathcal{P}_3$ of decoding in exactly three steps. The results of the table indicate that these two criteria are very similar; a degree distribution that is optimal for one of these criteria works very well also with respect to the other criterion.

For comparison, the table shows also results obtained with four other degree distributions introduced in Section II-B. Both uniform and degree-1 distributions perform rather poorly, the degree-1 distribution being worst. In contrast, the binomial distribution performs reasonably well, next followed by the soliton distribution (in fact, these distributions are similar).

For N=4 the reduced state space has 87 states. With the symbolic state transition matrix generated with the aid of Mathematica, the optimal weights can still be calculated and are presented in Table II together with the reference distributions. Not surprisingly, the conclusions we can draw are very similar as in the case $N = 3$. The distribution that is optimal in the *MaxPr* sense works very well also for the *MinAvg* criterion, and vice versa. Also the ranking of the other four distributions is the same as before. Binomial and soliton distributions work reasonably well with respect to both criteria (though, in relative terms, not quite as well as in the case $N = 3$), being almost equal (again the distributions are in this case very similar), while the other two are much poorer.

IV. Combinatorial Approach

When the objective of optimization is maximization of the probability $\mathcal{P}_N$ of successful decoding after reception of N encoded packets, i.e. at the first instant complete decoding is possible, the problem can be analyzed using another approach. The central observation is that in this case the order in which

TABLE I

OPTIMAL WEIGHTS AND PERFORMANCE IN THE CASE $N = 3$

	MinAvg	*MaxPr*	binomial	soliton	uniform	deg-1
p_1	0.524	0.517	3/7	2/6	1/3	1
p_2	0.366	0.397	3/7	3/6	1/3	0
p_3	0.109	0.086	1/7	1/6	1/3	0
$\mathrm{E}[T]$	4.046	4.049	4.133	4.459	4.725	5.5
$\mathcal{P}_3$	0.451	0.452	0.437	0.397	0.354	0.222

TABLE II

OPTIMAL WEIGHTS AND PERFORMANCE IN THE CASE $N = 4$

	MinAvg	*MaxPr*	binomial	soliton	uniform	deg-1
p_1	0.442	0.429	4/15	3/12	1/4	1
p_2	0.385	0.430	6/15	6/12	1/4	0
p_3	0.112	0.100	4/15	2/12	1/4	0
p_4	0.061	0.041	1/15	1/12	1/4	0
$E[T]$	5.580	5.590	6.255	6.276	7.182	8.333
$\mathcal{P}_4$	0.314	0.315	0.257	0.262	0.184	0.094

TABLE III

OPTIMAL WEIGHTS FOR MAXPR CRITERION IN CASES $N = 5, \ldots, 8$

N	5	6	7	8
p_1	0.370	0.327	0.294	0.268
p_2	0.451	0.467	0.480	0.491
p_3	0.102	0.099	0.093	0.085
p_4	0.055	0.068	0.082	0.099
p_5	0.021	0.024	0.021	0.013
p_6		0.014	0.020	0.027
p_7			0.009	0.010
p_8				0.007
$\mathrm{E}[T]$	7.111	8.613	10.097	11.565
$\mathcal{P}_N$	0.226	0.166	0.124	0.094

the packets are received is irrelevant; whatever the order of the N packets the decoding either is or is not successful at the moment when all the N packets have been received. The probability of success can then be found by a recursive combinatorial approach as detailed below.

A. Recursive algorithm

For completeness, let us now write $\mathcal{P}_n = P_n(p_1, \ldots, p_n)$. In this function we allow degree distributions with a positive probability for an empty packet and define implicitly $p_0 = 1 - \sum_{i=1}^{n} p_i$. Obviously we have $P_0 = 1$ and $P_1(p_1) = p_1$, which provide the seeds for the recursion.

In order to calculate $P_n(p_1, \ldots, p_n)$ we condition this probability on $n-m$ of the n received packets having degree 1, which happens with a probability equal to the $(n-m)$th point probability of the binomial distribution $\mathrm{Bin}(n, p_1)$. For successful decoding one must necessarily have $n - m \geq 1$, otherwise the decoding does not get started. Further, because the successful decoding after n received packets requires that no packets are wasted there must be no duplicates and all the $n-m$ degree-1 packets must be distinct. This happens with the probability $(n-1)!/m!\, n^{n-m-1}$.

Given the $n - m$ distinct degree-1 packets, we have a remaining decoding problem for the m other packets that originally are surely at least of degree 2, but whose degrees may be modified when the $n-m$ degree-1 packets are removed from the other packets in the decoding process, giving

$$\begin{aligned} P_n(p_1, \ldots, p_n) \;&=\; \sum_{m=0}^{n-1} \binom{n}{m} p_1^{n-m} (1-p_1)^m \\ &\times\; \frac{(n-1)!}{m!\, n^{n-m-1}}\; P_m(p_1^{(n,m)}, \ldots, p_m^{(n,m)}), \end{aligned} \tag{3}$$

where

$$p_j^{(n,m)} = \sum_{i=2}^{n} \frac{p_i}{1-p_1} \frac{\binom{m}{j}\binom{n-m}{i-j}}{\binom{n}{i}}, \quad j = 1, \ldots, m. \tag{4}$$

The first fraction in (4) gives the probability that a packet has degree i conditioned on that it is not a degree-1 packet. The second fraction is the probability of the event that when from an urn containing n balls (blocks), m of which are white (still unresolved), i balls are drawn without replacement (a degree-i packet is constructed) then exactly j of these balls are white (the reduced degree of the packet after removal of the resolved degree-1 packets is j).

Starting from the seeds $P_0 = 1$ and $P_1(p_1) = p_1$ the recursion equation (3) can be solved successively to yield

$$\begin{aligned} \mathcal{P}_2 \;&=\; \tfrac{1}{2}\, p_1^2 + 2p_1p_2, \\ \mathcal{P}_3 \;&=\; \tfrac{2}{9}\, p_1^3 + \tfrac{4}{3}\, p_1^2 p_2 + 2\, p_1 p_2^2 + 2\, p_1^2 p_3 + 4\, p_1 p_2 p_3, \;\ldots \end{aligned} \tag{5}$$

In principle, it is easy to let for instance Mathematica generate expressions (5) automatically to any desired order. However, the size of the expression grows fast. For $n = 4, \ldots, 10$ the number of the terms in the expression is 14, 42, 132, 429, 1430, 4862, 16796, becoming soon unmanageable. Alternatively, one can solve the recursion equation (3) numerically for a given degree distribution. This takes more time than using a pre-constructed expression of type (5) but with our C implementation run on a standard PC we can calculate the value of, e.g., $\mathcal{P}_{30}$ in a time of the order of one minute, $\mathcal{P}_{24}$ roughly in one second, and calculation of $\mathcal{P}_{20}$ takes only about 0.05 s, allowing optimization studies for systems of that size, even though a symbolic expression is beyond reach.

B. Numerical results

For the cases $N = 5, \ldots, 8$ the symbolic expressions of type (5) can be relatively easily handled and optimized numerically. The results are shown in Table III, where the estimates for $\mathrm{E}[T]$ were obtained by simulations. An example of the optimal degree distribution obtained using the numerical recursion for somewhat greater value $N = 16$ is shown in Fig. 2, with the optimum $\mathcal{P}_{16} = 0.01551$. The distribution has a strikingly irregular character. Same kind of irregularities do also show up already for smaller values of N in Table III. It should, however, be noted that the results exhibit a great degree of insensitivity with respect to some features of the degree distribution. This is demonstrated by the fact that the same maximal value $\mathcal{P}_{16} = 0.01551$ is obtained for instance with a distribution where only the probabilities $p_1 = 0.1565$, $p_2 = 0.5493$, $p_4 = 0.2095$, $p_8 = 0.0732$ and $p_{16} = 0.0115$ are non-zero.

In order to better understand the nature of this kind of insensitivity we calculated the second derivative matrix $A_{ij} =$

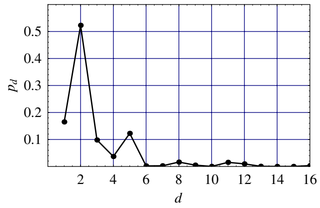

Fig. 2. Optimal degree distribution for $N = 16$.

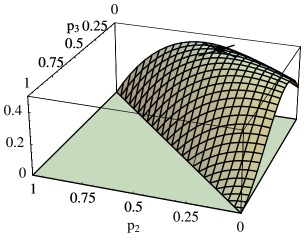

Fig. 3. $\mathcal{P}_3$ as a function of p_2 and p_3 with $p_1 = 1 - p_2 - p_3$. The principal directions at the maximum point are also shown.

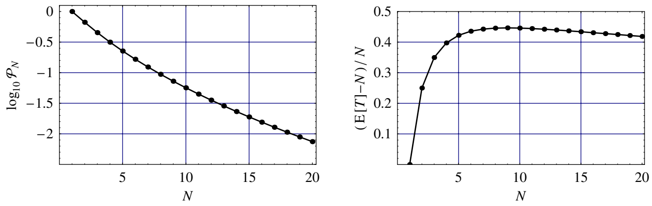

Fig. 4. Maximized success probability $\mathcal{P}_N$ (left) and the relative overhead for $N = 1, \ldots, 20$ packets.

$\partial^2 \mathcal{P}_N / \partial p_i \partial p_j$, $i, j = 2, \ldots, N$ at the optimum point with the variable p_1 eliminated by the norm condition $\sum_i p_i = 1$. The eigenvectors and eigenvalues of A determine the principal directions and curvatures in these directions. It turns out that the eigenvalues constitute a rapidly decreasing sequence, with the ratio of two consecutive eigenvalues being of the order of 10. For instance in the case $N = 3$ (A is a 2×2 matrix) the eigenvalues are $\lambda_1 = -6.97$ and $\lambda_2 = -0.71$. The $\mathcal{P}_3$ surface is illustrated in Fig. 3, where also the principal directions are shown. The function forms a ridge which is steep in one direction but flat in the direction along the top of the ridge.

The effect is much more pronounced when N is larger. For instance, for N=10 the largest and smallest eigenvalues are λ_1=-27.3 and λ_9=-2.53×10^{-6} showing that the function is extremely insensitive to changes in the direction of the last eigenvector. In fact, when the vector representing the distribution is constrained to lie on the intersection of hyperplanes with normals given by a few first eigenvectors and going through the optimal point, the $\mathcal{P}_N$ has almost the optimal value no matter what the location of the distribution vector is in the other directions. Our experiments indicate that typically three conditions of this type suffice to define a distribution which performs very close to the optimum.

Fig. 4 shows the probability of successful decoding $\mathcal{P}_N$ after N received packets as a function of N for $N = 1, \ldots, 20$. The results have been obtained by using the optimized degree distribution for each N. Also shown is the behaviour of the relative overhead $(\mathrm{E}[T] - N)/N$. This was obtained using a degree distributions optimized not for the *MinAvg* criterion but for the *MaxPr* criterion as above, and estimating $\mathrm{E}[T]$ by simulations. As is well known, for such small values of N the performance of LT codes is rather poor. The highest relative overhead 44.6 % occurs at $N = 9$. After that point a slow decline of the relative overhead starts but the codes become efficient only when N is of the order of 10000.

V. Conclusions

In this paper we have focussed on optimizing the degree distribution of LT codes when the message length N is small. We have specifically considered two optimization criteria, called *MinAvg* and *MaxPr*: 1) minimize the mean number of packets required for decoding the message, and 2) maximize the decoding probability with exactly N packets.

The decoding process constitutes a Markov chain which allows determining the optimal degree distribution. An effort was made to reduce the size of the state space as much as possible, notably by making use of the permutation isomorphism. Unavoidably though, due to the state space explosion, this approach is only feasible for very small values of N. With the second objective one can use an alternative combinatorial approach which leads to recursive equations for the success probability (recursion on N). By using this approach the optimal degree distribution can be obtained for considerably larger (still quite modest) values of N, symbolically for $N \leq 10$ and numerically up to order of $N = 30$.

One conclusion of this study is that the two optimization criteria discussed are very similar. We also found that there are just a few conditions that a good distribution has to satisfy; otherwise there is a lot of freedom in its precise definition. This suggests that in a well-chosen parametric form of the distribution just a few parameters need to be tuned in order to get nearly maximal performance.

References

[1] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *SIGCOMM*, 1998, pp. 56–67. [Online]. Available: citeseer.ist.psu.edu/byers98digital.html

[2] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM Journal of Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[3] R. G. Gallager, "Low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, January 1962.

[4] E. Hyytiä, T. Tirronen, and J. Virtamo, "Optimizing the degree distribution of LT codes with an importance sampling approach," in *RESIM 2006, 6th International Workshop on Rare Event Simulation*, Bamberg, Germany, Oct. 2006.

[5] M. Luby, "LT Codes," in *Proceedings of The 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002, pp. 271–282.

[6] D. J. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2004.

[7] P. Farrell and J. C. Moreira, *Essentials of Error–Control Coding*. John Wiley and Sons, 2006.

[8] S. M. Ross, *Introduction to Probability Models*, 7th ed. Academic Press, 2000.

[9] W. R. Inc., "Mathematica," http://www.wolfram.com/.