H-0209 November 23, 2003
Computer Science

IBM Research Report

Low-Penalty Codes for Storage Systems

Ami Tavory, Vladimir Dreizin, Shmuel Gal, Meir Feder
IBM Research Division
Haifa Research Laboratory
Haifa 31905, Israel

Low-Penalty Codes for Storage Systems

Ami Tavory, Vladimir Dreizin, Shmuel Gal, and Meir Feder
IBM's Haifa Research Labs

Abstract—**Networked storage-systems and communication systems typically use codes to protect data from failures. As reliability settings in these two types of systems are different, codes developed for storage systems can out-perform communication codes applied to storage systems. In this work we show such codes, as well as their applications.**

I. Introduction

To combat device failures, storage systems typically partition data into groups, and protect each data group by some form of an *ECC* (error-correcting code), *e.g.*, *RAID* (redundant array of independent disks) [1]. Coding theory has been extensively studied in the context of communications, where the important considerations are rate, error-correction capabilities, and computational complexity.

Coding theory in the context of storage systems, might also have other considerations:

• *Recovery penalty* and *update penalty* – When bits are updated, or when they need be recovered, other bits need be accessed. In most communication-based codes, the number of accessed bits is proportional to the group size and not to the actual number of errors. In the setting of storage systems, this influences the number of devices actively engaged in reliability-related operations. When communication-based codes are used for high-performance storage-systems, small group-sizes are therefore typically used. This, however, can be shown to lead to a decrease in the *MTTDL* (mean time to data loss).

• Importance of block-error probability – In many communication settings, the guarantee that each bit can be recovered with high probability, is sufficient (*i.e.*, a low bit error-probability guarantee). Parts of a corrupted message, which the code does not recover, can be retransmitted. In most storage applications, the guarantee should be that, with high probability, all bits can be recovered (*i.e.*, a low block error-probability guarantee). Bits from a data group that have been lost due to a device failure, and cannot be corrected by the code, are irrevocably lost.

• Dynamic data-group location – In communication settings, there is little question on data-group location, as once a group is encoded, it is simply transmitted. In storage systems, the location of data groups among devices is dynamic, as data can be corrected and migrated from failed devices to replacement devices. This blurs the distinction between coding theory and algorithm design.

• Difference in access types – In communication systems, the types of access to message bits are usually not considered. In storage systems, the cost of reading or writing a bit, is typically not the same as that of an *RMW* (read-modify-write) operation.

Email:{`tavory,dreizin,gals`}`@il.ibm.com,meir@eng.tau.ac.il`

In this work we study *LDGM* (low-density generation matrix) codes which are powerful, efficient, low-complexity codes, having low recovery and update penalties. We show how these codes can be applied for storage systems in two settings. We first show a simple application to large groups of *reference data*, which are data rarely modified. We then analyze a *random sparing* dynamic layout based on these codes. We then preform new analysis on repetition codes (*i.e.*, multi-way mirroring), which allows to estimate the MTTDL of a group mirroring sub-systems (as opposed to a single mirroring sub-system).

We prove the results discussed in the paper, and show software-simulations results verifying the analysis.

Related Work. Reliability has been extensively studied for communication channels, most relevantly the erasure channel and the binary symmetric channel [2], [3]. Bounds on channel capacity were found for uniform error-protection [4] and unequal error-protection [5], [6]. Well-known code-families for uniform error-protection are Hamming codes [7] and BCH codes [7] with some of their variants, *e.g.*, Reed-Solomon codes [8]. More recently, low-complexity codes have been studied [9], in particular, capacity-achieving codes, *e.g.*, Turbo codes [10], and LDPC and LDGM codes [11], [12], [13].

Reliability in storage systems was originally studied in the context of small-capacity systems [14], [15], [1], and in conjunction with performance improvement via parallelism, *e.g.*, RAID. The schemes were later extended in some directions. Concatenated codes were studied, *e.g.*, two-dimensional codes [16], and new RAID levels [17]. Questions on coding-group placement within devices were studied, *e.g.*, various distributed striping and sparing techniques [18], [19], [20], [21], [22], [23]. Effects of physical device-topologies were studied [24]. Storage reliability via coding was extended in the direction of disaster recovery as well [25].

The important idea of hierarchical protection of data based on data activity, was shown in work on HP-AutoRAID [26], a work to which ours is an extension. Differential coding based on data activity was studied in the context of very large, concrete systems [27], [28].

Later work in storage reliability has considered larger-capacity systems, and therefore a failure model taking into account multiple simultaneous errors. Some excellent analytical work can be found in [29], [30], [31]. New work oriented toward overall system design and implementation and the conjunction of several system aspects (*e.g.*, security, load-balancing, and meta-data location) can be found in a series of papers on OceanStore [27], [32], [28].

Definitions and Notations. We consider a system composed of n storage devices. Each devices is composed of c equal-sized blocks, and has a read/write bandwidth of r blocks per time unit. We model the devices' failure laws as being distributed i.i.d. exponentially with mean $\frac{1}{\lambda}$ [1]. The failure of a device corresponds to the erasure of all data on it. Given a systematic erasure-code composed of m data bits and $\alpha \cdot m$ check bits, we define the storage rate as $\frac{1}{1+\alpha}$. Given a component C, its *reliability function*, $R_C(t)$, is the probability that it will be functioning at time t [39]. The *system MTTDL* is the mean time to data loss in the system.

For any p, we use $[p]$ to denote the set $\{1, \ldots, p\}$. We use $I(\cdot)$ to denote the indicator function.

Paper Organization. The remainder of the paper is organized as follows. In Section II we describe the variant of LDGM codes we propose for storage systems. In Section III we discuss a simple application to reference data. In Section IV we discuss an application to a random-placement technique. In Section V we analyze multi-way mirroring.

II. LDGM Codes for Storage-Systems

In this section we discuss a variation of *LDGM* (low-density generation-matrix) codes. *LDPC* (low-density parity-check) and LDGM codes were extensively studied in the field of communication [33], [12], [11], [34], [35], due to their low computational-complexity and nearly-optimal rate. We propose a variation suitable for storage systems. The use of LDGM codes for storage was independently proposed previously [27], [32], [28], [35], for reasons of computational-complexity alone, and under the caveat that they are probabilistic.

Consider two coding alternatives with equal storage rates. In the first, all bits are encoded as a single group. In the second, the bits are partitioned into groups, and each group is coded. By the law of large numbers, it is clear that the former alternative is superior in terms of loss probability to the latter (whose loss probability approaches 1, as the number of groups grows). The former alternative might have other drawbacks. Reed-Solomon coding [7], for example, would necessitate accessing nearly all bits for the recovery of nearly any failure configuration. We are interested in codes for large data-groups which have a recovery penalty determined by the number of errors which took place, rather than being always proportional to the group size.

The complementary question, of bounded update penalty (defined similarly), was previously addressed in an excellent paper [31], where an inverse relationship between recovery locality and update locality was also shown.

This section is organized as follows. In Subsection II-A we give the relevant definitions and notations of LDGM codes. We next show two codes with low block error-probability, good recovery locality, and low computational-complexity. In Subsection II-B we show the first code, which has a sub-optimal storage rate. Using this code, we show in Subsection II-C a code with an asymptotically-optimal storage rate, but higher update penalty. In Subsection II-D we show simulation results.

A. Definitions and Notations

In this subsection we give the relevant definitions of LDGM codes. We consider a recursive LDGM code C composed of p levels, $C^{(1)}, \ldots, C^{(p)}$, each of which can be *un-augmented* or *augmented*. In Sub-subsection II-A.1 we give the definitions for a single level. In Sub-subsection II-A.2 we give the definitions for the recursive code.

A.1 A single level

A un-augmented level of an LDGM code is described by a *Tanner-graph*, which is a bipartite graph

$$G = (V, E) = (L \dot{\bigcup} R, E). \tag{1}$$

Let $m = |L|$. Each node in L represents a data bit; each node in R represents a parity bit. The value of any node v is denoted by *value* (v). An example of such a graph is shown in Figure 1. For any parity node $v^r \in R$, its *left-neighbor* set, $\overleftarrow{V}(v^r) \subseteq L$, is the set

$$\overleftarrow{V}(v^r) = \left\{ v^\ell \in L \mid (v^\ell, v^r) \in E \right\}. \tag{2}$$

For any data node $v^\ell \in L$, its *right-neighbor* set, $\overrightarrow{V}(v^\ell)$, is defined similarly. The parity bit corresponding to v^r is set to be the *XOR* (exclusive or) of the data bits corresponding to the nodes in $\overleftarrow{V}(v^r)$, *i.e.*,

$$\mathit{value}(v^r) = \bigoplus_{v^\ell \in \overleftarrow{V}(v^r)} \mathit{value}(v^\ell). \tag{3}$$

The edges in E thus represent parity constraints. The graph is *sparse*, *i.e.*, $|E| = O(|L|) = O(|R|)$.

The average left-node and right-node degrees are

$$\begin{aligned} a_\ell &= \frac{\sum_{v^\ell \in L} \left|\overrightarrow{V}(v^\ell)\right|}{|L|}, \\ a_r &= \frac{\sum_{v^r \in R} \left|\overleftarrow{V}(v^r)\right|}{|R|}, \end{aligned} \tag{4}$$

respectively. The fractions of edges whose left and right node-degree is i, are

$$\begin{aligned} \lambda_i &= \frac{\left|\left\{ e = (v^\ell, v^r) \in E \mid \left|\overrightarrow{V}(v^\ell)\right| = i \right\}\right|}{|E|}, \\ \rho_i &= \frac{\left|\left\{ e = (v^\ell, v^r) \in E \mid \left|\overleftarrow{V}(v^r)\right| = i \right\}\right|}{|E|}, \end{aligned} \tag{5}$$

respectively. These degrees are usually put into the form of generating functions

$$\begin{aligned} \lambda(x) &= \sum_{i=1}^{\infty} \lambda_i x^{i-1}, \\ \rho(x) &= \sum_{i=1}^{\infty} \rho_i x^{i-1}. \end{aligned} \tag{6}$$

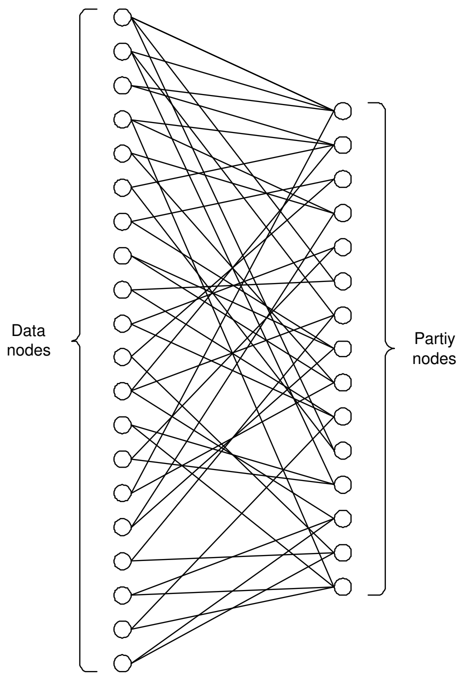

Fig. 1. A Tanner graph.

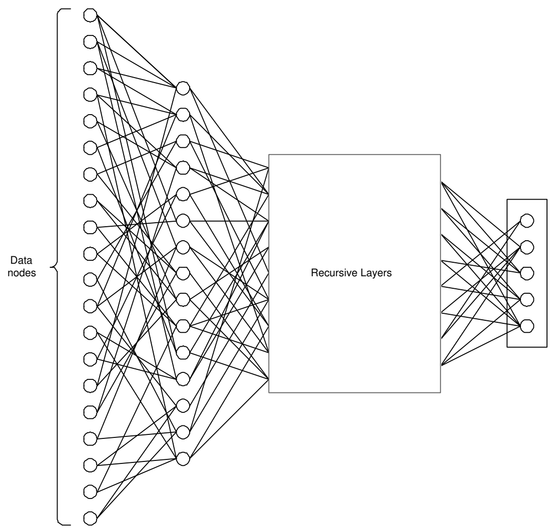

Fig. 2. A recursive LDGM encoding.

Note that $\lambda(x)$ and $\rho(x)$ completely define the un-augmented level.

Assume that all nodes in R are valid, and that some nodes in L have failed. The recovery process is iterative. In each step, a node $v^r \in R$ is chosen s.t. a single node $v^\ell \in \overleftarrow{V}(v^r)$ is failed, and the value of v^ℓ is then corrected using (3). Of course, the recovery process terminates prematurely if such a node v^r cannot be found. We denote by $\mathbf{P}_{\text{block}}(L)$ the block error-probability, which is the probability that the recovery process terminates while some bits in L have not been recovered. We denote by $\mathbf{P}_{\text{bit}}(L,\alpha)$ the α-bit error probability, which is the probability that the recovery process terminates while a fraction of at least α bits in L has not been recovered.

Check below whether decreases e-m or o of 1

Check - before omega or within omega

We will use the following theorem from [11].

Theorem 1: Assume a fraction of δ nodes from L originally fail.

1. In the family of codes for which

$$\forall_{x\in(0,\delta]}\delta\cdot\lambda(1-\rho(1-x))<x, \tag{7}$$

a fraction of $1-o(1)$ of the codes have

$$\forall_{\alpha<1}\mathbf{P}_{\text{bit}}(L,\alpha)=o\left(e^{-\alpha m}\right). \tag{8}$$

2. In the family of codes for which (7) holds and

$$\lambda_1=\lambda_2=0, \tag{9}$$

a fraction of $1-o(1)$ of the codes have

$$\mathbf{P}_{\text{block}}(L)=o\left(e^{-\Omega(m)}\right). \tag{10}$$

As seen in Theorem 1, a code for which (9) does not hold, does not guarantee the recovery of all nodes. On the other hand, it was shown in [11] that if (9) holds, then the storage rate is suboptimal. For this reason, an augmented level is commonly used. In an augmented level, the nodes in L are protected by both an LDGM code for which (9) does not hold, and by an expander-based code [36]. The LDGM code corrects all but a negligible number of failed nodes. The expander-based code corrects the rest.

A.2 The recursive code

The recovery process of $C^{(i)}$, described in Sub-subsection II-A.1, requires that all the nodes in R be valid. Since these nodes can fail, a recursive code is used. This is shown in Figure 2. In the recursive code, each level's R is taken to be the next level's L. A sequence of p levels is built, $C^{(1)},\ldots,C^{(p)}$. The last level, $C^{(p)}$, is composed of a non-recursive code (*e.g.*, Reed-Solomon).

Note that in general, it is not required that the $C^{(i)}$ be of the same type. Typically, however, a recursive LDGM code is composed of same-type levels (except for the last). Let the storage-rate of a level be denoted by β, and let $\beta'=\frac{1-\beta}{\beta}$. Let the number of original data bits be n. Level i is then an LDGM code with $\beta'^{i-1}n$ left nodes and $\beta'^{i}n$ right nodes. Assume that p is set s.t. $\beta'^{p-1}n\approx\sqrt{n}$. It follows that the total number of redundant nodes is

$$n\sum_{i\in[p]}\beta'^{i}\approx \tag{11}$$

$$\left(n-\beta'\sqrt{n}\right)\frac{\beta'}{1-\beta'}=\left(n-\frac{1-\beta}{\beta}\sqrt{n}\right)\frac{1-\beta}{2\beta-1}.$$

B. A Truncated Right-Regular Code

$\boxed{\textbf{m or n as the number of data bits}}$

In this subsection we define a right-regular code which is a variation of [13], and analyze its performance. The code is *truncated*, in its power series $\lambda(x)$ having a 0 coefficient of the x term. This code has low block error-probability, good recovery locality, and low computational-complexity. We use this code in Subsection II-C as well.

Definition 1: Let $\hat{C}^{(i)}$ be an un-augmented level of an LDGM code, defined by the generating functions:

$$\begin{aligned}\hat{\lambda}(x) &= \frac{\sum_{k=2}^{\hat{q}-1}\binom{\hat{\alpha}}{k}(-1)^{k+1}x^k}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}, \qquad (12)\\ \hat{\rho}(x) &= x^{\hat{a}-1},\end{aligned}$$

where $\hat{a} \geq 2$ and $\hat{q} \geq 3$, are arbitrary integers, and $\hat{\alpha} = 1/(\hat{a}-1)$. Let the average left-node degree be denoted by $\hat{a}_\ell$ (the average right-node degree is obviously $\hat{a}$).

Note that $\hat{\lambda}(x)$ is a normalized sum of x^i terms of the Taylor expansion of $1-(1-x)^{\hat{\alpha}}$, for $2 \leq i \leq \hat{q}$.

The main result in this subsection is the following.

Theorem 2: Let

$$\hat{\delta}_{\max} = \frac{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}{1-\hat{\alpha}}. \qquad (13)$$

Then we have the following attributes of $\hat{C}^{(i)}$:

1. Let δ denote the fraction of errors in L.

$$\delta \leq \hat{\delta}_{\max} \Rightarrow \mathbf{P}_{\text{block}}(L) = o\left(e^{-\Omega(m)}\right). \qquad (14)$$

2. • If the number of failed data-nodes is $|L_f|$ and the number of accessed data-nodes is $|L_a|$, then

$$|L_a| \leq \hat{a}\,|L_f|. \qquad (15)$$

• Let $\hat{C}$ be a recursive LDGM code composed of $\hat{C}^{(i)}$s. Then the expected number of accesses per single (data or parity) failed-node recovery approaches $\hat{a}+1$.

3. • If the number of modified data-nodes is $|L_u| \ll |L|$ and the number of accessed parity-nodes is $|R_a|$, then

$$|R_a| \approx |R| - |R|\left(1-\frac{\hat{a}_\ell\,|L_u|}{\hat{a}\,|R|}\right)^{\hat{a}} \qquad (16)$$

• Let $\hat{C}$ be a recursive LDGM code composed of $\hat{C}^{(i)}$s, protecting n original data-bits. Let a single data-node be modified. Then the ratio between the expected number of accessed nodes and total number of redundant nodes is

$$o\left(\frac{1}{n^{\Omega(1)}}\right), \qquad (17)$$

where n is the number of original data-bits.

4. The storage rate of the code is greater than

$$1-\frac{2\left(1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}\right)}{1-\frac{2}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}. \qquad (18)$$

5. The ratio between the fraction of errors corrected by $\hat{C}^{(i)}$, and the fraction corrected by an optimal code of the storage rate in (18), is

$$\frac{1}{2}-\frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}. \qquad (19)$$

6. The computational complexity of encoding and decoding is linear.

For much of the proof, we modify proofs from [11] and [13].

We first prove item 1 of Theorem 2,

Proof: To apply item 2 of Theorem 1, we first must show that the degree distributions are valid ones. To do so, it is sufficient to show that the coefficients of $\hat{\lambda}(x)$ are positive, and that $\hat{\lambda}(1) = \hat{\rho}(1) = 1$.

To show the positivity of the coefficients, note that

$$\begin{aligned}\binom{\hat{\alpha}}{k} = \frac{\hat{\alpha}(\hat{\alpha}-1)\cdots(\hat{\alpha}-k+1)}{k!} = \qquad (20)\\ (-1)^{k-1}\frac{\hat{\alpha}}{k}\left(1-\frac{\hat{\alpha}}{k-1}\right)\cdots\left(1-\frac{\hat{\alpha}}{2}\right)(1-\hat{\alpha}).\end{aligned}$$

To show that the sum of the coefficients is 1, note that

$$\begin{aligned}\hat{\lambda}(1) &= \frac{\sum_{k=2}^{\hat{q}-1}\binom{\hat{\alpha}}{k}(-1)^{k+1}1^k}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \overset{(a)}{=} 1, \qquad (21)\\ \hat{\rho}(1) &= 1^{\hat{a}-1} = 1,\end{aligned}$$

where (a) follows from the fact that, by induction, [11]

$$\sum_{k=1}^{\hat{q}-1}\binom{\hat{\alpha}}{k}(-1)^{k+1} = 1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}. \qquad (22)$$

We now show that the condition of item 2 in Theorem 1 holds.

Expanding $\hat{\lambda}(1-\hat{\rho}(1-x))$, we have

$$\begin{aligned}&\hat{\lambda}(1-\hat{\rho}(1-x)) \overset{(a)}{=} \qquad (23)\\ &\frac{\sum_{k=2}^{\hat{q}-1}\binom{\hat{\alpha}}{k}(-1)^{k+1}(1-\hat{\rho}(1-x))^k}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \overset{(b)}{\leq}\\ &\frac{\sum_{k=2}^{\infty}\binom{\hat{\alpha}}{k}(-1)^{k+1}(1-\hat{\rho}(1-x))^k}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \overset{(c)}{=}\\ &\frac{1-(1-(1-\hat{\rho}(1-x)))^{\hat{\alpha}}-\hat{\alpha}(1-\hat{\rho}(1-x))}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \overset{(d)}{=}\\ &\frac{1-\left(1-\left(1-(1-x)^{\hat{a}-1}\right)\right)^{\hat{\alpha}}-\hat{\alpha}\left(1-(1-x)^{\hat{a}-1}\right)}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \overset{(e)}{=}\\ &\frac{x-\hat{\alpha}+\hat{\alpha}(1-x)^{\frac{1}{\hat{\alpha}}}}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}, \qquad (24)\end{aligned}$$

where in the above, (a) follows from (12), (b) follows from the fact that by (20), for $y \leq 1$,

$$\binom{\hat{\alpha}}{k}(-1)^{k+1}(1-y)^k \geq 0, \tag{25}$$

(c) follows from the fact that the Taylor expansion of $1-(1-y)^{\hat{\alpha}}$ is

$$1-(1-y)^{\hat{\alpha}} = \sum_{k=1}^{\infty}\binom{\hat{\alpha}}{k}(-1)^{k+1}y^k, \tag{26}$$

and (d) and (e) follow from (12).

We also note for the above, that it was shown in [13] that

$$\boxed{\textbf{Check here, made serious changes}}$$

$$\breve{\lambda}(x) = \frac{\sum_{k=1}^{\hat{q}-1}\binom{\hat{\alpha}}{k}(-1)^{k+1}x^k}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}} \tag{27}$$

converges in the relevant range of x, and so

$$\hat{\lambda}(x) = \frac{\breve{\lambda}(x)\left(1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}\right)-\hat{\alpha}x}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \tag{28}$$

converges as well.

Using the above expansion, we now have that for $x \in (0, \hat{\delta}_{\max}]$,

$$\begin{aligned}
&\delta\hat{\lambda}(1-\hat{\rho}(1-x)) \overset{(a)}{\leq} \\
&\frac{\delta\left(x-\hat{\alpha}+\hat{\alpha}\cdot(1-x)^{\frac{1}{\hat{\alpha}}}\right)}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} \overset{(b)}{\leq} \\
&\frac{\delta(1-\hat{\alpha})x}{1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}} = \\
&\frac{\delta}{\hat{\delta}_{\max}}x \leq x,
\end{aligned} \tag{29}$$

where, in the above, (a) follows from (24), and (b) follows from the fact that

$$\forall_{0\leq x\leq 1} x-\hat{\alpha}+\hat{\alpha}\cdot(1-x)^{\frac{1}{\hat{\alpha}}} \leq x(1-\hat{\alpha}). \tag{30}$$

■

We now prove item 2 of Theorem 2.

Proof: The first part follows easily from description, in Sub-subsection II-A.1, of the recovery process. For the second part, assume that the number of original data-bits is n. Note that the number of accesses required for a failed node v is at most

$$\left\{\begin{array}{lll}\sqrt{n} & , & v \in \hat{C}^{(p)} \\ \hat{a} & , & \text{otherwise}\end{array}\right. . \tag{31}$$

Let n' be the total number of nodes not in $\hat{C}^{(p)}$. The average number of accessed nodes is

$$\frac{\hat{a}n'+\sqrt{n}\cdot\sqrt{n}}{n'+\sqrt{n}}. \tag{32}$$

Using (11), we get that the average number of accessed nodes is at most

$$\hat{a}+\frac{2\beta-1}{\beta} < \hat{a}+1, \tag{33}$$

since $\frac{2\beta-1}{\beta} < 1$ for $\frac{1}{2} < \beta < 1$. ■

We now prove item 3 of Theorem 2.

Proof: To prove the first part, we use the differential-equation approach from [11]. Some edges emanate from the $L_m \subset L$ modified nodes. Let the number of such edges be m'; let the edges be $e_1, \ldots e_{m'}$.

Assume a process of m' steps. Let each step take $\Delta t = \frac{1}{m'}$ time. Let r_t denote the average number of un-accessed parity-nodes at time t. At step i, the node v_i^r is accessed, where $v_i^r \in R$ is the terminating node of e_i. The value of $r_{i\Delta t}$ is updated, if necessary. It can be shown that the difference-equation system for r_t is:

$$\begin{aligned}
r_{t+\Delta t}-r_t &= -\frac{\hat{a}r_t}{\hat{a}|R|-\frac{t}{\Delta t}}, \\
r_0 &= |R|.
\end{aligned} \tag{34}$$

Manipulating and taking $\Delta t \to 0$, we get the differential-equation system:

$$\begin{aligned}
\frac{dr_t}{dt} &= -\frac{m'\hat{a}r_t}{\hat{a}|R|-tm'}, \\
r_0 &= |R|.
\end{aligned} \tag{35}$$

Solving for r_t and setting $t=1$, we get

$$r_1 = |R|\left(1-\frac{m'}{\hat{a}|R|}\right)^{\hat{a}}. \tag{36}$$

It remains to approximate m'. This random variable is distributed hyper-geometrically. For $|L_m| \ll |L|$, $m' \approx \hat{a}_l L_m$.

We now prove the second part. On the average, when updating a left node, $\hat{a}_l$ right nodes should be updated. For each level, the maximal number of nodes updated is not larger than the number of parity nodes in the level. It follows that for any $j \in [p]$, the average number of accessed nodes is bounded by

$$\sum_{i=0}^{j}\hat{a_l}^{i+1}+\sum_{i=j+1}^{\hat{p}}\beta^{i+1}n. \tag{37}$$

Minimizing by j yields that the number of accessed nodes is

$$\begin{aligned}
&\approx n^{\log_{\frac{\hat{a}_l\beta'}{1-\beta'}}(\hat{a}_l)}\left(\frac{\hat{a}_l}{\hat{a}_l-1}+\frac{1-\beta'}{2\beta'-1}\right) \\
&-\sqrt{n}\frac{1-\beta'}{2\beta'-1}-\frac{\hat{a}_l}{\hat{a}_l-1}.
\end{aligned} \tag{38}$$

The combination of (11) and (38) shows that the ratio of the number of accessed nodes to the number of redundant nodes is

$$o\left(n^{\log_{\frac{\hat{a}_l\beta'}{1-\beta'}}(\hat{a}_l)-1}\right). \tag{39}$$

Note that since $\frac{\beta'}{1-\beta'} > 1$, the power of n in (39) is negative.

■

We now prove item 4 of Theorem 2.

Proof: By definition, the storage rate of the code is the ratio between the size of the left-node set and the total number of nodes.

$$r = \frac{|L|}{\left|L \dot{\bigcup} R\right|} = \frac{1}{1+\frac{|R|}{|L|}} \geq 1 - \frac{|R|}{|L|} = \tag{40}$$

$$1 - \frac{\int_0^1 \hat{\rho}(x)\,dx}{\int_0^1 \hat{\lambda}(x)\,dx} \overset{(a)}{=}$$

$$1 - \frac{2\left(1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}\right)}{1-\frac{2}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}$$

where (a) follows from the fact that

$$\sum_{k=1}^{\hat{q}-1}\binom{\hat{\alpha}}{k}\frac{(-1)^{k+1}}{k+1} = \frac{\hat{\alpha}-\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}}{\hat{\alpha}+1}, \tag{41}$$

$$\int_0^1 \hat{\lambda}(x)\,dx =$$

$$\frac{\hat{\alpha}}{\hat{\alpha}+1}\frac{1-\frac{2}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}{2(1-\frac{\hat{q}}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha})},$$

$$\int_0^1 \hat{\rho}(x)\,dx = \frac{\hat{\alpha}}{\hat{\alpha}+1}.$$

■

We now prove item 5 of Theorem 2.

Proof: For any storage rate r, an optimal code could then correct a fraction of $1-r$ errors [7]. The ratio of correction capabilities is therefore,

$$\frac{\hat{\delta}_{\max}}{1-R} \geq \tag{42}$$

$$\frac{1-\frac{2}{\hat{\alpha}}\binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1}-\hat{\alpha}}{2(1-\hat{\alpha})} \overset{(a)}{=}$$

$$\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})},$$

where in the above, (a) uses the fact that [13]

$$\exists_c \forall_{\hat{q}\geq 2}\forall_{\hat{\alpha}<\frac{1}{2}} \frac{c\hat{\alpha}}{\hat{q}^{\hat{\alpha}+1}} \leq \binom{\hat{\alpha}}{\hat{q}}(-1)^{\hat{q}+1} \leq \frac{\hat{\alpha}}{\hat{q}^{\hat{\alpha}+1}}. \tag{43}$$

■

Item 6 of Theorem 2 is a known property of LDGM codes. It follows from the sparseness of G.

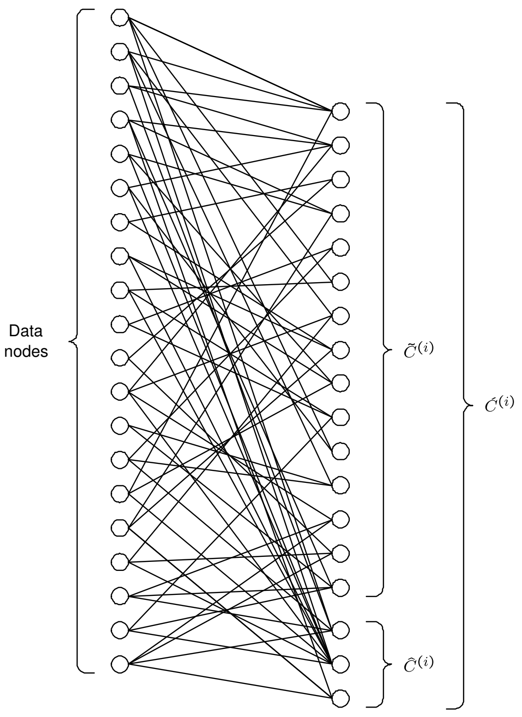

Fig. 3. The augmented level-code.

C. An Augmented Right-Regular Code

In this subsection we define an augmented right-regular code based on [13] and Subsection II-B. This code has low block error-probability, good recovery locality, and low computational-complexity. Its storage rate is asymptotically optimal; its update penalty is higher than the code from Subsection II-C.

We will require the following code from [13].

Definition 2: Let $\tilde{C}^{(i)}$ be an un-augmented level of an LDGM code, defined by the generating functions:

$$\tilde{\lambda}(x) = \frac{\tilde{\alpha}\sum_{k=1}^{\tilde{q}-1}\binom{\tilde{\alpha}}{k}(-1)^{k+1}x^k}{\tilde{\alpha}-\tilde{q}\binom{\tilde{\alpha}}{\tilde{q}}(-1)^{\tilde{q}+1}}, \tag{44}$$

$$\tilde{\rho}(x) = x^{\tilde{a}-1},$$

where $\tilde{a} \geq 2$ and $\tilde{q} \geq 2$, are arbitrary integers, and $\tilde{\alpha} = \frac{1}{\tilde{a}-1}$.

We now define a code that combines the previous two.

Definition 3: Let $\acute{C}^i = \acute{C}^i(\epsilon)$ be an LDGM level defined by $\tilde{C}^i$, and augmented by $\hat{C}^i$ (from Subsection II-B) coded at storage rate $\frac{2\epsilon}{\frac{1}{2}-\frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}$.

A level of this code is shown schematically in Figure 3. This idea is similar to augmenting an LDGM code by an Expander-based code [11].

The main result in this subsection is the following.

Theorem 3: Let $\tilde{\alpha}$ and $\tilde{q}$ be defined as in Definition 2. Let

$$\acute{\delta}_{\max} = \frac{\tilde{\alpha} - \tilde{q}\binom{\tilde{\alpha}}{\tilde{q}}(-1)^{\tilde{q}+1}}{\tilde{\alpha}}. \tag{45}$$

Then we have the following attributes of $\acute{C}^{(i)}$:

1. Let the fraction of errors which occurred be δ.

$$\delta \leq \acute{\delta}_{\max} \Rightarrow \mathbf{P}_{\text{block}}(L) = o\left(e^{-\Omega(m)}\right). \tag{46}$$

2. If the number of failed data-nodes is $|L_f| \gg 1$ and the number of accessed data-nodes is $|L_a|$, then

$$|L_a| \lessapprox \tilde{a}\,|L_f|. \tag{47}$$

3. • If the number of modified data-nodes is $|L_m| \ll |L|$ and the number of accessed parity-nodes is $|R_a|$, then

$$|R_a| \lessapprox |R| \left(1 - \left(1 - \frac{\tilde{a}_\ell\,|L_m|}{\tilde{a}\,|R|}\right)^{\tilde{a}} + \frac{2\epsilon}{\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}\epsilon\right) \tag{48}$$

• Let $\acute{C}$ be a recursive LDGM code protecting n original data-bits, whose each layer is is $\acute{C}^{(i)}$. Let a single data-node be modified. Then the ratio between expected accessed parity-nodes and total parity-nodes is

$$o\left(\frac{1}{n^{\Omega(1)}}\right). \tag{49}$$

4. The storage rate of the code is greater than

$$\left(1 - \frac{\tilde{\alpha} - \tilde{q}\binom{\tilde{\alpha}}{\tilde{q}}(-1)^{\tilde{q}+1}}{\tilde{\alpha} - \binom{\tilde{\alpha}}{\tilde{q}}(-1)^{\tilde{q}+1}}\right) \frac{1}{1 + \frac{2\epsilon}{\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}}. \tag{50}$$

5. The ratio between the fraction of errors corrected by the code and the fraction corrected by an optimal code of the same storage rate is

$$1 - \frac{1}{q^{\tilde{\alpha}+1}} \frac{1}{1 + \frac{2\epsilon}{\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}\epsilon}. \tag{51}$$

6. The computational complexity of encoding and decoding is linear.

We first prove items 1 of Theorem 3.

Proof: Using Item 1 of Theorem 1, it can be shown that $\tilde{C}^{(i)}$ corrects all but a negligible fraction of errors ϵ' s.t. $\epsilon' \ll \epsilon$. Using Item 2 of Theorem 1 and the proof of Theorem 2, it follows that $\hat{C}^{(i)}$ corrects at least an ϵ fraction of errors. ■

Items 2 and 3 of Theorem 3 can be proved as in Theorem 2.

We now prove items 4 and 5 of Theorem 3.

Proof: Let $\left|\tilde{L}\right|$ and $\left|\tilde{R}\right|$ be the sizes of the data and parity node-sets in the code $\tilde{C}^{(i)}$. The storage rate of $\acute{C}^{(i)}$ is

$$\begin{aligned} &\frac{\left|\tilde{L}\right|}{\left|\tilde{L}\right| + \left|\tilde{R}\right| + \frac{2\epsilon}{\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}\left|\tilde{L}\right|} = \\ &\frac{\left|\tilde{L}\right|}{\left|\tilde{L}\right| + \left|\tilde{R}\right|} \frac{\left|\tilde{L}\right| + \left|\tilde{R}\right|}{\left|\tilde{L}\right| + \left|\tilde{R}\right| + \frac{2\epsilon}{\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}\left|\tilde{L}\right|} \overset{(a)}{\geq} \\ &\frac{\left|\tilde{L}\right|}{\left|\tilde{L}\right| + \left|\tilde{R}\right|} \frac{1}{1 + \frac{2\epsilon}{\frac{1}{2} - \frac{1}{q^{\hat{\alpha}+1}(1-\hat{\alpha})}}}, \end{aligned} \tag{52}$$

where (a) follows from Definition 3. The proof now follows by applying bounds from [13] on the storage rate of $\tilde{C}^{(i)}$.

Item 6 of Theorem 3 follows as in Theorem 3. ■

D. Simulation Results

In this subsection we show simulation results for a single-layer of the truncated right-regular LDGM codes discussed in Subsection II-B:

• Figure 4 shows the single-layer storage-rate and maximal fraction of fixable nodes as a function of left and right edge-degrees (derived in (18) and (13) respectively). As noted, the rate of this code is sub-optimal, necessitating the use of the augmented code from Subsection II-C.

• We have performed 500 tests on the block error-probability. In each test a random code with $m = 10000$ left nodes, $\hat{q} = 4$, $\hat{a} = 4$, $\hat{\alpha} = \frac{1}{3}$, and 8236 right nodes was first built. Then, 10^6 iterations were performed. In each iteration 3900 left nodes were initially set to be corrupted, and were then attempted to be fixed. The code succeeded in every single iteration of every single test.

• Figures 5 and 6 show the number of updated nodes as a function of the fraction of modified left nodes, for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes. Note that as predicted, simulation results coincide with the expected value, given by (16). For small fractions of left nodes being updated, the number of update-IOs can be approximated as the number of updated left nodes times average left degree (a_l).

• Figure 7 shows the number of nodes, accessed by the recovery procedure, as a function of the fraction of failed left nodes, for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes. Note that the assumption of a accesses per each failed node gives the upper bound on the number of accessed nodes, as shown in (15).

III. Applications to Reference Data

The first simple application of the LDGM code is for protecting large amounts of reference data (*i.e.*, data which is

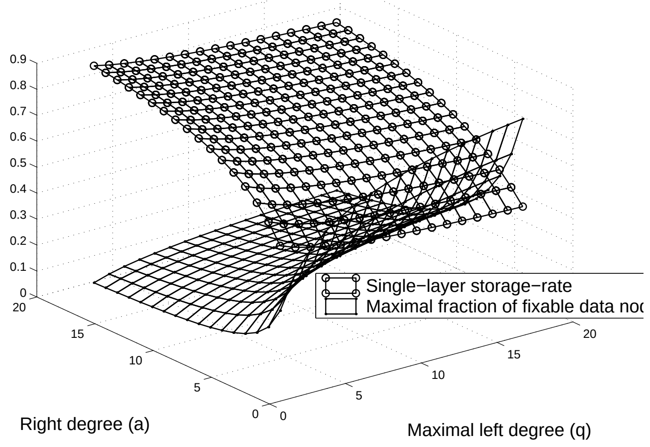

Fig. 4. Rates and error-correction capabilities as a function of edge degrees.

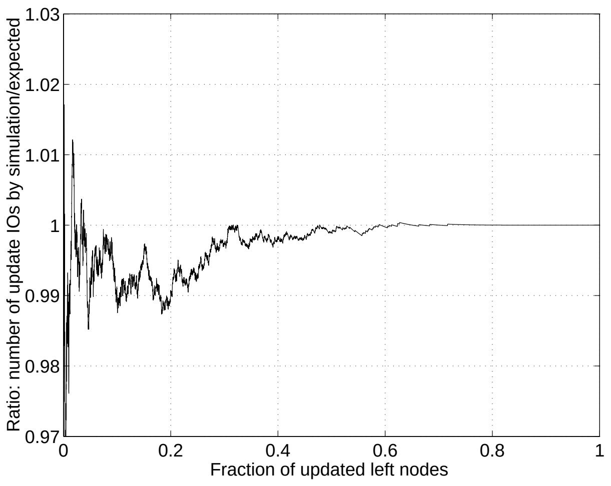

Fig. 6. Ratio $\frac{\text{the number of updated nodes by simulation}}{\text{the number of updated nodes by (16)}}$ as a function of the fraction of modified left nodes for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes.

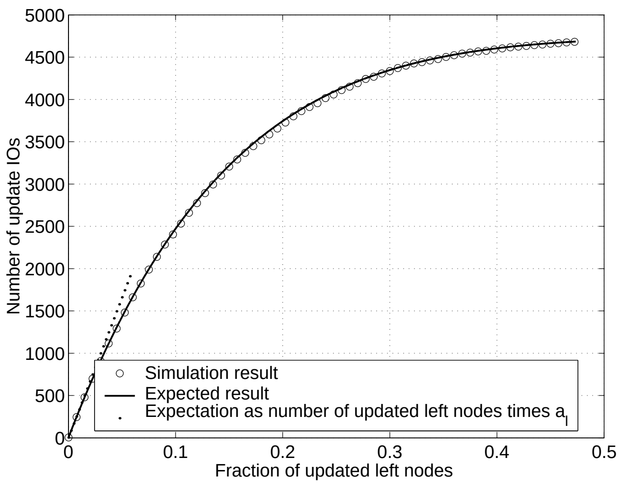

Fig. 5. Number of updated nodes as a function of the fraction of modified left nodes for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes.

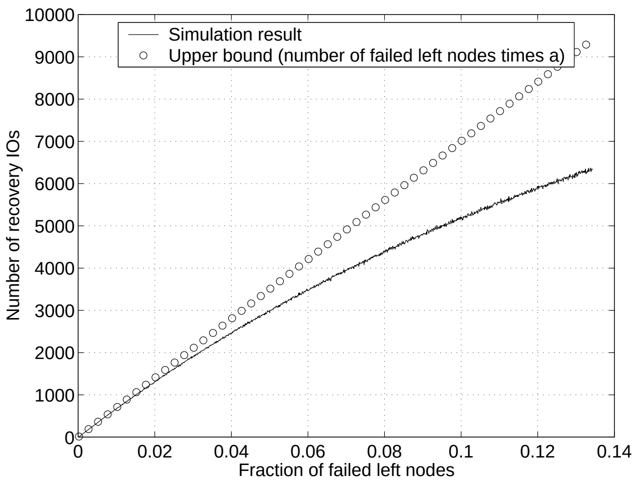

Fig. 7. Number of nodes, accessed by the recovery procedure, as a function of the fraction of failed left nodes, for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes.

not often updated). In this application, the bounded proportionality of accessed bits per erased bits, comes into effect. *E.g.*, consider a three-site reference-data system storing a copy of each datum in two of the three sites. While this system can survive a disaster obliterating a single site, or ongoing failures affecting some devices in all three sites, the MTTDL can be shown to decrease linearly in the number of devices. Using an LDGM code to protect all devices, would result in a system with similar performance, but with MTTDL increasing in the number of devices. Using a classic Reed-Solomon code to protect all devices, would result in a system whose devices would be engaged in much of the time in recovery-related operations.

IV. Applications to Random Sparing

Data which are active (as opposed to reference data), are not best protected by the method in Section IV; The update penalty is too high. Commonly, active data-groups are distributed in some manner over devices, in an attempt to parallelize request handling [18], [19], [20], [21], [22], [23]. We consider a *random-sparing* scheme, apparently similar to an independent OceanStore solution [27], but differing in the random location-decision and use of bounded-penalty codes.

We consider a *random-sparing* scheme, apparently similar to an independent OceanStore solution [27]but differing in the random location-decision and use of bounded-penalty codes. Using queuing theory and the concepts of random-variable negative dependence, we perform approximate analysis on the attributes of this scheme: MTTDL, coding-group average size and rate, required device memory, and reliability-related workload.

This section is organized as follows. In Subsection IV-A we describe the scheme. In subsection IV-B we analyze it. In Subsection IV-C we show simulation results. In Subsection IV-D we discuss some implications of the analysis.

A. Scheme Description

In this subsection we describe the random sparing scheme.

Initially, there are n devices in the system. The data is divided into data groups. For simplicity, we assume the size of each group is ℓ blocks, of which ℓ' can be corrected. We assume the code used has bounded recovery-locality. Specifically, we assume that the ratio of accessed blocks per failed blocks of a group, is at most k. We assume an *average fill-ratio* of β, *i.e.*, of the nc blocks in the system, βnc are taken (and so the number of groups is $\frac{\beta nc}{\ell}$). We place the restriction that a fraction of at most μ of all operations can be dedicated to recovery operations.

We divide the operating time into *rounds* of duration t_s, and *epochs*, each consisting of s rounds. For simplicity, we assume that $st_s = \frac{c}{\mu r}$, *i.e.*, each epoch lasts the time that would be required to sequentially read a device from start to end, normalized by μ.

Naturally, a request to a device can be blocked due to its servicing previous requests. We assume that each device has a queue of read and write requests, which it services subject to its bandwidth and constraint on fraction of recovery-operations. Memory is required for write requests in the queue, and for read requests which have been completed but whose contents are still needed. We later analyze the system-wide amount of such memory.

At some points, the scheme requires writing a (recovered) group element to a one of the operating devices. The selection of the operating device is random, but differs from the standard uniform selection between all operating devices which are not full [27]. Rather, a uniform selection is made between all operating devices (full or non-full), which does not contain a member of the data group. If a full device is selected, a *reassignment* takes place; A random element from the device is chosen to be written someplace else, and the process continues recursively. We explain the rationale for this later.

The scheme is composed of two concurrent processes, a *contracting* process and an *expanding* process, which we describe next.

The contracting process works as follows. In each round, the system observes the set of devices which have failed in the round. For each data group, the system identifies the members that are on failed devices. If these members cannot be recovered, then data has been lost at this time. Otherwise, the system randomly selects, for each failed member, a subset of k devices out of all subsets of k devices which can recover the element. For each of these devices, it inserts into its queue a request to read the required element. If the system has enough elements to recover an element, it recovers it, randomly selects a device, and inserts into its queue a request to write the element (possibly triggering reassignment).

We term this a contracting process, since it attempts to maintain the data groups into a continually contracting group of devices (those which are still operating).

The expanding process works as follows. At the beginning of each epoch, replacement devices are inserted into the system. The number of replacement devices is determined s.t. the expected number of operating devices in the end of the epoch will remain n. In each round, the system chooses, for each replacement device, $\frac{\beta c}{\ell t_s}$ random data groups which are not represented in the device. For each such group, it randomly selects a non-replacement device containing a member of the group, and inserts into its queue a request to read the required element. When the element has been read, the replacement devices writes it.

At the end of an epoch, all replacement devices which have not failed during an epoch, become (new) operating devices. For each element written to a replacement device during an epoch, the system modifies its marked location. It is now marked as being located in the (new) operating device. Its old location is marked as empty.

We term this a contracting process, since it attempts to maintain the data groups into a continually expanding group of devices.

From the above description, it is clear that the number of operating devices (originally n), and the average fill ratio of each device (originally β), change with time. In an arbitrary point in an epoch, we denote these sizes by n' and β', respectively.

The main result is the following theorem.

Theorem 4: Assume

$$\begin{aligned} \ell &\gg \log(n), & (53)\\ c\left(1-\beta\left(1+2\lambda st_s+2(\lambda st_s)^2\right)\right) &\gg 1, & (54)\\ \frac{1}{\lambda} &\gg \frac{c}{\mu r}. & (55)\end{aligned}$$

For some $\delta \gneq 0$, let the storage rate of each coding group be

$$\approx 1-(1+\delta)\lambda\left(t_s+\frac{k+1}{\mu r}\right). \qquad (56)$$

Then the MTTDL of random sparing is

$$\frac{\frac{c}{s\mu r}}{e^{\left(-\frac{1}{2\frac{\beta nc}{\ell}e^{-\lambda\left(t_s+\frac{k+1}{\mu r}\right)\frac{\ell\delta^2}{2}}}\right)}}. \qquad (57)$$

B. Analysis

Following is the proof-outline of Theorem 4. In Lemma 2 we bound the expectation of the number of full devices in a round. In Lemma 3 we bound the probability of a failure in a round, given the effective number of failed devices in the round. In Lemma 4 we approximate the number of pending recovery-requests, using Lemma 2. Using the number of pending requests, we approximate the effective number of failed devices in a round.

The analysis of random sparing is complicated by the fact that the devices' and groups' states are not independent, *e.g.*, if some data-groups have very many representatives in some set of devices, then the number of representatives of other data-groups is probably not very large. This makes it difficult to apply directly the Chernoff bound. For this reason, we use in some places in the analysis, the notion of *negative dependence* [37].

Definition 4: Let $X = \left\{x_1, \ldots, x_{|X|}\right\}$ be an ordered set of random variables. The elements of X are *negatively dependent* if for every disjoint index-sets $I \dot{\bigcup} J \subseteq [|X|]$, and any functions $f : \mathcal{R}^{|I|} \to \mathcal{R}$ and $g : \mathcal{R}^{|J|} \to \mathcal{R}$ that are both non-increasing or non-decreasing,

$$\begin{aligned}\mathbf{E}\left[f\left(x_i, i \in I\right) \cdot g\left(x_j, j \in J\right)\right] \leq \\ \mathbf{E}\left[f\left(x_i, i \in I\right)\right] \cdot \mathbf{E}\left[g\left(x_j, j \in J\right)\right].\end{aligned} \tag{58}$$

The following lemma contains useful properties of negatively-associated random variables which we will use. The statements of the lemma appear in [37], or are slight variations of them.

Lemma 1: Let X be an ordered set of random variables. Then
1. If $f : \mathcal{R} \to \mathcal{R}$ is a non-increasing or non-decreasing function, then the Chernoff bound can be applied to $\sum_{x \in X} f(x)$.
2. If Y is a set of negatively-associated random variables, and X and Y are independent, then $X \dot{\bigcup} Y$ is negatively dependent.

We first bound the expectation of the number of full devices in a round. This in turn, serves to bind the expected number of reassignments performed.

Lemma 2: At some round, let the fraction of operating devices' blocks be β'. Then, with high probability, the fraction of full devices is at most

$$\left(\frac{2}{\beta'\left(\frac{1}{\beta'} - 1\right)^3 c}\right)^{\frac{1}{3}}. \tag{59}$$

Proof: Let the number of devices be n', and define the vector $\underline{c}$, s.t. $\underline{c}[i]$ denotes the number of elements in device i. For some $\delta' \leq 1$ and $\beta'' \leq \beta'$, assume that a subset $S \subseteq [n']$ exists, s.t. $|S| \geq \delta' n'$, and $\forall_{i \in S}\underline{c}[i] \geq \beta'' c$. A straightforward calculation shows that the maximum fraction of full devices, is at most

$$\delta \leq \frac{\frac{\beta' n' c - \delta' n' \beta'' c}{c - \beta'' c}}{n'} = \frac{\beta' - \delta' \beta''}{1 - \beta''}. \tag{60}$$

From (60), to show that δ is small, we can show that $\delta' \overset{\beta'' \approx \beta'}{\approx} 1$.

Consider three processes $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, each inserting $\beta' n' c$ blocks into the n' devices. Each process inserts blocks in $\frac{\beta' n' c}{\ell}$ iterations. In each iteration, process $\mathcal{A}$ inserts a coding group into ℓ distinct devices. If a full device is encountered, reassignment is performed. Process $\mathcal{B}$ does the same for the case $c \to \infty$, ad so reassignment are not performed. Process $\mathcal{C}$ does the same as $\mathcal{B}$, except that at each iteration, the ℓ devices are chosen with replacement.

Define the vector $\underline{w}$ as the indicator of $\underline{c}[i]$ containing less than $(1-\epsilon)\beta' c$ elements, *i.e.*,

$$\underline{w}[i] = I\left(\underline{c}[i] \leq (1-\epsilon)\beta' c\right). \tag{61}$$

We would like to show that at the termination of process $\mathcal{A}$, with high probability, most entries of $\underline{w}$ are 0. It clearly suffices to show that at the termination of process $\mathcal{B}$, with high probability, most entries of $\underline{w}$ are 0.

At the termination of any of the three processes,

$$\forall_{i \in [n']}\mathbf{E}\left[\underline{c}[i]\right] \overset{(a)}{=} \frac{\sum_{i \in [n']} \mathbf{E}\left[\underline{c}[i]\right]}{n'} \overset{(b)}{=} \beta' c, \tag{62}$$

where (a) follows from the symmetry between devices, and (b) follows from linearity of expectation and the fact that $\mathbf{E}\left[\sum_{i \in [n']} \underline{c}[i]\right] = n' \beta' c$. At the termination of process $\mathcal{B}$, it follows from the Chernoff bound that for $i \in [n']$,

$$\mathbf{P}\left(\underline{w}[i] = 1\right) \leq e^{-\frac{\beta' c \epsilon^2}{2}}. \tag{63}$$

We cannot directly deduce from the low probability of the event $\underline{w}[n'] = 1$ in (63), the high-probability of a low-fraction of entries of $\underline{w}$ being 1. The Chernoff bound does not apply immediately , as the entries are not independent. If (63) would result from process $\mathcal{C}$, then condition 2 in Lemma 1 would hold, and the Chernoff bound would apply. For process $\mathcal{B}$, however, condition 2 in Lemma 1 does not hold, as the placements of the ℓ blocks in each iteration are not independent. Rather than using Lemma 1 directly, we show the Chernoff bound applies, by applying the Harris inequality in a slightly different way than used in [37].

Let $c'_n = \underline{c}[n']$. For any t,

$$\begin{aligned}
&\mathbf{E}\left[\Pi_{i \in [n']} e^{t \cdot \underline{w}[i]}\right] = \\
&\mathbf{E}\left[e^{t \cdot \underline{w}[n']} \cdot \Pi_{i \in [n'-1]} e^{t \cdot \underline{w}[i]}\right] = \\
&\mathbf{E}\left[\mathbf{E}\left[e^{t \cdot \underline{w}[n']} \Pi_{i \in [n'-1]} e^{t \cdot \underline{w}[i]} | c'_n\right]\right] \overset{(a)}{=} \\
&\mathbf{E}\left[\mathbf{E}\left[\mathbf{E}\left[e^{t \cdot \underline{w}[n']} | c'_n\right] \Pi_{i \in [n'-1]} e^{t \cdot \underline{w}[i]} | c'_n\right]\right] \overset{(b)}{=} \\
&\mathbf{E}\left[\mathbf{E}\left[e^{t \cdot \underline{w}[n']} | c'_n\right] \mathbf{E}\left[\Pi_{i \in [n'-1]} e^{t \cdot \underline{w}[i]} | c'_n\right]\right] \overset{(c)}{\leq} \\
&\mathbf{E}\left[\mathbf{E}\left[e^{t \cdot \underline{w}[n']} | c'_n\right]\right] \cdot \\
&\mathbf{E}\left[\mathbf{E}\left[\Pi_{i \in [n'-1]} e^{t \cdot \underline{w}[i]} | c'_n\right]\right] = \\
&\mathbf{E}\left[e^{t \cdot \underline{w}[n']}\right] \mathbf{E}\left[\Pi_{i \in [n'-1]} e^{t \cdot \underline{w}[i]}\right] \overset{(d)}{\leq} \\
&\mathbf{E}\left[e^{t \cdot \underline{w}[n']}\right] \Pi_{i \in [n'-1]} \mathbf{E}\left[e^{t \cdot \underline{w}[i]}\right] \leq \\
&\Pi_{i \in [n']} \mathbf{E}\left[e^{t \cdot \underline{w}[i]}\right].
\end{aligned} \tag{64}$$

In the above, (a) and (b) follow from the fact that given c'_n, $e^{t\cdot \underline{w}[n']}$ is obviously a constant. Inequality (c) follows from the Harris inequality [38]. Inequality (d) follows from a repeated application of the same idea to $\mathbf{E}\left[\Pi_{i\in[n'-1]} e^{t\cdot \underline{w}[i]}\right]$.

By (64), the Chernoff bound can be applied to $\sum_{i\in[n']} \underline{w}[i]$. It follows that for large enough n', we can approximate, with high probability $1 - \frac{\sum_{i\in[n']} \underline{w}[i]}{n'}$ by the right side of (63). Inserting into (60), we obtain

$$\frac{\beta' - \left(1 - e^{-\frac{\beta' c\epsilon^2}{2}}\right)\beta'(1-\epsilon)}{1 - \beta'(1-\epsilon)} \overset{(a)}{\approx} \frac{e^{-\frac{\beta' c\epsilon^2}{2}} + \epsilon}{\frac{1}{\beta'} - 1} \overset{(b)}{\leq} \left(\frac{2}{\beta'\left(\frac{1}{\beta'} - 1\right)^3 c}\right)^{\frac{1}{3}}, \tag{65}$$

where (a) follows from neglecting the second order term $\epsilon \cdot e^{-\frac{\beta' c\epsilon^2}{2}}$, and (b) follows from taking $\epsilon = \left(\frac{2}{\beta' c}\right)^{\frac{1}{3}}$. ■

We next bound the error probability in a round, assuming that all groups failing up to the round's start have been recovered.

Lemma 3: At some round, let the number of functioning devices be n', and let the effective number of failed device in the round be n'_f. Assume that for some δ, each group can be recovered if at most

$$\ell' = (1+\delta) n'_F \frac{\ell}{n'} \tag{66}$$

blocks of it are lost.

Then the probability of failure in the round, is

$$e^{\left(-\frac{1}{2\frac{\beta nc}{\ell} e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}}}\right)}. \tag{67}$$

Proof: Let the set of devices failing in the round be F (*i.e.*, $n'_f = |F|$). Define the vector $\underline{z}$ whose ith entry represents the number of elements of the ith group in the failed devices of the round, *i.e.*, for $i \in [n']$,

$$\underline{z}[i] = \left|\left\{ j \mid \exists_{C\in F} \underline{x}^i[j] \in C \right\}\right|. \tag{68}$$

It is easy to see that

$$\mathbf{P}\left(\underline{z}[i] = x\right) = \frac{\binom{\ell}{x}\binom{n'-\ell}{n'_f - x}}{\binom{n'}{n'_f}}. \tag{69}$$

Since the distribution is hyper-geometric, the $\underline{z}[i]$ are negatively dependent. From the Chernoff bound,

$$\mathbf{P}\left(\underline{z}[i] \geq (1+\delta) n'_F \frac{\ell}{n'}\right) \leq e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}}. \tag{70}$$

We define a vector $\underline{w}$ whose ith entry is the indicator of the group-failure event, *i.e.*,

$$\underline{w}[i] = I\left(\underline{z}[i] \geq (1+\delta) n'_F \frac{\ell}{n'}\right). \tag{71}$$

We are interested in the event that $\underline{w}$ is the all-0 vector, *i.e.*, no group had many elements in F. Noting that the elements of $\underline{w}$ are negatively dependent, and that a group failed if

$$\sum_{i\in\frac{\beta nc}{\ell}} \underline{w}[i] \geq 1 = \frac{\beta nc}{\ell} \cdot e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}} \left(1 + \frac{1}{\frac{\beta nc}{\ell} e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}}} - 1\right), \tag{72}$$

we have, by the Chernoff bound,

$$\ln\left(\mathbf{P}\left(\sum_{j\in\left[\frac{\beta nc}{\ell}\right]} w[j] \geq 1\right)\right) \overset{(a)}{\leq} -\frac{\frac{\beta nc}{\ell} e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}} \left(\frac{1}{\frac{\beta nc}{\ell} e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}}} - 1\right)^2}{2} \overset{(b)}{\approx} -\frac{1}{2\frac{\beta nc}{\ell} e^{-\frac{n'_F \frac{\ell}{n'} \cdot \delta^2}{2}}}, \tag{73}$$

where (a) follows from (72), and (b) follows from (53) and the fact that $n'_f = O(n')$. ■

A block is *pending* if it is waiting to be read from or written to some device. The number of pending blocks affects the effective number of failures per round.

Check what to do in next lemma regarding lamdda and

Lemma 4: Let m be the total number of system-wide pending blocks in steady state. Then

$$m \lessapprox n\left(\lambda\beta c t_s + \frac{e^{\frac{\Lambda}{\Xi}} - 1}{2 - e^{\frac{\Lambda}{\Xi}}}\right) \approx n\beta c\lambda\left(t_s + \frac{k+1}{\mu r}\right), \tag{74}$$

where

$$\begin{aligned} \Lambda &\approx \lambda\beta c\,(k+1), \\ \Xi &= \mu r. \end{aligned} \tag{75}$$

Proof:

Add explanation on prev rounds

The inter-failure time of devices from S is distributed. With high probability, the number of failed functioning devices in an epoch approaches $(1 \pm o(1))\,\lambda n s t_s$; the number of functioning devices is always in the approximate range $[n - n\lambda s t_s, n]$. It follows that the failure rate of functioning devices normalized by the remaining number of functioning devices can be upper bounded by

$$(1 \pm o(1))\,\frac{\lambda n s t_s}{(n - n\lambda s t_s)\, s t_s}. \tag{76}$$

The failure of each failed functioning-device's block generates k read requests approximately uniformly distributed among functioning devices. Since the inter-failure time of failed devices is exponential, and a random splitting of an exponential process is itself an exponential process [39], the read-request process per device is approximately distributed exponentially with rate

$$\frac{\lambda n\beta cst_s k}{(n - n\lambda st_s)\, st_s}. \tag{77}$$

The failure of each failed functioning-device's block generates a single write requests and possibly reassignment requests. By the same reasoning, the process generated by these requests per functioning device is approximately distributed exponentially, with rate

$$\frac{\lambda n\beta cst_s k}{(n - n\lambda st_s)\, st_s}\frac{1}{k(1-\gamma)} \overset{(a)}{\approx} \frac{\lambda n\beta cst_s}{(n - n\lambda st_s)\, st_s}, \tag{78}$$

where

$$\beta' = \beta + 2\frac{n_R\beta}{n} = \beta\left(1 + 2\lambda st_s + 2(\lambda st_s)^2\right), \tag{79}$$

$$\gamma = \left(\frac{2}{\beta'\left(\frac{1}{\beta'} - 1\right)^3 c}\right)^{\frac{1}{3}}, \tag{80}$$

and (a) follows from the fact that c is large.

To maintain equilibrium between the contraction and expansion processes, the number of replacement devices should be approximately

$$\left(n\lambda st_s + n\left(\lambda st_s\right)^2\right), \tag{81}$$

where the first term is due to the expected number of failed devices from S, and the second term is due to the expected number of failed replacement devices. By the same reasoning as above, the average rate of requests per functioning device due to the expansion process is

$$\frac{\left(n\lambda st_s + n\left(\lambda st_s\right)^2\right)\beta c}{(n - n\lambda st_s)\, st_s}, \tag{82}$$

which by (55) is negligible in comparison to (77) or (78).

Since the sum process of two exponential processes is exponential [39], the load on each functioning device can be modelled by an M/D/1 queue. The birth rate is given by the sum of (77) and (78), which by (55) is approximately $\lambda\beta c(k+1)$. The death rate is μr.

In Section V we deal with bounds on such queues. Corollary 1 of Subsection V-D gives the steady state distribution of each queue, given the birth and death rates. Using this and the fact that the lengths of the queues are negatively dependent, we obtain (74). ∎

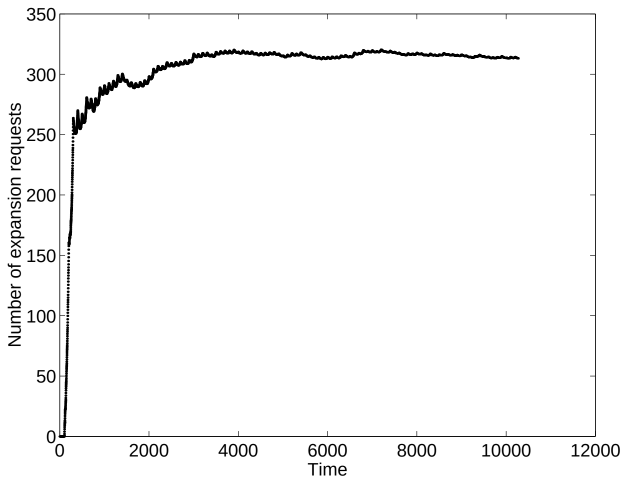

Fig. 8. num bal r reqs as f of time.

C. Simulation Results

In this subsection we show simulation results for the random-sparing scheme:

• Figure 4 shows the single-layer storage-rate and maximal fraction of fixable nodes as a function of left and right edge-degrees (derived in (18) and (13) respectively). As noted, the rate of this code is sub-optimal, necessitating the use of the augmented code from Subsection II-C.

• We have performed 500 tests on the block error-probability. In each test a random code with $m = 10000$ left nodes, $\hat{q} = 4$, $\hat{a} = 4$, $\hat{\alpha} = \frac{1}{3}$, and 8236 right nodes was first built. Then, 10^6 iterations were performed. In each iteration 3900 left nodes were initially set to be corrupted, and were then attempted to be fixed. The code succeeded in every single iteration of every single test.

• Figures 5 and 6 show the number of updated nodes as a function of the fraction of modified left nodes, for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes. Note that as predicted, simulation results coincide with the expected value, given by (16). For small fractions of left nodes being updated, the number of update-IOs can be approximated as the number of updated left nodes times average left degree (a_l).

• Figure 7 shows the number of nodes, accessed by the recovery procedure, as a function of the fraction of failed left nodes, for the case $\hat{q} = 4$, $\hat{a} = 7$, and $m = 10004$ left nodes. Note that the assumption of a accesses per each failed node gives the upper bound on the number of accessed nodes, as shown in (15).

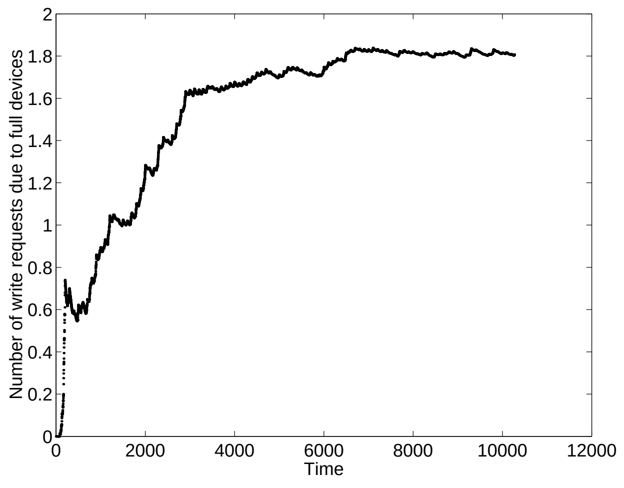

Fig. 9. num cong as f of time.

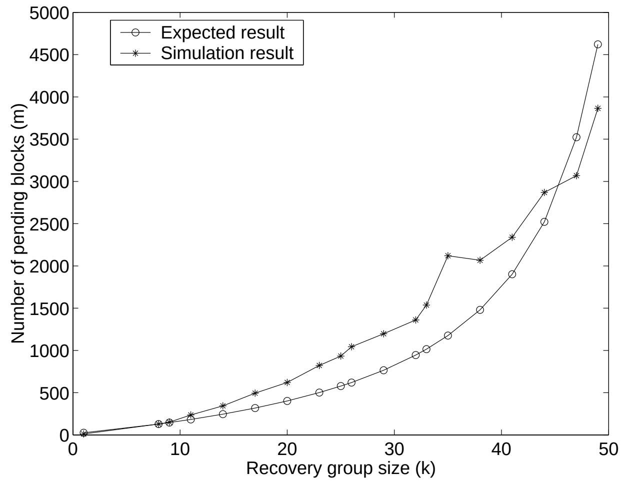

Fig. 12. p blcks f of k.

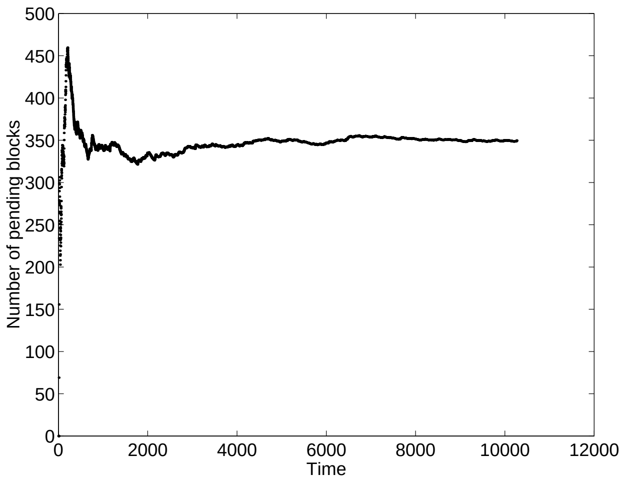

Fig. 10. num int mem as f of time.

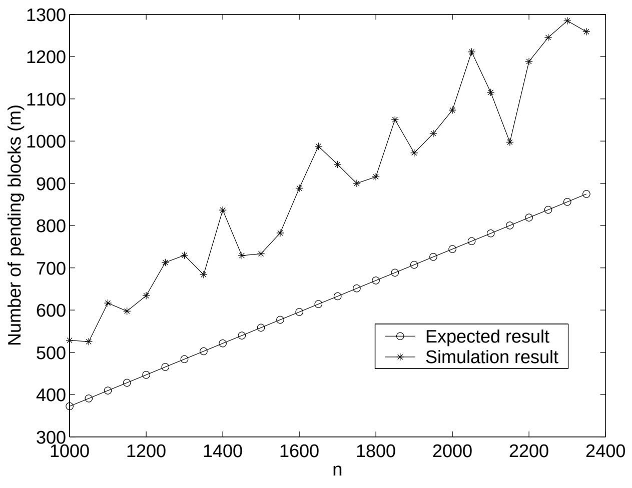

Fig. 13. p blcks f of n.

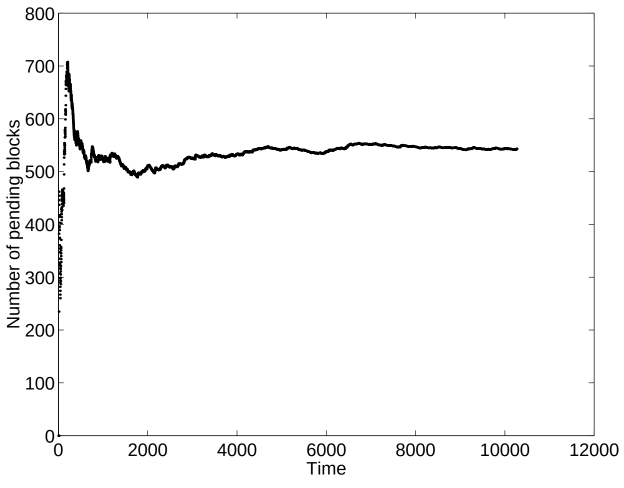

Fig. 11. num pend blks as f of time.

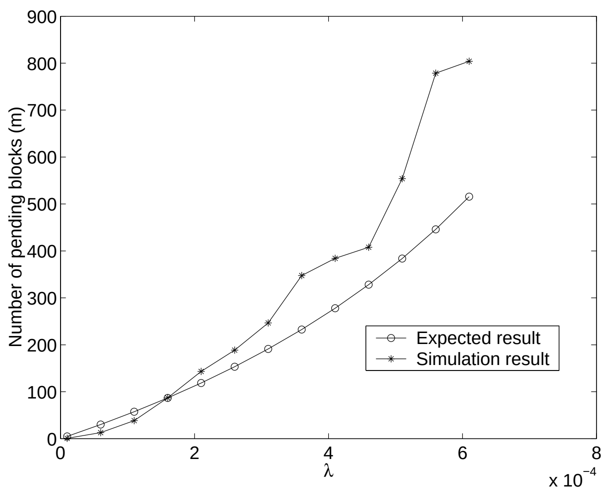

Fig. 14. p blcks f of lambda.

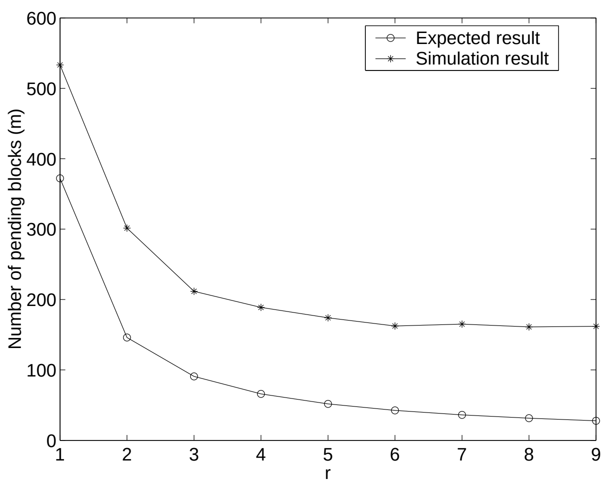

Fig. 15. p blcks f of r.

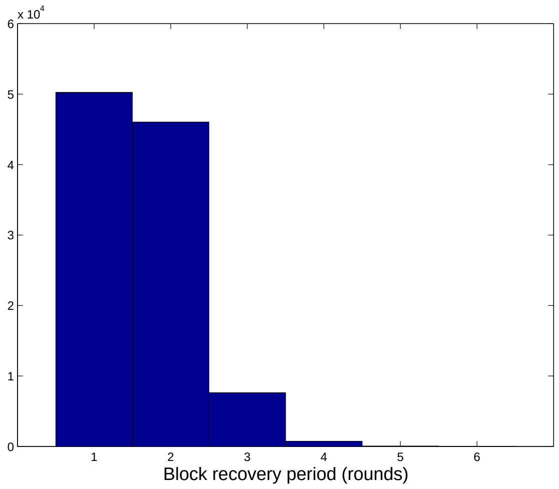

Fig. 18. block in rec time rounds hist.

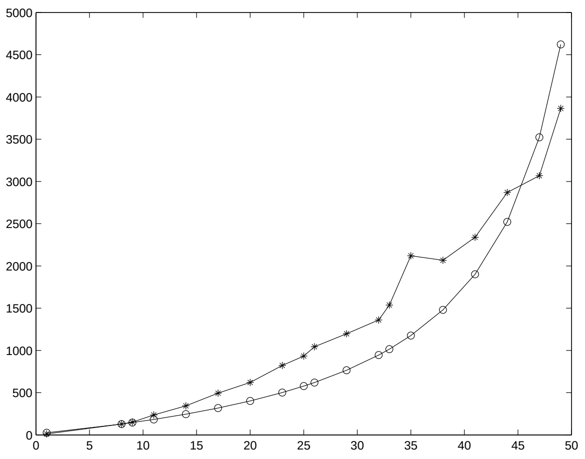

Fig. 16. p blcks f of rec grp sz.

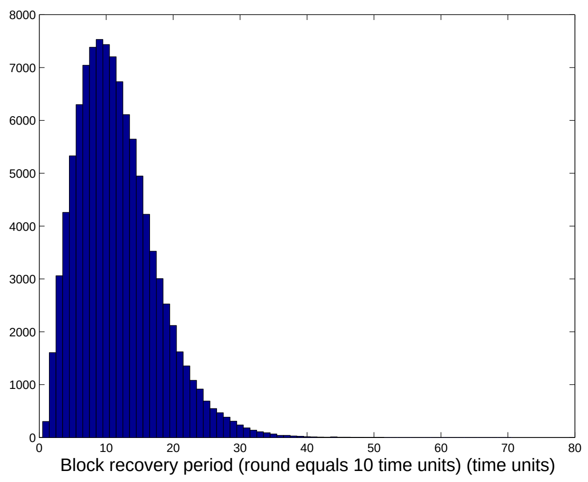

Fig. 19. Recovery histogram ticks.

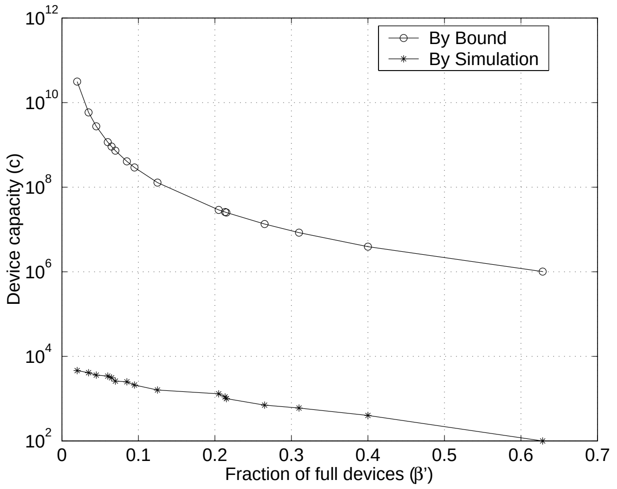

Fig. 17. full devs as f of beta tag.

D. Discussion

In this section we have shown how to create a level employing random sparing. This scheme appears to be similar to one indpendently proposed in OceanStore [27]. From Theorem 4, though, we deduce that the storage rate for small and large coding-groups must differ, as opposed to the uniform coding-rate in OceanStore. Theorem 4 also concurs with the bounds from Section **??**; as the amount of user data grows, the coding-group size must grow as well, and the performance must decrease. The scheme can be used however to build a hierarchy of levels.

The scheme relies on the use of codes with recovery locality. It was previously shown that codes with good update locality have bad recovery locality, and vice versa [31]. While this is true when considering a single code, the conclusion is somewhat different when considering random sparing. The analysis in Lemma 4 shows that if codes do

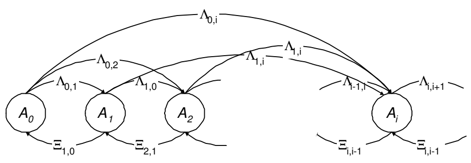

Fig. 20. Discrete-time Markov-graph of failed devices.

not have recovery locality, the load on each device-queue grows indefinitely. It follows that the redundancy of each coding group must grow as well. In this case, the update performance must degrade.

V. High-Intensity Data

For high-intensity data, high performance codes should be used. Assume a group of size n is coded in an MDS code of storage rate $\frac{1}{n}$. One such code is the repetition code, wherein an item is replicated n times. In the setting of storage, the repetition code is usually termed *(multi-way) mirroring.* The repetition code does not require a read-modify-write sequence of operations to all redundant items, when a data item is modified. It is easy to show that it is the only such code (up to trivial variations). This makes the repetition code a good candidate for use in high intensity data level.

A possible semi-Markov modelling of the system is shown in Figure 20. The model consists of $n+1$ states, $A_0, \ldots, A_n$. The state A_i indicates that i devices are failed. The model begins in state A_0; and terminates in state A_n, when data is lost. The various $\Lambda_{i,j}$ and $\Xi_{j,i}$ indicate the transition rates between states, and we define them later on (they do not have the exact interpretation of the corresponding M/M/1 queue values).

At each discrete multiple of t_d, the system detects the set of failed devices. Let A_i and A_j be arbitrary states, where $i \lesssim j$. A transition from A_i to A_j takes place when $j-i+1$ devices fail in time t_d. Note that neglecting the occurrence of two simultaneous events, valid in the M/M/1 queue, is no longer valid. A transition from state A_j to state A_i takes place when $j-i+1$ devices have been recovered. The recovery of a device is performed by reading a copy of the data from operating devices, and writing it to a replacement device (and therefore requires $\frac{c}{r}$ time).

Note that the assumption that the model state completely determines the system state, valid in the M/M/1 queue, is not valid. The transition from A_j to A_i depends on the length of time the system has been in state A_j, as well as the path by which A_j has been reached.

In general, a more complicated setting may be considered, where the number of *copies* is considered, rather than the number of devices. Exploiting recovery-parallelism, the number of copies recovered in a given time may depend on the number of existing copies. We do not analyze this setting in this work.

The use of birth-death processes for finding the MTTDL has been previously used for the case where the number of states is small [1], [17], [22]. When the number of states is large, the ability to recover many devices in parallel, is lost in the model. Also, as noted above, the exponential distribution does not coincide with the deterministic recovery-time, nor does it take into account the detection latency.

The section is organized as follows. In Subsection V-A we model the system in Figure 20 in a way in which is amenable for finding a lower bound on the MTTDL. In Subsection V-B we show a solution to the model in Subsection V-A. In Subsection V-C we show an improved approximation for the lower bound. In Subsection V-D we use the same ideas to find a simple bound on steady-state distributions for the M/D/1 queue, a result needed for Section **??**. In Subsection V-E we show simulation results on the MTTDL of multi-way mirroring.

A. Modelling the System State

The system in Figure 20 shows the model from the view-point of the system controller. In a time interval t_d, many devices can fail. Devices can be in different stages of recovery. This means that each state can be reached from many different states. The resulting system of equations is complex. Rather than solving it directly, we consider a different system and viewpoint.

We consider a system composed of n devices, with failure CDFs $1 - e^{-\hat{\lambda}_1 t}, \ldots, 1 - e^{-\hat{\lambda}_n t}$ $(t \geq 0)$, respectively. We will require in particular two cases: the case $n = 2$ with arbitrary $\hat{\lambda}_1$ and $\hat{\lambda}_2$, and the case of an arbitrary n with $\hat{\lambda}_1 = \cdots = \hat{\lambda}_n = \hat{\lambda}$. We consider the same failure-detection latency and recovery time as in the original system. If a device fails during the recovery of some other devices, the recovery process is aborted and re-initiated. An imaginary observer with zero-delay knowledge of the state of each device observes the system. We consider the view of this observer. The model then becomes that in Figure 21.

In this model, it can be shown that neglecting two simultaneous events is valid. Note that a solution for this model is a lower bound on the solution of a original system in which device failure during recovery does not re-initiate the recovery process.

Let $A(t)$ be the state at time t. Let

$$\begin{aligned} p_{i,j}(h+t,t) &= \mathbf{P}\left(A(h+t) = A_j \mid A(t) = A_i\right), \quad (83)\\ p_i(h+t,t) &= p_{0,i}(h+t,t). \end{aligned}$$

As opposed to an M/M/k type of setting, it is not possible to precisely define here

$$\begin{aligned} p_{i,j}(h) &= p_{i,j}(h+t,t), \quad (84)\\ p_i(h) &= p_{0,i}(h+t,t), \end{aligned}$$

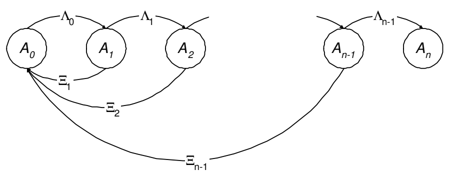

Fig. 21. Markov-graph of failed devices.

since the model state alone does not determine the system state. By equating the transition rates between states, we show that (84) can hold as an approximation, and be found by a system of approximate differential equations.

Definition 5: Let

$$\Xi(\Lambda) = \Lambda\left(\frac{1}{1-e^{-\Lambda\frac{c}{r}}\frac{1-e^{-\Lambda t_d}}{\Lambda t_d}}-1\right). \tag{85}$$

Let $p_i(t) = p_i(t, \hat{\lambda}_1, \ldots, \hat{\lambda}_n, \frac{c}{r}, t_d)$ be defined by the system of differential equations:

$$\begin{aligned} &p_i'(t) = \\ &\quad \Lambda_{i-1}p_{i-1}(t) + \Xi_{i+1}p_{i+1}(t) - (\Lambda_i + \Xi_i)p_i(t), \\ &p_0(0) = 1, \\ &p_1(0) = \cdots = p_n(0) = 0, \end{aligned} \tag{86}$$

for $i \in \{0, \ldots, n\}$, and Λ_i and Ξ_i defined as:

- For the case of system of two devices:

$$\begin{aligned} &\Lambda_0 = \hat{\lambda}_0 + \hat{\lambda}_1, \\ &\Lambda_1 = \frac{\hat{\lambda}_1\hat{\lambda}_2\left(\hat{\lambda}_1+\hat{\lambda}_2+\Xi(\hat{\lambda}_1)+\Xi(\hat{\lambda}_2)\right)}{\hat{\lambda}_1^2+\hat{\lambda}_2^2+\hat{\lambda}_1\Xi(\hat{\lambda}_1)+\hat{\lambda}_2\Xi(\hat{\lambda}_2)}, \\ &\Xi_1 = \\ &\quad \frac{\hat{\lambda}_1^2\Xi(\hat{\lambda}_2)+\hat{\lambda}_2^2\Xi(\hat{\lambda}_1)+\hat{\lambda}_1\Xi(\hat{\lambda}_1)\Xi(\hat{\lambda}_2)+\hat{\lambda}_2\Xi(\hat{\lambda}_1)\Xi(\hat{\lambda}_2)}{\hat{\lambda}_1^2+\hat{\lambda}_2^2+\hat{\lambda}_1\Xi(\hat{\lambda}_1)+\hat{\lambda}_2\Xi(\hat{\lambda}_2)}, \\ &\Lambda_{-1} = \Lambda_2 = \Xi_0 = \Xi_2 = 0. \end{aligned} \tag{87}$$

- For the case of n identical devices:

$$\begin{aligned} \Lambda_i &= \begin{cases} 0 & , \ j \in \{-1, n\} \\ (n-i)\hat{\lambda} & , \ \text{otherwise} \end{cases}, \\ \Xi_i &= \\ &\begin{cases} 0 & , \ i \in \{0, n, n+1\} \\ \Xi(\Lambda_i) & , \ \text{otherwise} \end{cases}. \end{aligned} \tag{88}$$

By equating transition rates, we show the following.

Theorem 5: The probability of data loss at time t is lower bounded by $p_n(t)$, given by Definition 5.

Proof: In the new model, recovery does not commence immediately once devices fail (due to the detection latency t_d). If it is known that an exponential event occurred up to time $t+t_d$, then the a-posteriori probability of its taking place at any time $t' \in [t, t+t_d]$ is equally likely.

Consider two random variables, t_i^b and t_i^d, with respective PDFs:

$$\begin{aligned} \Lambda_i e^{-\Lambda_i t} \quad &, \quad (t > 0), \\ \frac{1}{t_d} \quad &, \quad \left(t \in \left[\frac{c}{r}, \frac{c}{r} + t_d\right]\right). \end{aligned} \tag{89}$$

We define a third process $t_i^\ell = \min\left\{t^{d_i}, t^{b_i}\right\}$. The rate of the renewal process generated by t_i^ℓ, is then $\frac{1}{\mathbf{E}\left[t_i^\ell\right]}$. It can be shown that

$$\mathbf{E}\left[t_i^\ell\right] = \frac{1}{\Lambda_i}\left(1 - e^{-\Lambda_i\frac{c}{r}}\frac{1-e^{-\Lambda_i t_d}}{\Lambda_i t_d}\right). \tag{90}$$

Each renewal in the process generated by t^{l_i} is caused by either t^{b_i} or t^{d_i}. We define the probabilities

$$\begin{aligned} p_i^b &= \mathbf{P}\left(\min\left\{t_i^d, t_i^b\right\} = t_i^b\right), \\ p_i^d &= \mathbf{P}\left(\min\left\{t_i^d, t_i^b\right\} = t_i^d\right). \end{aligned} \tag{91}$$

It can be shown that

$$p_i^b = e^{-\Lambda_i\frac{c}{r}}\frac{1-e^{-\Lambda_i t_d}}{\Lambda_i t_d}, \tag{92}$$

$$p_i^d = 1 - e^{-\Lambda_i\frac{c}{r}}\frac{1-e^{-\Lambda_i t_d}}{\Lambda_i t_d}. \tag{93}$$

By using the standard technique [39] of equating the entry and exit rate of each state using the Chapman-Kolmogorov equations, it can be shown that in this case,

$$p_{i,j}'(t) = \sum_{k \neq j} q_{k,j}p_{i,k}(t) - \nu_j p_{i,j}(t) \tag{94}$$

holds approximately, where ν_j is the rate of the renewal process generated by t_j^ℓ, and $q_{i,k}$ is the rate of the renewal process generated by transferring from A_i to A_k.

It follows that we can insert into (94) the following:

$$\begin{aligned} q_{i,i+1} &= \Lambda_i = p_i^b\frac{1}{\mathbf{E}\left[t_i^\ell\right]}, \\ q_{i,i-1} &= \Xi_i = p_i^d\frac{1}{\mathbf{E}\left[t_i^\ell\right]}, \\ \nu_i &= \Lambda_i + \Xi_i = \frac{1}{\mathbf{E}\left[t_i^\ell\right]}. \end{aligned} \tag{95}$$

It now remains to find the various Λ_i and Ξ_i:

- For the system of 2 devices, the failure rate when in state A_0 is clearly $\Lambda_0 = \hat{\lambda}_1 + \hat{\lambda}_2$. With probability $\frac{\hat{\lambda}_1}{\hat{\lambda}_1+\hat{\lambda}_2}$, the device with $\hat{\lambda}_2$ fails first. In this case, $\Lambda_1 = \hat{\lambda}_2$ and $\Xi_1 = \Xi(\hat{\lambda}_2)$. Similarly, with probability $\frac{\hat{\lambda}_2}{\hat{\lambda}_1+\hat{\lambda}_2}$, $\Lambda_1 = \hat{\lambda}_1$ and

$\Xi_1 = \Xi(\hat\lambda_1)$. By applying the Bayes rule, we get:

$$\mathbf{E}\left[t_1^\ell\right] = \tag{96}$$
$$\frac{\hat\lambda_1}{\hat\lambda_1+\hat\lambda_2}\frac{1}{\hat\lambda_2+\Xi(\hat\lambda_2)} + \frac{\hat\lambda_2}{\hat\lambda_1+\hat\lambda_2}\frac{1}{\hat\lambda_1+\Xi(\hat\lambda_1)} =$$
$$\frac{\hat\lambda_1^2+\hat\lambda_2^2+\hat\lambda_1\Xi(\hat\lambda_1)+\hat\lambda_2\Xi(\hat\lambda_2)}{\left(\hat\lambda_1+\Xi(\hat\lambda_1)\right)\left(\hat\lambda_1+\hat\lambda_2\right)\left(\hat\lambda_2+\Xi(\hat\lambda_2)\right)},$$
$$p_i^b =$$
$$\frac{\hat\lambda_1\hat\lambda_2\left(\hat\lambda_1+\hat\lambda_2+\Xi(\hat\lambda_1)+\Xi(\hat\lambda_2)\right)}{\left(\hat\lambda_1+\Xi(\hat\lambda_1)\right)\left(\hat\lambda_1+\hat\lambda_2\right)\left(\hat\lambda_2+\Xi(\hat\lambda_2)\right)},$$
$$p_i^d =$$
$$\frac{\left(\hat\lambda_1^2\Xi(\hat\lambda_2)+\hat\lambda_2^2\Xi(\hat\lambda_1)+\hat\lambda_1\Xi(\hat\lambda_1)\Xi(\hat\lambda_2)+\hat\lambda_2\Xi(\hat\lambda_1)\Xi(\hat\lambda_2)\right)}{\left(\hat\lambda_1+\Xi(\hat\lambda_1)\right)\left(\hat\lambda_1+\hat\lambda_2\right)\left(\hat\lambda_2+\Xi(\hat\lambda_2)\right)}.$$

Combining (95) and (96), we have

$$\Lambda_1 = \tag{97}$$
$$\frac{\hat\lambda_1\hat\lambda_2\left(\hat\lambda_1+\hat\lambda_2+\Xi(\hat\lambda_1)+\Xi(\hat\lambda_2)\right)}{\hat\lambda_1^2+\hat\lambda_2^2+\hat\lambda_1\Xi(\hat\lambda_1)+\hat\lambda_2\Xi(\hat\lambda_2)},$$
$$\Xi_1 =$$
$$\frac{\hat\lambda_1^2\Xi(\hat\lambda_2)+\hat\lambda_2^2\Xi(\hat\lambda_1)+\hat\lambda_1\Xi(\hat\lambda_1)\Xi(\hat\lambda_2)+\hat\lambda_2\Xi(\hat\lambda_1)\Xi(\hat\lambda_2)}{\hat\lambda_1^2+\hat\lambda_2^2+\hat\lambda_1\Xi(\hat\lambda_1)+\hat\lambda_2\Xi(\hat\lambda_2)}.$$

• For the system of n identical devices, when in state A_i, the time to device failure is clearly distributed exponentially with rate $\Lambda_i = (n-i)\hat\lambda$. Ξ_i can be found as before. ■

B. Estimating the MTTDL

In this subsection we modify the method in [40] to solve (86) for the MTTDL.

Theorem 6: Assume that

$$\frac{1}{\frac{c}{r}+\frac{t_d}{2}} \gg n\lambda. \tag{98}$$

Then the MTTDL of the system in Figure 21 is approximately lower bounded by

$$\sum_{j=1}^{n}\frac{1}{\Lambda_{j-1}}\prod_{i=j}^{n-1}\left(1+\frac{\Xi_i}{\Lambda_i}\right) \approx \frac{\left(\frac{1}{\frac{c}{r}+\frac{t_d}{2}}\right)^{n-1}}{n!\lambda^n}, \tag{99}$$

where $\Xi_i = \Xi(\Lambda_i)$.

Proof: For the system in Figure 21, the equation system (86) becomes the following:

$$\begin{aligned} p_0'(t) &= -\Lambda_0 p_0(t) + \sum_{i=1}^{n-1}\Xi_i p_i(t), \qquad (100)\\ p_j'(t) &= -\left(\Xi_j+\Lambda_j\right)p_j(t) + \Lambda_{j-1}p_{j-1}(t),\\ p_n'(t) &= \Lambda_{n-1}p_{n-1}(t),\\ p_1(0) &= 1, \end{aligned}$$

where $j \in [n-1]$.

Using the Laplace transform:

$$\begin{aligned} \int_0^\infty p_j(t)e^{-st}\,dt &= a_j(s), \qquad (101)\\ \int_0^\infty p_j'(t)e^{-st}\,dt &= -p_j(0) + s a_j(s), \end{aligned}$$

we obtain the equation system:

$$\begin{aligned} (-s-\Lambda_0)a_0(s) + \sum_{i=1}^{k-1}\Xi_i a_i(s) = -p_0(0) &= -1, \qquad (102)\\ (-s-\Xi_j-\Lambda_j)a_j(s) + \Lambda_{j-1}a_{j-1}(s) = -p_j(0) &= 0,\\ -s a_n(s) + \Lambda_{n-1} &= 0. \end{aligned}$$

Note that once the system enters A_n, it will not leave it. It follows that defining the MTTDL by $\bar t_n$, we have

$$\bar t_n = \int_0^\infty R_S(t)\,dt = \int_0^\infty t p_n'(t)\,dt. \tag{103}$$

By properties of the reverse Laplace-transform, this is

$$\bar t_n = -\left.\frac{d\,(s a_n(s))}{d\,s}\right|_{s=0}. \tag{104}$$

We therefore need to solve the equation system (102) for $a_n(s)$. To do so, we apply Cramer's rule, obtaining

$$a_n(s) = \frac{g_n(s)}{g(s)}, \tag{105}$$

where

$$g(s) = \tag{106}$$
$$\Delta\begin{pmatrix} -R_0^O & \Xi_1 & \Xi_2 & \cdots & \Xi_{n-1} & \\ \Lambda_0 & -R_1^O & & & & \\ & \ddots & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \Lambda_{n-2} & -R_{n-1}^O & \\ & & & & \Lambda_{n-1} & -s \end{pmatrix},$$

$$g_n(s) = \tag{107}$$
$$\Delta\begin{pmatrix} -R_0^O & \Xi_1 & \Xi_2 & \cdots & \Xi_{n-1} & -1 \\ \Lambda_0 & -R_1^O & & & & \\ & \ddots & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \Lambda_{n-2} & -R_{n-1}^O & \\ & & & & \Lambda_{n-1} & \end{pmatrix},$$

and

$$R_i^O = \Lambda_i + \Xi_i + s. \tag{108}$$

We first develop $g(s)$ and $g_n(s)$. For the former, we have

$$g(s) = -s\gamma_n(s), \tag{109}$$

where

$$\gamma_n(s) = \tag{110}$$
$$\Delta\begin{pmatrix} -R_0^O & \Xi_1 & \Xi_2 & \dots & \Xi_{n-1} \\ \Lambda_0 & -R_1^O & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & \Lambda_{n-2} & -R_{n-1}^O \end{pmatrix}.$$

For the latter, we have

$$g_n(s) = (-1)^{n+1}\prod_{i=0}^{n-1}\Lambda_i. \tag{111}$$

Inserting (109) and (111) into (105) and (104), we have

$$\bar{t}_n = \frac{g_n'(0) + \gamma_n'(0)}{\gamma_n(0)}. \tag{112}$$

In the matrix within the determinant in (106), we may add all rows to the last without altering $g(s)$. To the matrix within the determinant in (107), we may append an all-0 column and the row $-s, -s, \dots, -1$, without altering $g_n(s)$. After doing so, we have

$$g_n(s) + \gamma_n(s) = \tag{113}$$
$$\Delta\begin{pmatrix} -R_0^O & \Xi_1 & \Xi_2 & \dots & \Xi_{n-1} & -1 \\ \Lambda_0 & -R_1^O & & & & \\ & \ddots & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \Lambda_{n-2} & -R_{n-1}^O & \\ -s & -s & -s & -s & -s & -1 \end{pmatrix} -$$
$$\Delta\begin{pmatrix} -R_0^O & \Xi_1 & \Xi_2 & \dots & \Xi_{n-1} & 0 \\ \Lambda_0 & -R_1^O & & & & \\ & \ddots & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \Lambda_{n-2} & -R_{n-1}^O & \\ -s & -s & -s & -s & -s & -1 \end{pmatrix} \overset{(a)}{=}$$
$$\Delta\begin{pmatrix} -R_0^O & \Xi_1 & \Xi_2 & \dots & \Xi_{n-1} & -1 \\ \Lambda_0 & -R_1^O & & & & \\ & \ddots & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \Lambda_{n-2} & -R_{n-1}^O & \\ -s & -s & -s & -s & -s & 0 \end{pmatrix} \tag{114}$$

where (a) follows from the fact that the determinant is distributive over matrices identical in all rows but one.

We note that the determinant in (114) is

$$s(-1)^n B_n(s), \tag{115}$$

where

$$B_n(s) = \tag{116}$$
$$\Delta\begin{pmatrix} \Lambda_0 & -R_1^O & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & \Lambda_{n-2} & -R_{n-1}^O \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Furthermore, the following recurrence system is satisfied at $s = 0$:

$$\begin{aligned} B_k(0) &= \prod_{i=0}^{k-2}\Lambda_i + (\Lambda_{k-1} + \Xi_{k-1})\, B_{k-1}(0), \\ B_1(0) &= 1. \end{aligned} \tag{117}$$

Solving (117), we obtain

$$B_k(0) = \sum_{j=1}^{k}\prod_{i=0}^{j-2}\Lambda_i \prod_{i=j+1}^{k}(\Lambda_{i-1} + \Xi_{i-1}). \tag{118}$$

Setting $s = 0$ in (110), we have

$$\gamma_n(0) = (-1)^{n+1}\prod_{i=0}^{n-1}\Lambda_i. \tag{119}$$

Inserting (118) and (119) into (112), we get

$$\begin{aligned} \bar{t}_n &= \\ & \frac{1}{\gamma_n(0)}\left.\frac{d\,(g_n(s) + \gamma_n(s))}{d\,s}\right|_{s=0} = \\ & (-1)^k\frac{B_n(0)}{\gamma_n(0)} = \\ & \sum_{j=1}^{n}\frac{1}{\Lambda_{j-1}}\prod_{i=j}^{n-1}\left(1 + \frac{\Xi_i}{\Lambda_i}\right). \end{aligned} \tag{120}$$

By applying (98) to (85), we have

$$\min_i\left\{\frac{\Xi_i}{\Lambda_i}\right\} \gg 1, \tag{121}$$

and (120) simplifies to

$$\bar{t}_n \approx \frac{\prod_{i=1}^{n-1}\Xi_i}{\prod_{i=0}^{n-1}\Lambda_i}. \tag{122}$$

This can be further simplified by substituting

$$\begin{aligned} \Lambda_i &= (n-i)\lambda, \\ \Xi_i &\overset{(a)}{\approx} \left(\frac{1}{\frac{c}{r} + \frac{t_d}{2}}\right), \end{aligned} \tag{123}$$

where (a) follows from the application of (98) to (85). ■

C. Improving the MTTDL Estimation

The system model from Subsection V-B assumes that any device failure while recovery, restarts the recovery process. This is needed for the definition of the state-transition rates. This is clearly a drawback. The MTTDL found in Section V-B therefore only lower bounds the true MTTDL. In this subsection we attempt to rectify this by considering "compound devices", *i.e.*, devices composed of a sub-group of devices.

The main point we prove in this subsection is the following.

Theorem 7: Assume that

$$\frac{1}{\frac{c}{r}+\frac{t_d}{2}} \gg 2\lambda. \tag{124}$$

Then the MTTDL of the system in Figure 21 is approximately

$$\frac{\left(\frac{1}{\frac{c}{r}+\frac{t_d}{2}}\right)^{n-1}}{2^{n-1}\lambda^n}. \tag{125}$$

We first analyze the failure-time distribution for the general case of two devices.

Lemma 5: A system of 2 devices whose failure times have respective CDFs $1-e^{-\hat{\lambda}_1 t}$ and $1-e^{-\hat{\lambda}_2 t}$, has a failure time having an approximate CDF

$$1-e^{-\frac{2\hat{\lambda}_1\hat{\lambda}_2}{\Xi}t} \tag{126}$$

where

$$\Xi = \frac{1}{\frac{c}{r}+\frac{t_d}{2}}, \tag{127}$$

assuming that

$$\frac{1}{\frac{c}{r}+\frac{t_d}{2}} \gg \max\left\{\hat{\lambda}_1, \hat{\lambda}_2\right\}. \tag{128}$$

Proof: For this case, (86) becomes

$$\begin{aligned} p_0'(t) &= \Xi_1 p_1(t) - \Lambda_0 p_0(t), \\ p_1'(t) &= \Lambda_0 p_0(t) - (\Xi_1+\Lambda_1)p_1(t), \\ p_2'(t) &= \Lambda_1 p_1(t), \end{aligned} \tag{129}$$

where

$$\begin{aligned} &\Lambda_0 = \hat{\lambda}_0 + \hat{\lambda}_1, \\ &\Lambda_1 = \frac{\hat{\lambda}_1\hat{\lambda}_2\left(\hat{\lambda}_1+\hat{\lambda}_2+\Xi(\hat{\lambda}_1)+\Xi(\hat{\lambda}_2)\right)}{\hat{\lambda}_1^2+\hat{\lambda}_2^2+\hat{\lambda}_1\Xi(\hat{\lambda}_1)+\hat{\lambda}_2\Xi(\hat{\lambda}_2)}, \\ &\Xi_1 = \\ &\frac{\hat{\lambda}_1^2\Xi(\hat{\lambda}_2)+\hat{\lambda}_2^2\Xi(\hat{\lambda}_1)+\hat{\lambda}_1\Xi(\hat{\lambda}_1)\Xi(\hat{\lambda}_2)+\hat{\lambda}_2\Xi(\hat{\lambda}_1)\Xi(\hat{\lambda}_2)}{\hat{\lambda}_1^2+\hat{\lambda}_2^2+\hat{\lambda}_1\Xi(\hat{\lambda}_1)+\hat{\lambda}_2\Xi(\hat{\lambda}_2)}. \end{aligned} \tag{130}$$

Solving (129) for $p_0(t)$ and $p_1(t)$, we get

$$\begin{aligned} p_0(t) &= -\frac{r_2+\Lambda_0}{r_1-r_2}e^{r_1 t} + \frac{r_1+\Lambda_0}{r_1-r_2}e^{r_2 t}, \\ p_1(t) &= \frac{(\Lambda_0+r_1)(\Lambda_0+r_2)}{\Xi_1(r_1-r_2)}\left(-e^{r_1 t}+e^{r_2 t}\right), \end{aligned} \tag{131}$$

where

$$\begin{aligned} &r_{1,2} = \\ &\frac{-\Lambda_0-\Lambda_1-\Xi_1 \pm \sqrt{(\Lambda_0+\Lambda_1+\Xi_1)^2-4\Lambda_0\Lambda_1}}{2} = \\ &\frac{\Lambda_0+\Lambda_1+\Xi_1}{2}\left(-1 \pm \sqrt{1-\frac{4\Lambda_0\Lambda_1}{(\Lambda_0+\Lambda_1+\Xi_1)^2}}\right). \end{aligned} \tag{132}$$

Using the approximation in (128), we derive the following approximations for $i=1,2$: $\hat{\lambda}_i t_d \ll 1$ and $\hat{\lambda}_i \frac{c}{r} \ll 1$. Applying these approximations to (85), and using the power-series approximation of e^x for small x, we get $\Xi(\hat{\lambda}_2) \approx \Xi(\hat{\lambda}_2) \approx \Xi = \frac{1}{\frac{c}{r}+\frac{t_d}{2}}$. Applying this to (130), we get

$$\Xi_1 \approx \Xi = \frac{1}{\frac{c}{r}+\frac{t_d}{2}}. \tag{133}$$

Applying (133) and using $\Xi \gg \max\{\hat{\lambda}_1, \hat{\lambda}_2\}$ in (130), we get

$$\Lambda_1 \approx \frac{2\hat{\lambda}_1\hat{\lambda}_2}{(\hat{\lambda}_1+\hat{\lambda}_2)} \overset{(a)}{\leq} 2\max\left\{\hat{\lambda}_1, \hat{\lambda}_2\right\}, \tag{134}$$

where (a) follows from the Arithmetic - Geometric Mean inequality. Combining (134) with (133), we get $\frac{\Xi_1}{\max\{\Lambda_1,\Lambda_2\}} \gg 1$. Using this fact, we approximate

$$\begin{aligned} r_1 &\approx -(\Lambda_0+\Lambda_1+\Xi_1) \\ r_2 &\approx -\frac{\Lambda_0\Lambda_1}{\Lambda_0+\Lambda_1+\Xi_1} \approx -\frac{\Lambda_0\Lambda_1}{\Xi_1}. \end{aligned} \tag{135}$$

The lemma follows by the fact that

$$|r_1| \gg |r_2| \Rightarrow e^{r_1 t} \ll e^{r_2 t}. \tag{136}$$

■

We now prove Theorem 7.

Proof: Lemma 5 gives the approximate failure-CDF of a system composed of two devices. From (126) we see that this distribution is exponential with inverse mean

$$\begin{aligned} \tilde{\lambda} &= \frac{2\hat{\lambda}_1\hat{\lambda}_2}{\Xi} \approx \\ &2\hat{\lambda}_1\hat{\lambda}_2\left(\frac{c}{r}+\frac{t_d}{2}\right). \end{aligned} \tag{137}$$

Given a system of $n \gneq 2$ devices, we can recursively consider the system as composed of two devices, one simple, and one "compound", both with failure times distributed exponentially. The corresponding state diagram is shown in Figure 22. In general, we can decompose a system of n devices recursively, s.t. each device represents a compound device, an example of which is shown in Figure 23.

When analyzing the reliability of such a system, it is convenient to view it as a *full binary* tree (*i.e.*, a tree where each node has 0 or 2 children), as in diagrams (a) and (b) of

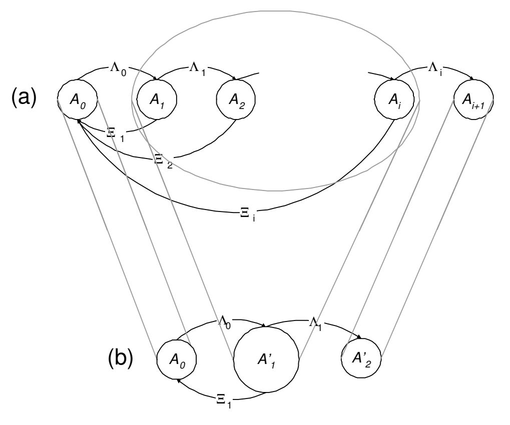

Fig. 22. Reduction to a compound system.

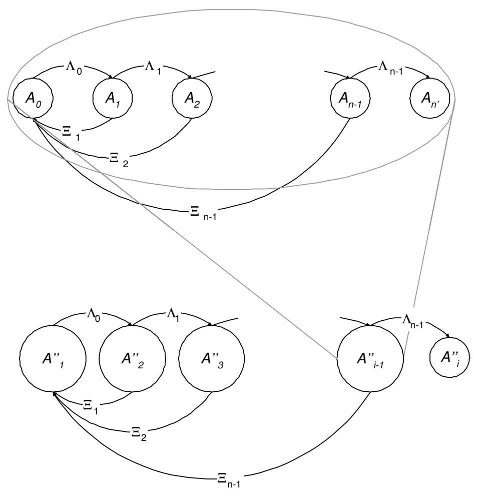

Fig. 23. A recursive analysis.

Figure 24. In the tree of each diagram, any leaf is a device node (indicated by D). Any inner node is a compound node (indicated by C), the reliability of which is determined by its children nodes. The system reliability is that of the root. Lemma 5 shows how to combine the MTTDL of two subtrees. Surprisingly, the MTTDL of the entire tree depends only on the number of leaves, and not on the tree topology chosen.

■

D. Lower Bounds on M/D/1 Steady-State Distributions

In this subsection we find a simple bound on steady-state distributions for the M/D/1 queue. This is needed for Section IV.

For the case of the M/D/1 queue, equating the transition rates yields the following simple lower-bound approximation. In Figure 25 we see a graph corresponding to the

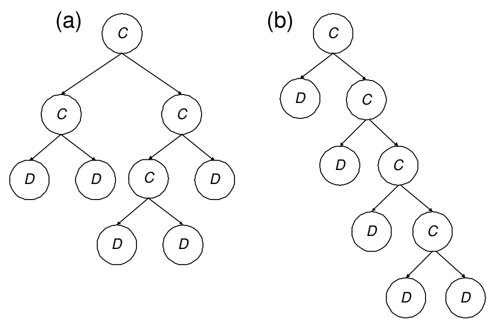

Fig. 24. A tree formed by the recursive analysis

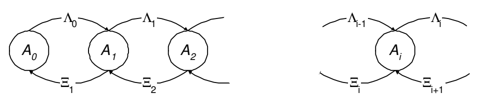

Fig. 25. Markov-graph of birth-death process of failed devices.

M/D/1 process. Specifically, assume the process is a birth-death process, with births generated by a Poisson process with mean interval times $\frac{1}{\lambda}$, and deaths occurring deterministically after $\frac{1}{\xi}$ time.

Corollary 1: For the given M/D/1 queue, let P_i be the steady-state distribution of state A_i. Then

$$P_i = \left(e^{\frac{\lambda}{\xi}} - 1\right)^i \left(2 - e^{\frac{\lambda}{\xi}}\right). \tag{138}$$

Proof: Following [39], we equate the transition rates, resulting, in a manner similar to the one used in Subsection V-A, in

$$P_i = \begin{cases} \frac{\Xi P_1}{\lambda} & , \quad i = 0 \\ \frac{\Xi P_{i+1} + \lambda P_{i-1}}{\Xi + \lambda} & , \quad i \neq 0 \end{cases}, \tag{139}$$

where

$$\Xi = \lambda \left(\frac{1}{1 - e^{-\frac{\lambda}{\xi}}} - 1 \right) \tag{140}$$

by 85.

The proof follows by solving (139), subject to the constraint $\sum_{i=0}^{\infty} P_i = 1$.

■

E. Simulation Results

In this subsection we show simulation results for multi-way mirroring:

• Figures 26, 27, and 28 show the behavior of a system of two devices (Subsection V-C) for different settings of $\hat{\lambda}_1$, $\hat{\lambda}_2$, and Ξ. Each of the figures compares the CDFs obtained

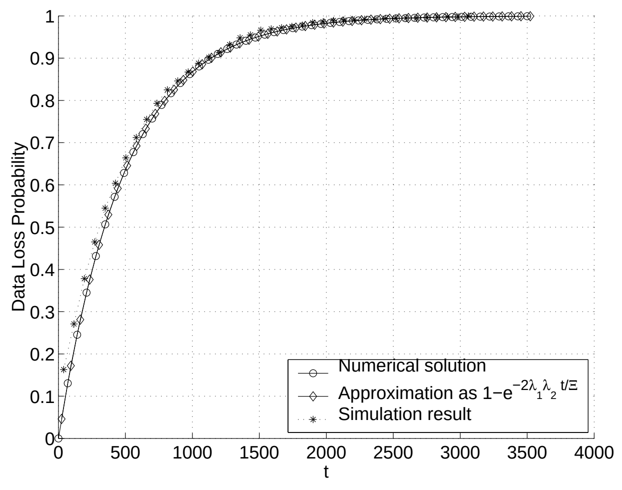

Fig. 26. The CDF of the two-devices system MTTDL, for the case $\hat{\lambda}_1 = \hat{\lambda}_2 = \frac{1}{30}$ and $\Xi = 1$.

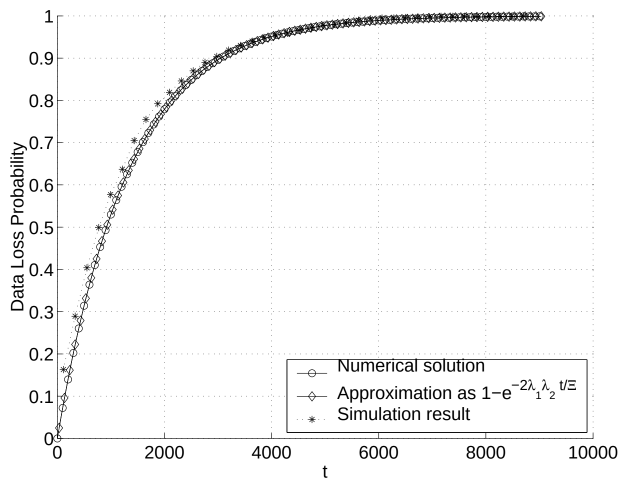

Fig. 27. The CDF of the two-devices system MTTDL, for the case $\hat{\lambda}_1 = \hat{\lambda}_2 = \frac{1}{50}$ and $\Xi = 1$.

by the following: the numerical solution of a differential-equation model of the system, the approximated solution shown in Lemma 5, and the results obtained by a software simulation of the system of devices. Note that the simulation results coincide with the approximation of Lemma 5 for both $\hat{\lambda}_1 = \hat{\lambda}_2$ and $\hat{\lambda}_1 \gg \hat{\lambda}_2$ settings. This result justifies the assumption that the MTTDL of a "compound device" has exponential distribution.

• Figure 29 and 30 show the behavior of a system as a function of the number of devices (n). Figure 29 compares the MTTDLs obtained by the following: the results obtained by a software simulation of the system of devices, the exact and approximated expression from Theorem 6, and the approximation of Theorem 7. Note that Theorem 7, based on the "compound device" approach, achieves the best estimation. Figure 30 compares, for two different settings, the MTTDLs obtained by the following: the results obtained by a software simulation of the system of devices, and the approximation of Theorem 7. Note that Theorem 7 indeed gives a lower bound on the system MTTDL.

VI. Conclusions and Future Work

In this work we have presented and analyzed codes for storage systems. We have shown that it is easy to construct LDGM codes which have bounded recovery penalty. We have shown algorithms which incorporate codes of this type. Finding bounds and optimal codes subject to bounded penalty constraints, is left to future research. We have analyzed the distribution of the time to data loss of multi-way mirroring. The fact that such radically different codes and schemes can be applied to storage systems, suggests that storage reliability should be hierarchical [41]. This in turn raises many questions on the number of levels the hierarchy should contain, the code appropriate to each level, and the transition policy between levels. We leave these interesting questions to future research.

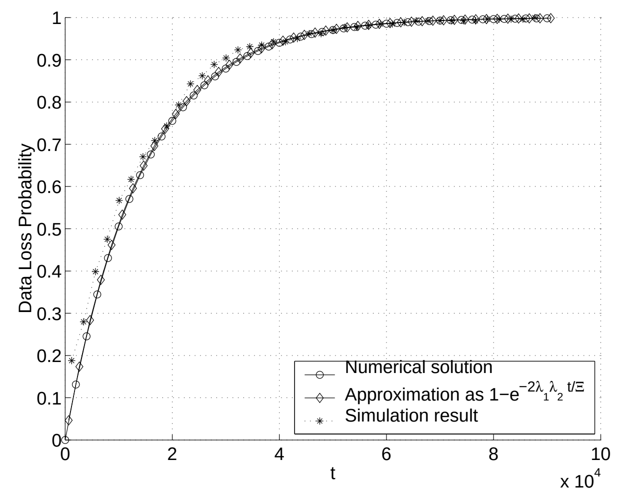

Fig. 28. The CDF of the two-devices system MTTDL, for the case $\hat{\lambda}_1 = \frac{1}{30} \gg \hat{\lambda}_2 = \frac{1}{900}$ and $\Xi = 1$.

VII. Acknowledgements

Thanks to Dana Ron and David Burshtein of the Dept. of EE-Systems at Tel Aviv University, and Alain Azagury, Michael Factor, Kalman Meth, Julian Satran, and Dafna Sheinwald of IBM, for useful discussions.

References

[1] G. A. Gibson, *Redundant Disk Arrays: Reliable Parallel Secondary Storage*, ser. ACM Distinguished Dissertations. Cambridge, MA: MIT Press, 1992.

[2] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, ser. Wiley Series in Telecommunications. New York, NY, USA: John Wiley & Sons, 1991.

[3] P. Elias, "Coding for two noisy channels," in *Information Theory Third London Symposium*. London: Buterworth's Scientific Publications, Sept. 1955, pp. 61–76.

[4] C. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, July 1948.

[5] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan,

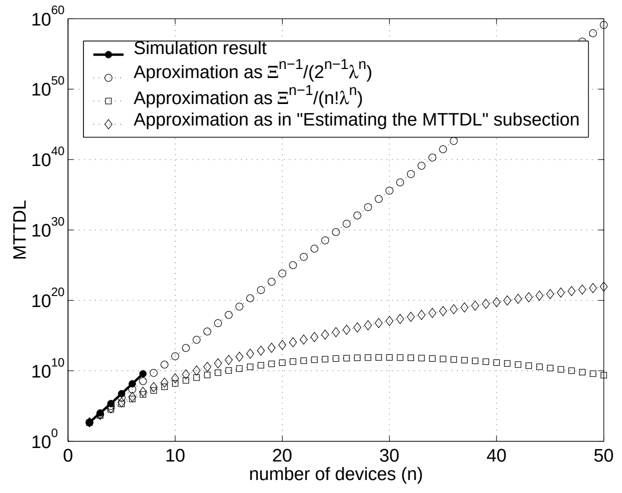

Fig. 29. The system MTTDL as a function of the number of devices (n), for the case $\hat{\lambda}_1 = \ldots = \hat{\lambda}_n = \frac{1}{30}$ and $\Xi = 1$.

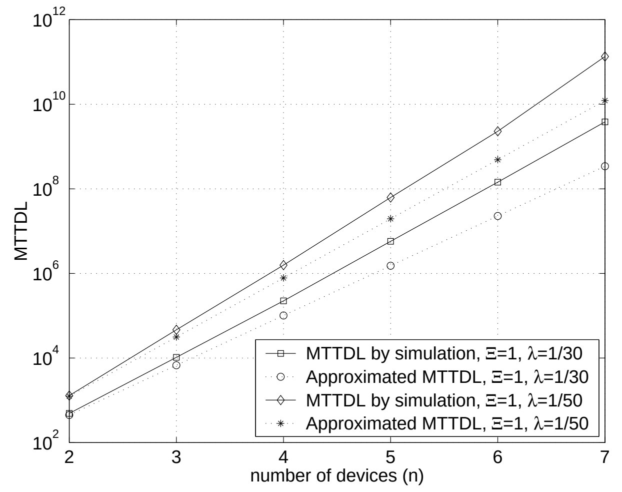

Fig. 30. The system MTTDL as a function of the number of devices (n).

"Priority encoding transmission," in *Proceedings: 35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico.* 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA: IEEE Computer Society Press, Nov. 1994, pp. 604–612.

[6] S. Boucheron and K. Salamatian, "About priority encoding transmission," *IEEE Trans. Inform. Theory*, vol. 46, pp. 697–705, Mar. 2000.

[7] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, ser. North-Holland Mathematical Library. North-Holland, 1977, vol. 16.

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

[9] N. Alon and M. Luby, "A linear time erasure-resilient code with nearly optimal recovery," *IEEETIT: IEEE Transactions on Information Theory*, vol. 42, 1996.

[10] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding:turbo-codes," in *Proceedings of IEEE ICC'93*.

[11] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing (STOC '97).* New York: Association for Computing Machinery, May 1997, pp. 150–159.

[12] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEETIT: IEEE Transactions on Information Theory*, vol. 47, 2001.

[13] A. Shokrollahi, "New sequences of linear time erasure codes approaching the channel capacity," pp. 65–76, 1999.

[14] J. A. Katzman, "System architecture for nonstop computing," in *14th IEEE Cpmputer Society International Conference (COMPCON)*, Feb. 1977, pp. 77–80.

[15] D. A. Patterson, G. Gibson, and R. H. Katz, "A case for redundant arrays of inexpensive disks (raid)," in *Proceedings of the ACM Conference on Management of Data (SIGMOD)*, June 1988, pp. 109–116.

[16] Q. M. Malluhi and W. E. Johnston, "Coding for high availability of a distributed-parallel storage system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 12, pp. 1237–1252, Dec. 1998.

[17] Q. Xin, E. Miller, D. Long, S. Brandt, T. Schwarz, and W. Litwin, "Reliability mechanisms for very large storage systems," in *In Proceedings of the 20th IEEE / 11th NASA Goddard Conference on Mass Storage Systems and Technologies. IEEE*, Apr. 2003.

[18] M. Holland, G. A. Gibson, and D. P. Siewiorek, "Architectures and algorithms for on-line failure recovery in redundant disk arrays," *Journal of Distributed and Parallel Databases*, vol. 2, no. 3, pp. 295–335, July 1994.

[19] J. Menon and D. Mattson, "Distributed sparing in disk arrays," in *Proceedings of the COMPCOM Conference*, Feb. 1992, pp. 410–421.

[20] J. M. Menon and R. L. Mattson, "Comparison of sparing alternatives for disk arrays," in *Proceedings the 19th Annual International Symposium on Computer Architecture,ACM SIGARCH*, Gold Coast, Australia, May 1992, pp. 318–329.

[21] A. Thomasian and J. Menon, "RAID5 performance with distributed sparing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, no. 6, pp. 640–657, June 1997.

[22] X. Wu, J. Li, and H. Kameda, "Reliable analysis of disk array organizations by considering uncorrectable bit errors," in *Proceedings of The 16th Symposium on Reliable Distributed Systems (SRDS '97).* Washington - Brussels - Tokyo: IEEE, Oct. 1997, pp. 2–9.

[23] J. Chandy and A. L. N. Reddy, "Failure evaluation of disk array organizations," in *Proceedings of the 13th International Conference on Distributed Computing Systems*, R. Werner, Ed. Pittsburgh, PA: IEEE Computer Society Press, May 1993, pp. 319–327.

[24] R. G. S. Kirkpatrick, W. Wilcke and H. Huels, "Percolation in dense storage arrays," in *Messina Symposium,*.

[25] F. Chang, M. Ji, S.-T. Leung, J. MacCormick, S. Perl, and L. Zhang, "Myriad: Cost-effective disaster tolerance," in *Proceedings of the FAST '02 Conference on File and Storage Technologies (FAST-02).* Berkeley, CA: USENIX Association, Jan. 28–30 2002, pp. 103–116.

[26] J. Wilkes, R. Golding, C. Staelin, and T. Sullivan, "The HP AutoRAID hierarchical storage system," in *High Performance Mass Storage and Parallel I/O: Technologies and Applications*, H. Jin, T. Cortes, and R. Buyya, Eds. New York, NY: IEEE Computer Society Press and Wiley, 2001, pp. 90–106.

[27] J. Kubiatowicz, D. Bindel, P. Eaton, Y. Chen, D. Geels, R. Gummadi, S. Rhea, W. Weimer, C. Wells, H. Weatherspoon, and B. Zhao, "OceanStore: An architecture for global-scale persistent storage," *ACM SIGPLAN Notices*, vol. 35, no. 11, pp. 190–201, Nov. 2000.

[28] H. Weatherspoon, M. Delco, and S. Zhuang, "Typhoon: An archival system for tolerating high degrees of file server failure," 1999, available through http://www.cs.berkeley.edu/~hweather/Typhoon/TyphoonReport.html.

[29] G. A. Gibson, L. Hellerstein, R. M. Karp, R. H. Katz, and D. A. Patterson, "Failure correction techniques for large disk arrays," *Third Int'l Conf. on Architectural Support for Programming Languages and Operating Systems*, p. 123, Apr. 1989.

[30] L. Hellerstein, G. A. Gibson, R. M. Karp, R. H. Katz, and D. A. Patterson, "Coding techniques for handling failures in large disk arrays," *Algorithmica*, vol. 12, no. 2/3, pp. 182–208, Aug./Sept. 1994.

[31] Y. Chee, Colbourn, and Ling, "Asymptotically optimal erasure-

resilient codes for large disk arrays," *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, vol. 102, 2000.

[32] D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, B. Zhao, and J. Kubiatowicz, "Oceanstore: An extremely wide-area storage system," in *Proceedings of the Nine International Symposium on Architectural Support for Programming Languages and Operating Systems (ASPLOS IX)*, Nov. 2000.

[33] M. Luby, "LT codes," in *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2002.

[34] A. Shokrollahi, "An introduction to low-density parity-check codes," *Lecture Notes in Computer Science*, vol. 2292, pp. 175–197, 2002.

[35] J. A. Cooley, J. L. Mineweaser, L. D. Servi, and E. T. Tsung, "Software-based erasure codes for scalable distributed storage," in *20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSS'03)*. IEEE, April 2003, p. 157.

[36] M. Sipser and D. Spielman, "Expander codes," *IEEE Transactions on Information Theory (special issue devoted to coding theory)*, pp. 1710–1722, 1996.

[37] D. Dubashi and D. Ranjan, "Balls and bins: A study in negative dependence," *BRICS*, 1996.

[38] T. E. Harris, "A lower bound for the critical probability in a certain percolation process," in *Proc. Cam. Phil. Soc.*, vol. 56, 1960, pp. 13–20.

[39] S. M. Ross, *Introduction to Probability Models*, 8th ed. Harcourt Publishers Ltd, Dec. 2002.

[40] B. A. Kozlov and I. A. Ushakov, *Reliability Handbook*. New York: Holt, Rinehart and Winston Inc., 1970.

[41] J. Wilkes, R. Golding, C. Staelin, and T. Sullivan, "The HP AutoRAID hierarchical storage system," *ACM Transactions on Computer Systems*, vol. 14, no. 1, pp. 108–136, Feb. 1996.