Contemporary Engineering Sciences, Vol. 9, 2016, no. 21, 1015 - 1029
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ces.2016.6584

Comparative Study of LRC and RS Codes

Vladimir Vinnikov

OOO Acronis, Altufievskoe sh, 44, 127566, Moscow, Russia
Computing Centre RAS, Vavilov st. 40, 119333, Moscow, Russia

Lyudmila Ivanichkina

OOO Proekt IKS, Altufievskoe sh, 44, 127566, Moscow, Russia
MIPT, 9 Institutskiy per., 141700, Dolgoprudny, Moscow Region, Russia

Andrew Neporada

OOO Proekt IKS, Altufievskoe sh, 44, 127566, Moscow, Russia

Abstract

This paper is concerned with the properties of local reconstruction codes (LRC) both hierarchical and nonuniform, providing the reliability of data storages. The LRCs belong to erasure codes, which recover initial data from corruption at the cost of storage overhead due to introduced redundancy. The study is carried out via Markov chain solutions to reveal advantages and drawbacks of the LRCs in comparison with classical Reed–Solomon (RS) codes. Thus, we investigate dependencies of storage reliability on mean time of disk replacement for the codes under consideration.

Keywords: Mean time to data loss (MTTDL), erasure codes, Reed–Solomon codes, local reconstruction codes, storage reliability, Azure storage

1 Introduction

The methods and algorithms of data preservation are among the most important problems of computer science. Nowadays, the requirements to operate

Big Data demand the modern data storage systems to contain thousands and tens of thousands of hard disk drives. According to statistical laws, every large ensemble is prone to regular failure of its elements. The greater is the size of deployed storage devices, the higher is the rate of singular and collective failures. To compensate for disk failures and avoid the irrecoverable data losses, the stored data is made redundant. Often the redundancy relies upon erasure codes, like widely implemented Reed–Solomon (n, k)-codes. In this case the data reliability is provided by placing n encoded data fragments to different disks of the storage.

Among all erasure codes the Reed–Solomon (n, k)-code provide maximum level of error tolerance $(n - k)$ with minimum disk space overhead. However, the up-to-date storage systems have constraints on practical implementations of these codes. These restricting factors come from distributed hardware architecture, where network channels with limited bandwidth and nonzero latency of data transfer interconnect storage disks as well as computation costs of arithmetic operations in the finite Galois fields. Thus, the content recovery of a failed disk requires to transfer data fragments from k disks and, therefore, perform k network operations. To reduce maintenance traffic, various software manufacturers develop erasure codes that reconstruct initial data on the lesser number of fragments and network operations by the cost of additional disk space overhead. The commonly used trade-off alternative to Reed–Solomon codes is a class of locally repairable codes (LRC). Generally, each variant of LRC falls into either of two categories, namely nonuniform LRC and hierarchical LRC. The prominent example of the nonuniform LRC is implemented in Microsoft Azure storage [4]. This erasure code differs from classical one by calculating extra parity fragments for subgroups of a a data block. Each of local parities is intended to recover singular erasure using only data fragments of the respective subgroup. The local redundancy also can be organized in a hierarchical form (see e.g. [10]), where each data fragment of the block at $\mathcal{L}$-th tier acts as a block at the next $(\mathcal{L} + 1)$-th tier. These nested codes recursively process each resulting block, thus increasing the error tolerance without driving the required processing power to unreasonable levels.

2 Nonuniform LRC

For the sake of simplicity, we consider the following example of nonuform LRC (nLRC) algorithm. First, assume that the initial data block consists of six data fragments in two triads. Next, apply erasure code to obtain two global parity fragments for the block and one local parity fragment for each triad. Thus, the encoded data comprise of ten fragments in grouped by five. We denote that code as $(10, 2, 2)$-nLRC. One can estimate reliability of this LRC via Markov chain, modeling states of stored block and transitions from one state to another. We describe the block state $S(x)$ as

$$S(x) = \{x_{D1}|x_{D2}|x_{LP1}|x_{LP2}|x_{GP}|\},$$

where x_{Di} is the number of failed disks with data fragments in an i-th group, x_{LPi} is the number of failed disks with local parity fragments in an i-th group, x_{GP} is the number of failed disks with global parity fragments, and $x = x_{D1} + x_{D2} + x_{LP1} + x_{LP1} + x_{GP}$ is total number of failed disks within an encoded data block.

Contrary to Reed–Solomon codes the number of failed disks in LRC is insufficient to define states of irrecoverable data loss $S_{DL}(x)$. The occurrence of irrecoverable data loss also depends on erasure distribution across fragments of different types. For example, the four-disk failures permit both operational and data loss states:

$$\begin{aligned} S(4) = {} & \{3|1|0|0|0\} \cup \{3|0|0|1|0\} \cup \{2|2|0|0|0\} \cup \{2|1|1|0|0\} \cup \{2|1|0|1|0\} \\ & \cup \{2|1|0|0|1\} \cup \{2|0|1|1|0\} \cup \{2|0|0|1|1\} \cup \{1|3|0|0|0\} \cup \{1|2|1|0|0\} \\ & \cup \{1|2|0|1|0\} \cup \{1|2|0|0|1\} \cup \{1|1|1|1|0\} \cup \{1|1|1|0|1\} \cup \{1|1|0|1|1\} \\ & \cup \{1|0|1|1|1\} \cup \{1|0|0|1|2\} \cup \{0|3|1|0|0\} \cup \{0|2|1|1|0\} \cup \{0|1|1|1|1\} \\ & \cup \{0|1|1|0|2\} \cup \{0|0|1|1|2\}, \end{aligned}$$

$$\begin{aligned} S_{DL}(4) = {} & \{3|0|1|0|0\} \cup \{3|0|0|0|1\} \cup \{2|0|0|0|2\} \cup \{1|0|1|0|2\} \cup \{0|3|0|1|0\} \\ & \cup \{0|3|0|0|1\} \cup \{0|2|0|1|1\} \cup \{0|2|0|0|2\} \cup \{0|1|0|1|2\}. \end{aligned}$$

Therefore, the reliability to store the LRC-encoded data block is described via Markov chain model and is determined by the transition rates λ between aggregated states (see Table 1). The absorbing states, which describe data loss are $S_{DL}(4)$ and $S_{DL}(5)$. According to the theorem of state aggregation, these absorbing states can be joined into singular state S_{DL}. Without restricting the generality, we assume that the recovery of all failed disks in each state except initial and absorbing ones proceeds in parallel with the rate μ, i.e. all failures detected in a regular state are repaired simultaneously with transition to initial state $S(0)$. Thus, the Markov chain has graphical form represented on the Fig.1. One can see on that figure that the local parity fragment gives the ability to recover the corresponding triad from a single erasure. In case of two and three erasures, triad is restored using both local and global parities. However, if the global parity fragment fails then its recovery requires all data fragments.

To estimate reliability of an erasure code we use the mean time to data loss (MTTDL) parameter. The value of this parameter indicates the average operational time for a stored block to reach irrecoverable state from initial

Table 1: Aggregated Markov states for a $(10, 2, 2)$ nonuniform LRC

$S\,(1)$		$S\,(2)$		$S\,(3)$		$S\,(4)$	
$S_{1,1}$	$\{1\vert 0\vert 0\vert 0\vert 0\}$	$S_{2,1}$	$\{1\vert 1\vert 0\vert 0\vert 0\}$	$S_{3,1}$	$\{2\vert 1\vert 0\vert 0\vert 0\}$	$S_{4,1}$	$\{3\vert 1\vert 0\vert 0\vert 0\}$
	$\{0\vert 0\vert 0\vert 1\vert 0\}$		$\{1\vert 0\vert 0\vert 1\vert 0\}$		$\{2\vert 0\vert 0\vert 1\vert 0\}$		$\{3\vert 0\vert 0\vert 1\vert 0\}$
$S_{1,2}$	$\{0\vert 1\vert 0\vert 0\vert 0\}$		$\{0\vert 0\vert 1\vert 1\vert 0\}$		$\{1\vert 2\vert 0\vert 0\vert 0\}$		$\{2\vert 2\vert 0\vert 0\vert 0\}$
$S_{1,3}$	$\{0\vert 0\vert 1\vert 0\vert 0\}$	$S_{2,2}$	$\{2\vert 0\vert 0\vert 0\vert 0\}$		$\{1\vert 1\vert 1\vert 0\vert 0\}$		$\{2\vert 1\vert 1\vert 0\vert 0\}$
$S_{1,4}$	$\{0\vert 0\vert 0\vert 0\vert 1\}$		$\{1\vert 0\vert 1\vert 0\vert 0\}$		$\{1\vert 1\vert 0\vert 1\vert 0\}$		$\{2\vert 1\vert 0\vert 1\vert 0\}$
			$\{1\vert 0\vert 0\vert 0\vert 1\}$		$\{1\vert 1\vert 0\vert 0\vert 1\}$		$\{2\vert 1\vert 0\vert 0\vert 1\}$
			$\{0\vert 2\vert 0\vert 0\vert 0\}$		$\{1\vert 0\vert 1\vert 1\vert 0\}$		$\{2\vert 0\vert 1\vert 1\vert 0\}$
			$\{0\vert 1\vert 0\vert 1\vert 0\}$		$\{1\vert 0\vert 0\vert 1\vert 1\}$		$\{2\vert 0\vert 0\vert 1\vert 1\}$
			$\{0\vert 1\vert 0\vert 0\vert 1\}$		$\{0\vert 2\vert 1\vert 0\vert 0\}$		$\{1\vert 3\vert 0\vert 0\vert 0\}$
			$\{0\vert 0\vert 1\vert 0\vert 1\}$		$\{0\vert 1\vert 1\vert 1\vert 0\}$		$\{1\vert 2\vert 1\vert 0\vert 0\}$
			$\{0\vert 0\vert 0\vert 1\vert 1\}$		$\{0\vert 1\vert 1\vert 0\vert 1\}$		$\{1\vert 2\vert 0\vert 1\vert 0\}$
			$\{0\vert 0\vert 0\vert 0\vert 2\}$		$\{0\vert 0\vert 1\vert 1\vert 1\}$		$\{1\vert 2\vert 0\vert 0\vert 1\}$
				$S_{3,2}$	$\{3\vert 0\vert 0\vert 0\vert 0\}$		$\{1\vert 1\vert 1\vert 1\vert 0\}$
					$\{2\vert 0\vert 1\vert 0\vert 0\}$		$\{1\vert 1\vert 1\vert 0\vert 1\}$
					$\{2\vert 0\vert 0\vert 0\vert 1\}$		$\{1\vert 1\vert 0\vert 1\vert 1\}$
					$\{1\vert 0\vert 1\vert 0\vert 1\}$		$\{1\vert 0\vert 1\vert 1\vert 1\}$
					$\{1\vert 0\vert 0\vert 0\vert 2\}$		$\{1\vert 0\vert 0\vert 1\vert 2\}$
					$\{0\vert 3\vert 0\vert 0\vert 0\}$		$\{0\vert 3\vert 1\vert 0\vert 0\}$
					$\{0\vert 2\vert 0\vert 1\vert 0\}$		$\{0\vert 2\vert 1\vert 1\vert 0\}$
					$\{0\vert 2\vert 0\vert 0\vert 1\}$		$\{0\vert 1\vert 1\vert 1\vert 1\}$
					$\{0\vert 1\vert 0\vert 1\vert 1\}$		$\{0\vert 1\vert 1\vert 0\vert 2\}$
					$\{0\vert 1\vert 0\vert 0\vert 2\}$		$\{0\vert 0\vert 1\vert 1\vert 2\}$
					$\{0\vert 0\vert 1\vert 0\vert 2\}$	$S_{4,2}$	$\{3\vert 0\vert 1\vert 0\vert 0\}$
					$\{0\vert 0\vert 0\vert 1\vert 2\}$		$\{3\vert 0\vert 0\vert 0\vert 1\}$
							$\{2\vert 0\vert 0\vert 0\vert 2\}$
							$\{1\vert 0\vert 1\vert 0\vert 2\}$
							$\{0\vert 3\vert 0\vert 1\vert 0\}$
							$\{0\vert 3\vert 0\vert 0\vert 1\}$
							$\{0\vert 2\vert 0\vert 1\vert 1\}$
							$\{0\vert 2\vert 0\vert 0\vert 2\}$
							$\{0\vert 1\vert 0\vert 1\vert 2\}$

errorless state. The expression for a block MTTDL (see [1]) is

$$MTTDL_{MC} = MTFF_{MC} = \sum_{i=1}^{\mathcal{N}} T_i,$$

where $\mathcal{N}$ is a number of resulting Markov states, $(T_i)_{i=1}^{\mathcal{N}}$ is a vector solution for a linear system

$$-\left(T_1, T_2, \ldots, T_{\mathcal{N}}\right) \times \mathcal{E} = (1\ 0\ \ldots\ 0)\,.$$

where $\mathcal{E}$ is a square submatrix of transition rate matrix, holding only rows with transitions from nonabsorbing states (1).

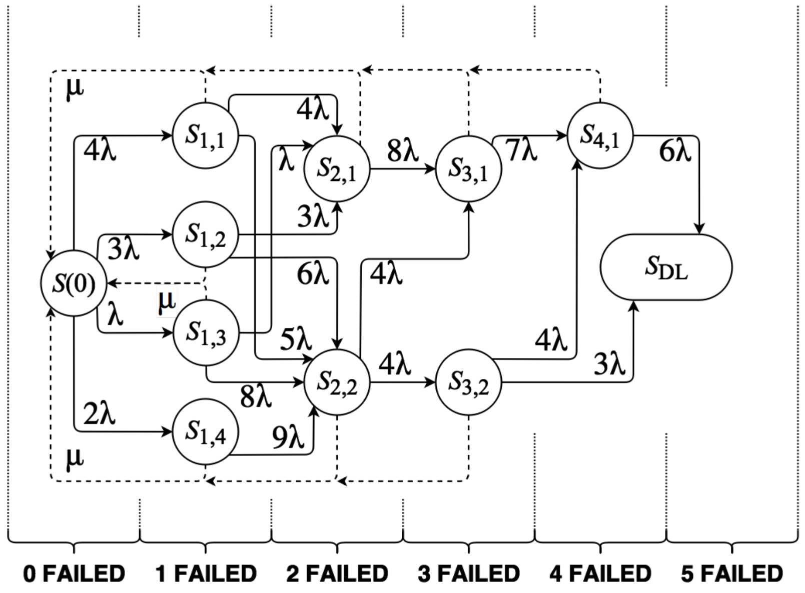

Figure 1: Markov chain with aggregated states and transition rates between them for nonuniform LRC

One can see on the Fig.1 that the encoded data block can unconditionally endure three disk failures and a subset $S_{4,1}$ of four-disk failures. In accordance to above described Markov chain, the submatrix $\mathcal{E}$ of transition rates has the form

$$\mathcal{E} = \begin{pmatrix}
-10\lambda & 4\lambda & 3\lambda & \lambda & 2\lambda & 0 & 0 & 0 & 0 & 0 \\
\mu & -9\lambda-\mu & 0 & 0 & 0 & 4\lambda & 5\lambda & 0 & 0 & 0 \\
\mu & 0 & -9\lambda-\mu & 0 & 0 & 3\lambda & 6\lambda & 0 & 0 & 0 \\
\mu & 0 & 0 & -9\lambda-\mu & 0 & \lambda & 8\lambda & 0 & 0 & 0 \\
\mu & 0 & 0 & 0 & -9\lambda-\mu & 0 & 9\lambda & 0 & 0 & 0 \\
\mu & 0 & 0 & 0 & 0 & -8\lambda-\mu & 0 & 8\lambda & 0 & 0 \\
\mu & 0 & 0 & 0 & 0 & 0 & -8\lambda-\mu & 4\lambda & 4\lambda & 0 \\
\mu & 0 & 0 & 0 & 0 & 0 & 0 & -7\lambda-\mu & 0 & 7\lambda \\
\mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -7\lambda-\mu & 4\lambda \\
\mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6\lambda-\mu
\end{pmatrix} \qquad (1)$$

3 Hierarchical LRC

An idea of local reconstruction codes permits an alternative variant to the nonuniform LRC, which achieves a trade-off between storage and traffic overheads by means of hierarchical encoding. The hierarchical LRC (hLRC) are constructed as follows. Let $\mathcal{L}$ be the depth of hierarchical LRC scheme. So, the k_0 fragments of the initial data block are encoded with Reed–Solomon (n_0, k_0)-scheme. Next, each of the n_0 resulting fragments is split into k_1 second-tier fragments, which are further encoded with (n_1, k_1)-scheme. After that, the algorithm is applied recursively down to specified nesting level $\mathcal{L}$ with erasure parameters $n_{\mathcal{L}-1}, m_{\mathcal{L}-1}$. Thus, the hierarchical LRC code is modeled by the Markov chain with following data block states $S(x)$ as

$$S(x) = \{x_{D1} | x_{D2} | x_P |\},$$

where x_{Di} is the number of failed disks containing an i-th $(\mathcal{L})$-tier data fragment, x_P is the number of failed disks with $(\mathcal{L})$-tier parity fragment, and $x = x_{D1} + x_{D2} + x_P$ is a total number of failed disks within an encoded data block.

For example, we consider the two-tier LRC code with erasure parameters $n_i = 3$, $k_i = 2$, $\mathcal{L} = 2$, $i \in \{0; \mathcal{L} - 1\}$ with initial and resulting block sizes of four and nine, respectively

$$\left(\prod_{i=0}^{\mathcal{L}-1} n_i, \prod_{i=0}^{\mathcal{L}-1} k_i \right).$$

Thus, the corresponding Markov states are as follows (see Table 2).

Table 2: Aggregated Markov states for a $(3, 2)$ hierarchical LRC

$S(1)$		$S(2)$		$S(3)$		$S(4)$		$S(5)$	
$S_{1,1}$	$\{1\|0\|0\}$	$S_{2,1}$	$\{1\|1\|0\}$	$S_{3,1}$	$\{1\|1\|1\}$	$S_{4,2}$	$\{2\|1\|1\}$	$S_{5,3}$	$\{3\|1\|1\}$
	$\{0\|1\|0\}$		$\{1\|0\|1\}$	$S_{3,2}$	$\{2\|1\|0\}$		$\{1\|2\|1\}$		$\{1\|3\|1\}$
	$\{0\|0\|1\}$		$\{0\|1\|1\}$		$\{2\|0\|1\}$		$\{1\|1\|2\}$		$\{1\|1\|3\}$
		$S_{2,2}$	$\{2\|0\|0\}$		$\{1\|2\|0\}$	$S_{4,3}$	$\{3\|1\|0\}$		
			$\{0\|2\|0\}$		$\{1\|0\|2\}$		$\{3\|0\|1\}$		
			$\{0\|0\|2\}$		$\{0\|2\|1\}$		$\{1\|3\|0\}$		
					$\{0\|1\|2\}$		$\{1\|0\|3\}$		
				$S_{3,3}$	$\{3\|0\|0\}$		$\{0\|3\|1\}$		
					$\{0\|3\|0\}$		$\{0\|1\|3\}$		
					$\{0\|0\|3\}$				

The Markov chain on the Fig.2 shows that the hierarchically encoded data block can always withstand three disk failures and subsets $S_{4,2} \cup S_{4,3}$, $S_{5,3}$ of four- and five-disk failures respectively. The corresponding submatrix $\mathcal{E}$ of transition rates has the form (2)

$$\mathcal{E} = \begin{pmatrix} -9\lambda & 9\lambda & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mu & -8\lambda-\mu & 6\lambda & 2\lambda & 0 & 0 & 0 & 0 & 0 & 0 \\ \mu & 0 & -7\lambda-\mu & 0 & 3\lambda & 4\lambda & 0 & 0 & 0 & 0 \\ \mu & 0 & 0 & -7\lambda-\mu & 0 & 6\lambda & \lambda & 0 & 0 & 0 \\ \mu & 0 & 0 & 0 & -6\lambda-\mu & 0 & 0 & 6\lambda & 0 & 0 \\ \mu & 0 & 0 & 0 & 0 & -6\lambda-\mu & 0 & 3\lambda & \lambda & 0 \\ \mu & 0 & 0 & 0 & 0 & 0 & -6\lambda-\mu & 0 & 6\lambda & 0 \\ \mu & 0 & 0 & 0 & 0 & 0 & 0 & -5\lambda-\mu & 0 & \lambda \\ \mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5\lambda-\mu & 3\lambda \\ \mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4\lambda-\mu \end{pmatrix} \tag{2}$$

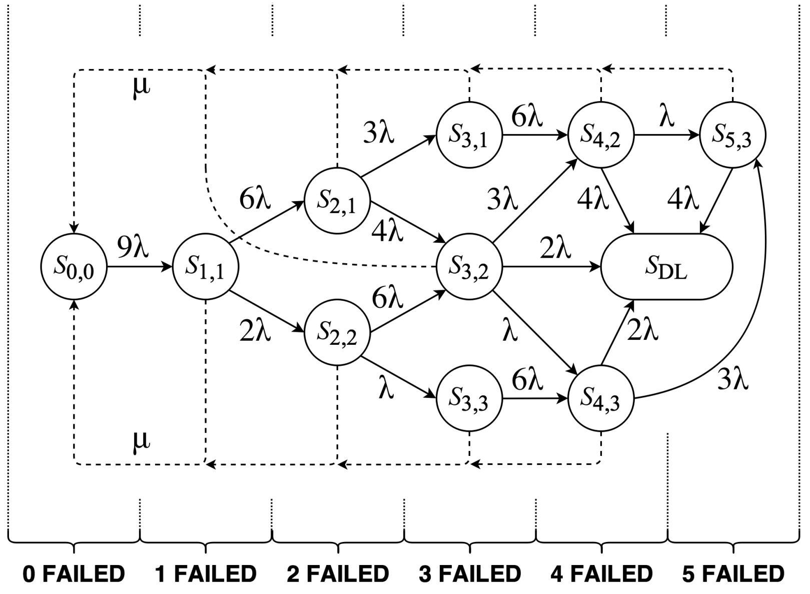

Figure 2: Markov chain with aggregated states and transition rates between them for hierarchical LRC

4 Results

First, we establish Reed–Solomon (n, k)-scheme as reference erasure code. The approximate expression for a block MTTDL is derived from classical analytical model based on Markov chains [5]

$$MTTDL_{Block} = \frac{\mu^{n-k}}{\lambda^{n-k+1} \prod_{l=0}^{n-k} (n-l)}, \tag{3}$$

where μ and λ are the respective rates of disk failure and data recovery from a failed disk, n and k are the parameters of the considered erasure code.

Next we estimate the reasonable parameters for erasure codes like the disk failure rate λ and the recovery rate μ. The disk failure rate is $(200000\ hours)^{-1}$

for the specified mean time to failure (MTTF) of 200000 hours. The recovery rate has the reference value of one $(\,hour)^{-1}$. It is reasonable to rescale λ and μ into $(\,year)^{-1}$ for numerical calculation of block MTTDLs, since the matrices (1) and (2) while containing large coefficients reduce the precision of the solution of the linear system. The R-language program code to compute respective mean times to data loss is provided in the listing below.

```
lambda = (24.0*365.0)/200000.0
n = 9
k = 6
B <- matrix(c(1, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
MTFF <- matrix(c(0, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
MTFF2 <- matrix(c(0, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
MTTDL <- matrix(c(0, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
X <- matrix(c(0, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
X2 <- matrix(c(0, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
Delta <- matrix(c(0, 0 ,0 ,0, 0, 0, 0, 0, 0, 0),nrow=10)
PR <- matrix(c(n:k),nrow = 1)
PRD = prod(PR)
for (t in 1:10)
{
mu <- (24.0*365.0)/t
E <- matrix(c(
-10*lambda, 4*lambda, 3*lambda, lambda, 2*lambda, 0, 0, 0, 0, 0,
mu, -9*lambda-mu, 0, 0, 0, 4*lambda, 5*lambda, 0, 0, 0,
mu, 0, -9*lambda-mu, 0 , 0, 3*lambda, 6*lambda, 0, 0, 0,
mu, 0, 0, -9*lambda-mu, 0, lambda, 8*lambda, 0, 0, 0,
mu, 0, 0, 0, -9*lambda-mu, 0 , 9*lambda, 0, 0, 0,
mu, 0, 0, 0, 0, -8*lambda-mu, 0, 8*lambda, 0, 0,
mu, 0, 0, 0, 0, 0, -8*lambda-mu, 4*lambda, 4*lambda, 0,
mu, 0, 0, 0, 0, 0, 0, -7*lambda-mu, 0, 7*lambda,
mu, 0, 0, 0, 0, 0, 0, 0, -7*lambda-mu, 4*lambda,
mu, 0, 0, 0, 0, 0, 0, 0, 0, -6*lambda-mu),nrow=10)
X <- -qr.solve (E,B)
E2 <- matrix(c(
-9*lambda, 9*lambda, 0, 0, 0, 0, 0, 0, 0, 0,
mu, -8*lambda-mu, 6*lambda, 2*lambda, 0, 0, 0, 0, 0, 0,
mu, 0, -7*lambda-mu, 0 , 3*lambda, 4*lambda, 0, 0, 0, 0,
mu, 0, 0, -7*lambda-mu, 0, 6*lambda, lambda, 0, 0, 0,
mu, 0, 0, 0, -6*lambda-mu, 0, 0, 6*lambda, 0, 0,
mu, 0, 0, 0, 0, -6*lambda-mu, 0, 3*lambda, lambda, 0,
mu, 0, 0, 0, 0, 0, -6*lambda-mu, 0, 6*lambda, 0,
mu, 0, 0, 0, 0, 0, 0, -5*lambda-mu, 0, lambda,
mu, 0, 0, 0, 0, 0, 0, 0, -5*lambda-mu, 3*lambda,
mu, 0, 0, 0, 0, 0, 0, 0, 0, -4*lambda-mu),nrow=10)
X2 <- -qr.solve (E2,B)
MTFF[t] = sum(X)
MTFF2[t] = sum(X2)
MTTDL[t] = (mu^(n-k))/((lambda^(n-k+1))*PRD)
Delta[t] = (MTFF[t]-MTFF2[t])/MTFF[t];
}
MTFF
MTFF2
MTTDL
Delta
```

This listing can be executed in various [R] environments like online ones ([2, 7, 9]).

The log-log plots on Fig.3,4 show, that the increase in recovery time reduces the MTTDL of data block in exponential manner. One can see on the plot of the Fig.5 that the relative difference $\delta MTTDL_{Block}$ between block MTTDLs for nRLC and hLRC schemes

$$\delta MTTDL_{Block} = \frac{MTTDL^{nLRC}_{Block} - MTTDL^{hLRC}_{Block}}{MTTDL^{nLRC}_{Block}} \times 100\%$$

is about minus 19 per cent. This means that the hierarchical LRC is more preferable than the nonuniform local recovery code.

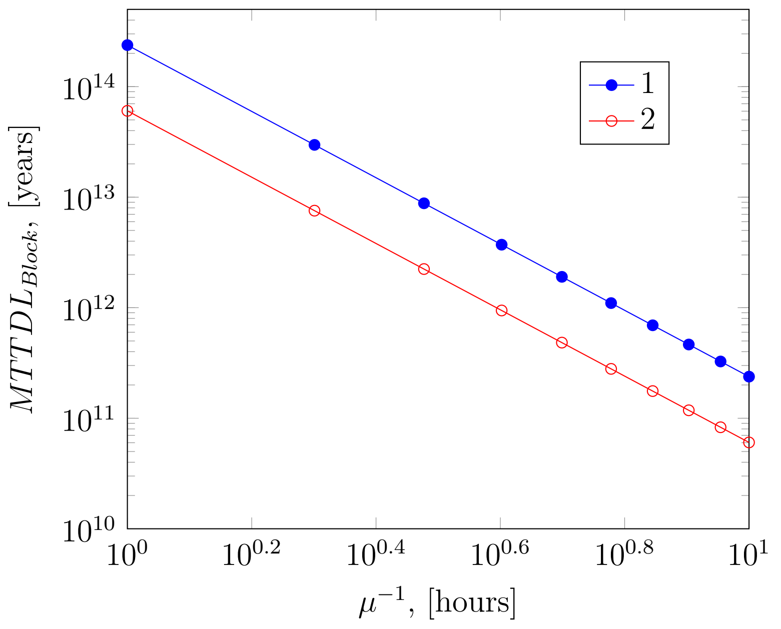

Figure 3: A block MTTDL against a mean time to data recovery for 1 – $(10, 2, 2)$ nLRC and 2 – $(9, 6)$ RS schemes

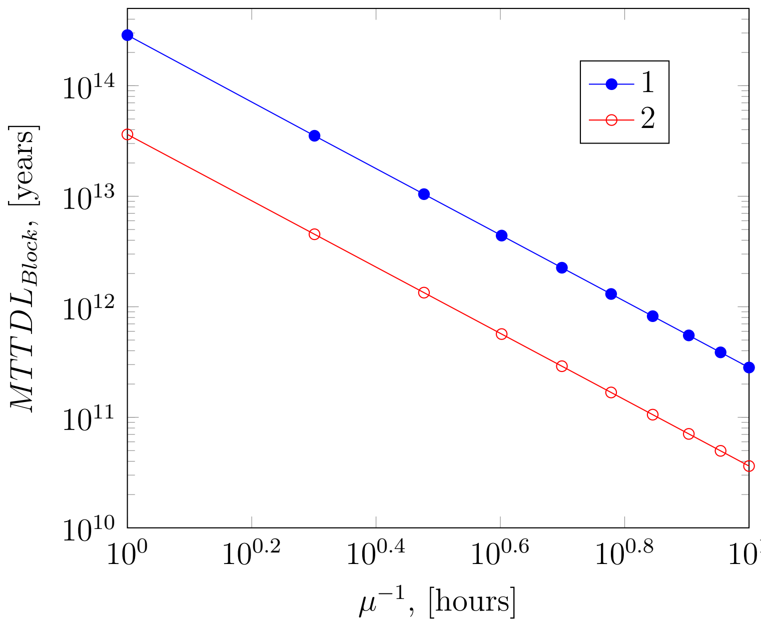

Figure 4: A block MTTDL against a mean time to data recovery for $1 - (3^2, 2^2)$ hLRC and $2 - (10, 7)$ RS schemes

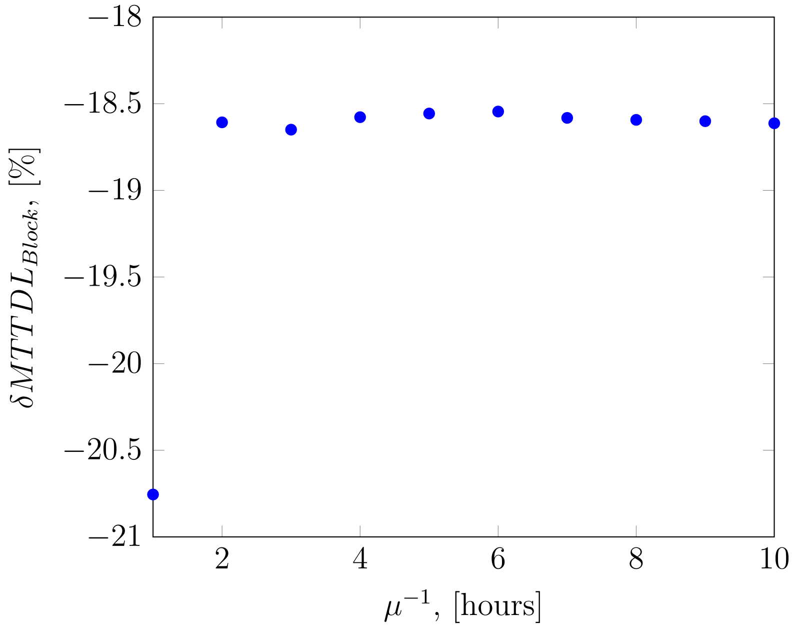

Figure 5: Difference between block MTTDLs for nLRC and hLRC schemes relative to nLRC

According to the plots on Fig.3,4, the $(9, 6)$ and $(10, 7)$ Reed–Solomon codes provide lower block reliability than both nLRC and hLRC schemes, since the

LRCs tolerate not only three erasures, but also some of four- and five-erasure patterns. At the same time, the four-erasures-tolerant Reed–Solomon codes give the reliability of more than four orders higher than any of considered LRC (see Tables 3,4) and five-erasures-tolerant $(9,4)$ RS has the MTTDL of eight–nine orders higher. Since the MTTDL values of nLRCs with guaranteed three-erasures tolerance $\epsilon = 3$ for the initial block size $k = 6$ are situated in between the MTTDL values of (n,k) RS code, corresponding to $n = k + \epsilon = 9$ and $n = k + \epsilon + 1 = 10$, we can regard the former as a $(\hat{n}, k)$ RS code having a noninteger $\hat{n} \in (9.0;\ 10.0)$. To obtain equivalent value of $\hat{n}$ we revisit equation (3) and introduce the following formal expansion of positive integer argument $n \in \mathcal{N}$ over the set of real numbers $\mathcal{R}$

$$\frac{1}{\prod_{l=0}^{n-k}(n-l)} = \frac{(k-1)!}{n!} = \frac{(k-1)!}{\Gamma(\hat{n}+1)},\ \hat{n} = n, \tag{4}$$

where $\Gamma(\cdot)$ is Euler's gamma function.

Therefore, substituting equation (4) into (3) one can get the equivalent MTTDL computed as

$$\widehat{MTTDL}_{Block}^{RS} = \frac{\mu^{\hat{n}-k} \cdot (k-1)}{\lambda^{\hat{n}-k+1} \cdot \Gamma(\hat{n}+1)}. \tag{5}$$

Minimizing the Euclidian norm of a relative difference $\hat{\delta}$ between MTTDL for LRC and equivalent MTTDL for RS code

$$\hat{\delta} = \sqrt{\sum_{\mu^{-1}=1}^{10} \left(\frac{MTTDL_{Block}^{LRC} - \widehat{MTTDL}_{Block}^{RS}}{MTTDL_{Block}^{LRC}} \right)^2} \times 100\%, \hat{n} = \arg\min_{\hat{n}} \hat{\delta} \tag{6}$$

we get $\hat{\delta}_{nLRC} \approx 36.7\%$ and $\hat{n}_{nLRC} \approx 9.159$. To compute the equivalent MTTDL for hLRC the specified and obtained values of k, $\hat{n}_{hLRC}$ and $\hat{\delta}_{hLRC}$ respectively are as follows: $k = 4$, $\hat{n}_{hLRC} \approx 7.029$, $\hat{\delta}_{hLRC} \approx 8.2\%$. Since the primary governing parameter for a MTTDL is a number of tolerable erasures ϵ, the provided equations permit the substitution $\hat{n} = k + \hat{\epsilon}$ and allow one to compare local reconstruction and Reed–Solomon codes with different initial block sizes k. Thus, the $(10,2,2)$ nLRC can be formally approximated via $(4+\hat{\epsilon}, 4)$ RS code with $\hat{\epsilon}_{nLRC} \approx 3.01$ giving $\hat{\delta}_{nLRC} \approx 3.3\%$ (see Table 4). Similarly, the $(3^2, 2^2)$ hLRC has the best fit of $(6+\hat{\epsilon}, 6)$ RS code with $\hat{\epsilon}_{hLRC} = 3.178$ resulting in $\hat{\delta}_{hLRC} = 42.9\%$ (see Table 3).

Because the equations (3), (4) are only approximate while neglecting the high order λ and μ terms, there is a variability of $\hat{n}$ along the recovery time axis μ^{-1} on Fig.6. However, this variation can be mitigated via applying more precise analytical expressions for block MTTDL.

Table 3: Block MTTDL for $(6+\hat{\epsilon}, 6)$ Reed–Solomon codes

μ^{-1}, [hours],	1	2	3	4	5	6	7	8	9	10
MTTDL, [years], $\hat{\epsilon}=3$	6.04 10^{13}	7.55 10^{12}	2.24 10^{12}	9.44 10^{11}	4.83 10^{11}	2.80 10^{11}	1.76 10^{11}	1.18 10^{11}	8.28 10^{10}	6.04 10^{10}
MTTDL, [years], $\hat{\epsilon}=3.159$	2.94 10^{14}	3.28 10^{13}	9.13 10^{12}	3.68 10^{12}	1.82 10^{12}	1.02 10^{12}	6.28 10^{11}	4.12 10^{11}	2.84 10^{11}	2.03 10^{11}
MTTDL, [years], $\hat{\epsilon}=3.178$	3.55 10^{14}	3.92 10^{13}	1.08 10^{13}	4.33 10^{12}	2.13 10^{12}	1.19 10^{12}	7.31 10^{11}	4.78 10^{11}	3.29 10^{11}	2.35 10^{11}
MTTDL, [years], $\hat{\epsilon}=4$	1.21 10^{18}	7.55 10^{16}	1.49 10^{16}	4.72 10^{15}	1.93 10^{15}	9.32 10^{14}	5.03 10^{14}	2.95 10^{14}	1.84 10^{14}	1.21 10^{14}

Table 4: Block MTTDL for $(4+\hat{\epsilon}, 4)$ Reed–Solomon codes

μ^{-1}, [hours],	1	2	3	4	5	6	7	8	9	10
MTTDL, [years], $\hat{\epsilon}=3$	2.17 10^{14}	2.72 10^{13}	8.05 10^{12}	3.40 10^{12}	1.74 10^{12}	1.01 10^{12}	6.34 10^{11}	4.25 10^{11}	2.98 10^{11}	2.17 10^{11}
MTTDL, [years], $\hat{\epsilon}=3.01$	2.41 10^{14}	2.99 10^{13}	8.82 10^{12}	3.31 10^{12}	1.90 10^{12}	1.09 10^{12}	6.88 10^{11}	4.61 10^{11}	3.23 10^{11}	2.35 10^{11}
MTTDL, [years], $\hat{\epsilon}=3.02$	2.67 10^{14}	3.29 10^{13}	9.66 10^{12}	4.05 10^{12}	2.06 10^{12}	1.19 10^{12}	7.48 10^{11}	4.99 10^{11}	3.50 10^{11}	2.55 10^{11}
MTTDL, [years], $\hat{\epsilon}=4$	5.44 10^{18}	3.40 10^{17}	6.71 10^{16}	2.12 10^{16}	8.70 10^{15}	4.19 10^{15}	2.26 10^{15}	1.33 10^{15}	8.28 10^{14}	5.44 10^{14}

As the comparison shows, the two-tier $(3^2, 2^2)$ hLRC has a significantly larger resulting spatial overhead of five parity fragments than its $(7, 4)$ Reed–Solomon counterpart does. At the same time, the $(9, 4)$ RS code greatly outperforms the hierarchical one by several orders of magnitude. However, the increased spatial overhead emerges as a tradeoff for reduced encoding/decoding

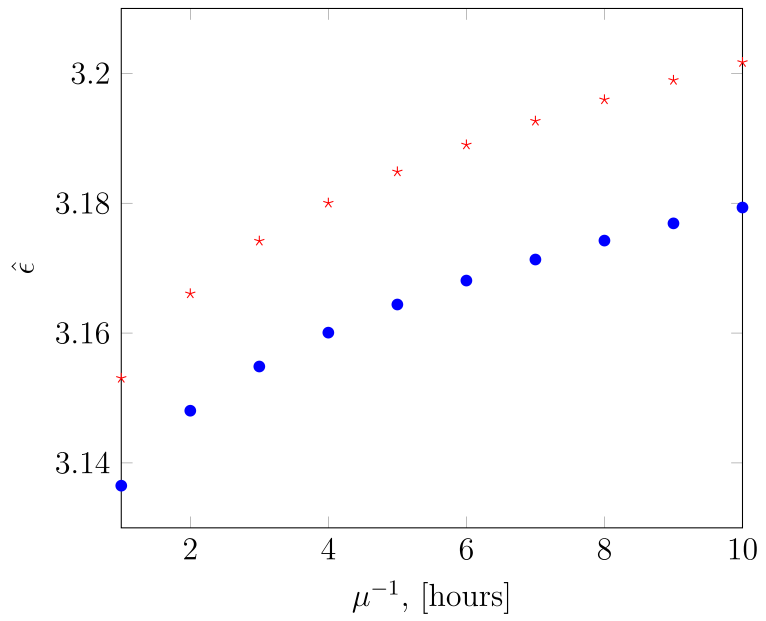

Figure 6: Dependence of $(6+\hat{\epsilon},6)$ RS equivalent erasure tolerance $\hat{\epsilon}$ on recovery time for ● – $(10,2,2)$ nLRC and ⋆ – $(3^2,2^2)$ hLRC schemes

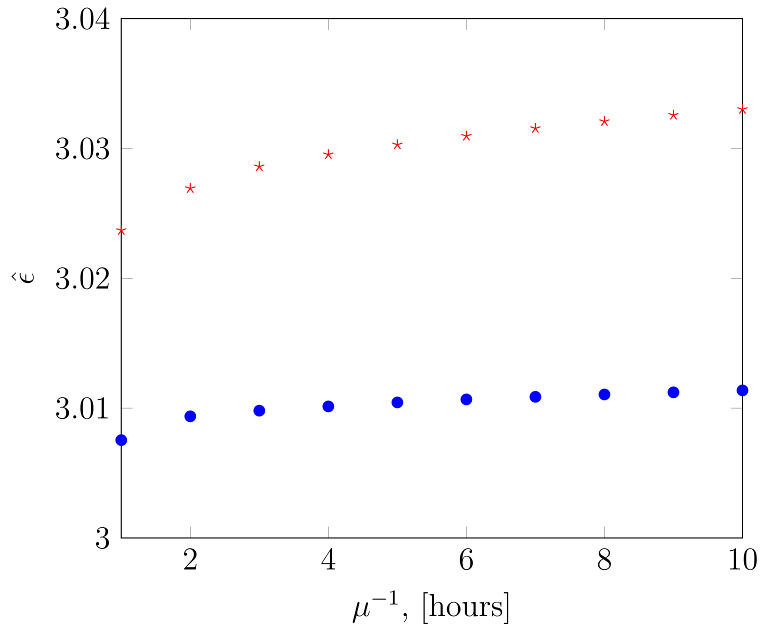

Figure 7: Dependence of $(4+\hat{\epsilon},4)$ RS equivalent erasure tolerance $\hat{\epsilon}$ on recovery time for ● – $(10,2,2)$ nLRC and ⋆ – $(3^2,2^2)$ hLRC schemes

computational costs. In other words, the direct systematic RS encoder has complexity of $\mathcal{O}((n-k)k)$ (e.g. see [8]), so the $(7,4)$ variant takes $\mathcal{O}(12)$

operations on $\mathcal{GF}\left(2^w\right)$, whereas the two-tier $\left(3^2, 2^2\right)$ variant requires $\mathcal{O}\left(2\right)$ operations on $\mathcal{GF}\left(2^{2w}\right)$ and $3\mathcal{O}\left(2\right)$ operations on $\mathcal{GF}\left(2^w\right)$.

5 Conclusion

The conducted study confirmed the widely used thesis that local reconstruction codes can be used as an alternative to traditional Reed–Solomon codes to reduce the average number of computing and network operations at a cost of a larger resulting block size at the same level of reliability. At the same time, the guaranteed erasure tolerance of Reed–Solomon code with the same initial and resulting block sizes is higher at least by one erasure. The findings indicate that the resulting number of guaranteed tolerable failures in multi-tier schemes serves as the initial approximation to the $(\hat{n} - k)$ value of equivalent RS $(\hat{n}, k)$ code and is a primary contributor to the data block reliability. The comparison also revealed that the nonuniform local reconstruction code requires lower spatial overhead than the hierarchical counterpart. However, the hLRC saves more computational resources while performing finite field arithmetical operations and comply more flexibly with various configurations. We demonstrated that the proposed technique could be applied to represent an arbitrary erasure code as a Reed–Solomon code with the equivalent noninteger number of tolerable erasures. In this case the sought-for value of $\hat{\epsilon}$ serves as a measure of introduced redundancy to compare erasure-tolerant coding schemes.

Acknowledgements. The article is published within applied scientific research performed with financial support of the Ministry of Education and Science of the Russian Federation. Subsidy provision agreement 14.579.21.0010. Universal identifier of the agreement is RFMEFI57914X0010.

References

[1] J.A. Buzacott, Markov approach to finding failure times of repairable systems, *IEEE Transactions on Reliability,* **R-19** (1970), no. 4, 128-134. http://dx.doi.org/10.1109/tr.1970.5216431

[2] Coding Ground, Computer software, Tutorials Point. Web. http://www.tutorialspoint.com/execute_r_online.php

[3] Constellation ES, High-capacity storage designed for seamless enterprise integration, Data Sheet, Seagate. http://www.seagate.com/docs/pdf/datasheet/disc/ds_constellation_es.pdf

[4] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, Jin Li, and S. Yekhanin, Erasure coding in windows azure storage, *Presented as part of the 2012 USENIX Annual Technical Conference (USENIX ATC 12),* (2012), 15-26.

[5] L. Ivanichkina and A. Neporada, Computer simulator of failures in super large data storage, *Contemporary Engineering Sciences,* **8** (2015), no. 35, 1679-1691. http://dx.doi.org/10.12988/ces.2015.59270

[6] L. Ivanichkina and A. Neporada, Mathematical methods and models of improving data storage reliability including those based on finite field theory, *Contemporary Engineering Sciences,* **7** (2014), no. 28, 1589-1602. http://dx.doi.org/10.12988/ces.2014.411236

[7] R-Fiddle, Computer software, Web. `http://www.r-fiddle.org/#/`

[8] A. Soro and J. Lacan, FNT-based Reed-Solomon erasure codes, *2010 7th IEEE Consumer Communications and Networking Conference,* (2010), 1-5. http://dx.doi.org/10.1109/ccnc.2010.5421749

[9] Sphere Engine, Computer software, ideone.com Web. `http://ideone.com`

[10] I. Tamo, D.S. Papailiopoulos and A.G. Dimakis, Optimal locally repairable codes and connections to matroid theory, *2013 IEEE International Symposium on Information Theory,* (2013), 1814-1818. http://dx.doi.org/10.1109/isit.2013.6620540

Received: June 14, 2016; Published: August 29, 2016