Contemporary Engineering Sciences, Vol. 7, 2014, no. 28, 1589 - 1602
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ces.2014.411236

Mathematical Methods and Models of Improving Data Storage Reliability Including Those Based on Finite Field Theory

L. Ivanichkina and A. Neporada

Altufievskoye Shosse 44, Acronis, 127566
Moscow, Russia

Abstract

The trend towards long-term explosive data growth that has been getting stronger in recent years leads to changing requirements for reliable and scalable storage systems. Contemporary approaches to reliability based on hardware implementations reveal bottlenecks that lead to increasing risk of data loss on large and growing volumes of data. Novel approaches already exist which are based on software methods to ensure data consistency over time in large data sets, but efficiency of these approaches has not been extensively explored in practice yet, especially because experimenting with petabyte scale systems in real world requires substantial investment of time and resources. Therefore, mathematical modeling of new methods becomes an important means to identification of most efficient solutions for problems of reliable and scalable data storage.

Keywords: Improving Data Storage Reliability, Finite Field Theory

1. Introduction

According to International Data Corporation (IDC) [1], in 2012, the total global volume of digital data was 2837 EB (1 EB (exabyte) = GB (gigabyte)) and it is prone to double every two years reaching 40,000 EB or about 5,200 GB per each person on the planet by 2020. Relevance of providing the required level of

reliability for stored data is increasing as the volume of created, stored, and consumed data grows exponentially.
On the one hand, as the volume of stored data in an organization is rising, the costs related to the implementation of scalable and robust data storage system that would be able to provide efficient access to information increase but, on the other hand, the risks of data loss due to hardware failure are increasing. According to research conducted by Google [2], the probability of hard disk drive failure is 5-8% per year. This means chances are that in large organizations storing hundreds of terabytes to tens of petabytes of data at least one HDD fails every day. These conditions render typical approaches like replication or RAIDs increasingly less efficient since they are unable to provide a high level of reliability with relatively low redundancy. For example, if one of the HDDs in a RAID fails, it may take up to several hours to restore data on this drive after its replacement. Bearing this in mind, the possibility of another drive failure in this RAID before the restore procedure is completed should be taken into account.

All the aspects listed above make it necessary to search for and apply new solutions that would allow for creating data storage systems where a high level of reliability and availability is combined with the capability to infinitely scale up the solution for future growth. At the same time, they should provide for an affordable total cost of ownership for businesses.

2. Overview of Major Approaches to Increase Data Storage Reliability

Provision of data storage reliability involves both error detection methods and methods for error correction and restore in case one or more drives with data fail or connection to one or more machines in a distributed system is lost.

Parity bit is one bit of data that is added to a stream or a set of bits allowing you to determine whether the number of 1-bits in a bit set is even or odd. It allows you to detect an odd number of errors and correct one error provided the remaining bits in the set are known. It is widely used in RAIDs to restore data.

Cyclic Redundancy Check (CRC) is a family of non-cryptographic hash functions based on the properties of division with a remainder of binary polynomials (polynomials over the GF (2) field). Thanks to high performance and the possibility of hardware implementation, these are widely used to detect errors when transferring and storing data.

Cryptographic hash functions allow for a more reliable detection of errors in data but provide much lower performance compared to CRC sums.

Replication is the storage of data copies on several drives or machines in a distributed system. As a rule, three copies of data are stored in systems with replication. On the one hand, it allows you to detect one replica with errors if it is different from the two other identical replicas. On the other hand, storing at least three replicas allows you to avoid situations when the drive that is used as a source for restoring data after the second replica drives failure breaks down, which may cause data loss. Compared to other approaches, significant redundancy is the drawback of replication.

RAID (Redundant Array of Independent Disks) is a hardware way to provide for data storage reliability. It is a set of several drives managed by a hardware controller and connected to each other by high speed data transfer channels. External systems consider this array as a single drive. Depending on the array type, it allows you to provide a different degree of failure tolerance.

There are several levels of RAID specification, with RAID5 and RAID6 being the most appropriate ones for data storage reliability. In both schemes, data is split into blocks that are repeatedly written onto the array drives along with one (RAID5) or two (RAID6) blocks of checksums.

With n drives in a RAID5 array, for every $(n-1)$ blocks of payload one checksum block is calculated as XOR of the remaining blocks. Every n blocks of data is written with a shift on one drive in relation to the previous block set allowing you to increase the data read speed by means of parallelization. The minimum number of drives in RAID5 is three. The larger the number of drives in the array is, the higher the data read speed will be.

In RAID6, two blocks are allocated for checksums that are calculated by two different algorithms. This allows you to survive when two drives fail. For example, the RAID6 scheme implemented in Red Hat Linux uses checksums based on algebraic operations in Galois fields [7]. The minimum number of drives in RAID6 is four. Figure 1 shows an example of data and checksum block distribution for a RAID6 array of 5 drives. Source data blocks are marked as d_j^i, while checksum blocks are marked as p^i, q^i.

d_1^1	d_2^1	d_3^1	p^1	q^1
q^2	d_1^2	d_2^2	d_3^2	p^2
p^3	q^3	d_1^3	d_2^3	d_3^3

Figure 1. Example of data and checksum block distribution for a RAID6 array of 5 drives. The advantage of RAID5 and RAID6 is the possibility to potentially parallelize data writing and reading. For example, in theory, a RAID5 array of n drives allows you to accelerate the data reading speed by $(n-1)$ times provided that the data blocks are intact and a restore is not required.

The disadvantage of RAID5 and RAID6 is a low number of disk failures that can be survived without data loss. This led to the creation of RAID schemes including those based on Reed–Solomon codes [8] that would be able to survive a larger number of failures.

Error-correcting coding is data encoding schemes allowing you to split the object into k fragments of the same length and recode them into n fragments so that the original object could be restored from any k fragments of the resulting n fragments. Cases when fragments could be not only lost but also corrupted require a special mechanism based on checksums, for instance to detect corrupted fragments.

Let's consider w character long words from the $\{0,1\}$ alphabet. (n, k) encoding scheme for each set of k w-bit words maps to the set of n w-bit words where $n > k$. As a rule, 8 or 16 is selected as w to work with 1 or 2 byte long words, respectively. To apply error-correcting coding to a file of random length, it should be split into sets of w-bit words and each of these sets must be encoded. After that the n resulting words for each set are written onto n machines in a distributed system.

The $\frac{k}{n}$ ratio is called a rate of encoding, whereby the size of disk space required for storing the object for the r rate of encoding increases by $\frac{1}{r}$ times. Approaches like replication and RAID systems that were described above are examples of error-correcting codes. Replication with three replicas, for example, may be described as $(n = 3, k = 1)$ error-correcting code, RAID5 of three drives as $(n = 3, k = 2)$ code and RAID6 of four drives as $(n = 4, k = 2)$ code.

Consider the dependence of the availability of data on the parameters of erasure code (erasure-resilient code, or error-correcting code) [3]. We assume that a system consists of a collection of N hard drives (N machines in a distributed system), and M of these machines (servers) are temporarily unavailable. Consider that there is a data block, divided into n fragments in such a way that data can be reconstructed from any$k$ fragments. To compute a probability P_0 that a block is available, i.e. the data block can be reconstructed from fragments, we sum the conditional probabilities of data block availability upon the condition that some fragments, from 0 to $(n - k)$ fragments, are not available. Given the number of unavailable fragments is i, the probability that the block is available is equal to the number of ways in which we can arrange unavailable fragments on unreachable servers multiplied by the number of ways i in which we can arrange available fragments $(n - i)$ on reachable servers, divided by the total number of ways in which we can arrange all of the fragments on all of the servers. In this case, the probability P_0 is equal to:

$$P_0 = \sum_{i=0}^{n-k} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{n}}$$

For replication with n copies of data block, the availability $P_{replication}$ of a block can be computed by the formula for P_0 at $k = 1$.

Schemes of erasure-resilient coding allow you to select suitable (n, k) parameters for the coding scheme such that systems employing these erasure codes have a reliability that is considerably higher than replicated systems with similar storage redundancy. Example: a distributed system consists of 1,000,000 machines, and 10% of these machines are not available. When using (32, 16) scheme, the availability of a data block is equal to 0.999999998. When employing replication with the same redundancy (two copies of each data block), the availability of a data block is no better than 0.99.

With a suitable choice of parameters of erasure-resilient coding, mean time to failure (MTTF) of an erasure encoded system can also be shown to be significantly higher than that of a replicated system with the same redundancy [3].

Hence, erasure-resilient coding enables many orders of magnitude higher reliability than that of a replicated system with the same redundancy.

Geo-replication. This approach uses geo-replication systems that store copies of data across geographically distributed data centers. Such an approach provides data safety, as well as the continuous running of the system during natural disasters (water flood, earthquake or fire situation in a data center) or technogenic accidents (global electrical power outage).

Storing data at geographically distributed data centers necessarily imposes some restrictions on distributed data storage design. For example, with a cross-network delay during data communication between data centers it can be complex to implement "read-after-write consistency" (data consistency model, when data changes made by a user can be automatically seen by other user). More real model for geo-replication is "eventual consistency" (data consistency model, when all of data changes will be reliably available for all the users sooner or later).

3. Erasure-Resilient Codes

3.1 Basic concepts

Consider a finite field (Galois field) F_q, where q is a number of elements of the field. $u \in F_q^k$ is a k-dimensional vector with vector components from F_q. Code C maps vector $u \in F_q^k$ to vector $x \in F_q^n$, where $n > k$, i.e. it maps a k element set from F_q to a n element set from F_q.

We will consider only linear codes, when the given mapping can be written as the production of original k-dimensional vector u by generator matrix G:

$$uG = x$$

Vector u is a vector of information symbols (or information vector), and vector x is a code word (code vector).

There are many of erasure-resilient codes (or error-correcting codes), but we will be mostly interested in the **systematic codes**, when a set of coefficients of derived n-dimensional vector contains k coefficients of original k-dimensional vector. Remaining $(n-k)$ coefficients are check symbols. Generator matrix for systematic code must contain unity submatrix with an accuracy to the permutation of rows (obviously, the properties of code are independent of the permutation of rows of the generator matrix), and it can be written as $G = [I_k \quad A]$, where I_k is the unity submatrix. Therefore, if data were not corrupted or lost, there is no need to perform the complex process of decoding while reading data because it is enough to read k fragments that match the original words.

To determine if the vector x is a code word (code vector), you use parity-check matrix H, that meets the following condition: $Hx^T = 0$, if and only if the vector x is a code word (code word). For a systematic code, the parity-check matrix is

$$H = [-A^T \quad I_{n-k}].$$

The number of errors that can be corrected with the help of error-correcting code (erasure-resilient code), is tightly linked with a concept of Hamming distance. For the two vectors x and y the Hamming distance between these vectors is the number of coefficients in which they differ, which is the size of set $\{i: x_i \neq y_i\}$. For code C the Hamming distance is the minimal distance between any two different code words.

Singleton bound is a theoretic bound on the maximum number of code words in a code with Hamming distance d between dimensional vectors with coefficients from the field F_q:

$A_q(n,d) \leq q^{n-d+1}$, where $A_q(n,d)$ is a code size that is the size of the set of all the possible code words.

For the linear code $A_q(n,d) = q^k$ therefore, we have the following formula:

$$k \leq n - d + 1$$

Consequently, linear (n,k)-code that achieves equality in the Singleton bound can correct $(n-k)$ errors. Such a code is called **Maximum Distance Separable code** (MDS code). The maximum distance of the separable codes enables you to correct the maximum number of errors in comparison with other erasure codes with the same n and k parameters, so that MDS codes are the most effective codes in terms of redundancy.

Let the code be denoted by generator matrix G. Consider a recovery of source data from a partially known code word. The following equation is valid for any vector of source data:

$$\begin{pmatrix} G_1^1 & \dots & G_k^1 \\ \dots & \dots & \dots \\ G_1^n & \dots & G_k^n \end{pmatrix} \begin{pmatrix} u_1 \\ \dots \\ u_k \end{pmatrix} = \begin{pmatrix} c_1 \\ \dots \\ c_n \end{pmatrix}$$

It is clear from the matrix multiplication rule that each coefficient of the obtained code word corresponds to its row of the generator matrix.

Let there be given a code word x, for which we have $(n-k)$ unknown coefficients. In this case, we remove those rows from the generator matrix, which corresponds to unknown coefficients, and then we obtain a matrix G'. To obtain the inverse of a matrix G', using an additive and multiplicative rules in the given finite field. Let x' be obtained from the code word x via the removal of unknown coefficients, then the original vector u can be found by the following formula:

$$\begin{pmatrix} u_1 \\ \cdots \\ u_k \end{pmatrix} = G'^{-1}x'$$

Note that data recovery requires that any square submatrix of matrix G is inversible.

3.2 Characteristics of error-correcting codes

Error-correcting codes (erasure-resilient codes) can be classified by the following parameters.

- The mean number of errors and guaranteed number of errors that can be corrected (disk outage, unavailability of remote machines in distributed system). For non-MDS codes, the guaranteed number of errors differs from the mean number of errors that can be corrected.
- A degree of coding and its reciprocal value that characterizes a storage redundancy.
- Theoretical algorithmic complexity of coding, decoding, and updating of data.
- Ability to change the number of coding symbols and to recover the individual coding symbols with no need for complete decoding and coding of data using error-correcting coding scheme with new parameters.

3.3 Reed-Solomon codes

3.3.1 Description of Reed-Solomon codes

Reed-Solomon codes belong to a class of linear maximum distance separable codes. These codes exist for any parameters (n,k). Reed-Solomon codes are defined by generator polynomial over Galois fields with roots in the same field.

Consider a Galois field $GF(N)$, where N is a power of a prime integer. For N, we usually take 2^8 or 2^{16}. Elements of this field can be represented as polynomials of degree equal to or less than $(n-1)$ with the coefficients 0 and 1. Addition and multiplication in the Galois field is performed modulo some irreducible polynomial, i.e. polynomial that may not be factored into the product of two non-constant polynomials. The addition of two polynomials is calculated
as a polynomial with the coefficients equal to the XOR of the corresponding coefficients of the two polynomials. Multiplication is a more complex operation and requires special tables for the efficient computation.

Since Reed-Solomon code is a linear code, it can be specified by a generator matrix. Then source data can be decoded according to the aforementioned algorithm with the inversion of the submatrix of the generator matrix. Vandermonde matrix can be used a generator matrix for the Reed-Solomon codes:

$V = \begin{pmatrix} 1 & a_1 & a_1^2 & \dots & a_1^{k-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & a_n & a_n^2 & \dots & a_n^{k-1} \end{pmatrix}$, where $(a_1, \dots, a_n)$ are the different non-zero elements of the given Galois field $GF(N)$. This approach is based on the fact that every k vectors like $\left(1, a_i, \dots, a_i^{k-1}\right)$, corresponding to the k different elements of the Galois field, form a basis and any square submatrix of this Vandermonde matrix is inversible.

Theoretical complexity of the basic operations for Reed-Solomon code is equal to$O(n(n-k))$ for coding and decoding of data, and $O(n-k)$ for updating the code word after the change of one coefficient of the input vector (vector of source data).

Reed-Solomon codes enable you to flexibly increase or decrease the dimensionality of code words by adding or removing checking symbols and to restore any single symbol in a code word with no need to re-encode the source data from scratch.

Cauchy code represents a more efficient modification of Reed-Solomon codes, which uses the Cauchy matrix as a generator matrix, and replaces multiplication in the Galois field with XOR.

3.3.2 Advantages and drawbacks of Reed-Solomon codes

The main advantages of the Reed-Solomon codes are the following: existence of the coding algorithm for the any given n and k parameters, optimal number of the errors corrected, and also the ability to increase the number of symbols in coding word and to restore any single symbol in coding word without the need to re-encode the source data. The drawbacks of Reed-Solomon codes and their modifications: relatively low coding and decoding rates, because of the computation of each checking symbol requires n multiplications, as well as an inefficient data recovery, because it requires the complete decoding of the source data. The problem of the low performance of the Reed-Solomon codes was partly solved in the modifications of the Reed-Solomon codes, such as Cauchy code.

3.4 Parity-array codes

Parity-array codes are XOR-based codes. Simplicity of the implementation of parity-array codes enables its application in RAIDs. In the general case, parity-array codes are not maximum distance separable codes. Contrary to Reed-Solomon codes, which are applied to k w-bit symbols to obtain n w-bit symbols, parity-array codes are applied to a set of k r-dimensional arrays of w-bit symbols to obtain n r-dimensional arrays.

Parity-array codes can be divided into vertical codes and horizontal codes. Horizontal codes are used for storing either source data or code symbols on each hard drive, while vertical codes are used for storing both source data and code symbols on each hard drive.

The best known and widely applied parity-array codes are EVENODD, X-code, Weaver, HoVer, and Blaum-Roth codes.

3.4.1 Evenodd

EVENODD [10] is an error-correcting code with the parameters $k = p, n = p + 2$, where p is a prime number, $p > 2$. Let us consider an example of EVENODD code for $(n = 7, k = 5)$. Coding/decoding algorithms can be easily extended for other values.

Let us assume that the source data $\begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} & d_{0,4} \\ d_{1,0} & d_{1,1} & d_{1,2} & d_{1,3} & d_{1,4} \\ d_{2,0} & d_{2,1} & d_{2,2} & d_{2,3} & d_{2,4} \\ d_{3,0} & d_{3,1} & d_{3,2} & d_{3,2} & d_{3,4} \end{pmatrix}$ are placed across 5 hard drives, where $d_{i,j}$ data are placed on one hard drive for any specified j. We want to compute two data arrays for two hard drives with check sums based on the source data.

For the first hard drive with check sums, we compute a vector C_0, each coefficient of which is equal to the result of XOR of the elements in the corresponding row in the source data table. For the second hard drive with check sums, we compute the coefficients of vector C_1 as XOR of words, which are placed on one diagonal of the table.

EVENODD code can be adapted for any arbitrary k by adding "virtual drives" with zero data so that the number of drives with real or virtual data was the next prime integer greater than k.

The advantage of EVENODD code is a capability to adapt this scheme for any k, and high coding rate due to XOR operations. Disadvantages of EVENODD code is a low updating and decoding rate in the worst case scenario.

3.4.2 X-code

X-code is a maximum distance separable code for $n = p, k = p - 2$, where p is a prime number. This code provides optimum performance. For this code, the arrangement of source data and code symbols in RAID can be represented with a table in which two first rows $(p - 2)$ contain source data, and two last rows contain code symbols.

Coefficients in each row with code symbols are computed as the XOR of words, which are placed on diagonals of the source data table. For one of these rows, we take back slash diagonals, and for the other row we take other diagonals.

3.4.3 Weaver

Weaver [11] is a family of vertical parity-array codes that allow you to survive up to 12 disk failures. There are two maximum distance separable codes with optimal performance for $k = 2, n = 4$ и $k = 3, n = 6$ cases. One of the key features of WEAVER codes is the fact that every data block participates in calculating the fixed number of checksum blocks allowing you to limit the top overhead for some operations.

3.5 Low Density Parity Check codes

3.5.1 Description

Low Density Parity Check codes (LDPC codes) [13] is a family of XOR based codes described using the Tanner graph, special bichromatic graphs. Performance of these codes is much higher compared to maximum distance separable codes. They are asymptotically separable codes with maximum distance.

To set an LDPC code with parameters (n, k) it takes a bipartite graph with k apexes corresponding to the source data on the one side and $(n - k)$ apexes on the other side corresponding to code symbols. It is required for at least one verge to be part of each of the code apexes. In this case, to code the source-dimensional vector, the vector coefficients are assigned to the apexes of the bipartite graph on one side of the graph and to each code symbol of the apex on the other side of the graph, and each code symbol is calculated as modulo 2 sum of all the coefficients mapped to the apexes having the common verge with the apex for this coefficient.

With the set encoding degree $k \to \infty$ the ratio of the code symbol number $(n - k)$ to the number of errors of the code is able to correct on average works for 1, which means the code is asymptotically maximum distance separable.

Despite good practical properties, the most well-known LDPC codes (Raptor [14], Tornado [15]) are covered by patents and cannot be freely used commercially.

4. The Comparison Table Containing the Main Features of the Covered Approaches

Code	Max. number of errors	Coding complexity	Decoding complexity	Updating complexity
RAID5, n drives	1	$O(n)$	$O(n)$	$O(1)$
RAID6, n drives	2	$O(n)$	$O(n)$	$O(1)$
Reed–Solomon with parameters (n, k)	k	$O(n(n-k))$	$O(n(n-k))$	$O(n-k)$
EVENODD	2	$O(n^2)$	Specific case dependent	$O(1)$ or $O(n)$ depending on specific case
X-code	2	$O(n^2)$	$O(n)$ for each of failed drives	$O(1)$
LDPC	Depends on code	$O(n)$	$O(n)$	$O(1)$

5. Data Restore Methods with the Effective Network and Disk Usage

Naive strategy to restore data encoded using the error-correcting scheme (n, k) is as follows: decode the object from the remaining correct fragments, then encode the object using the same scheme to obtain missing fragments and write them on the disk or transfer over to the network. The strategy has several apparent drawbacks:

- High cost of communication: you need to read one fragment from each of the k drives or load one fragment from each of the k machines in a distributed system.
- High cost of computing resources entailed by encoding and decoding fragments.
- The machine performing the restore process becomes a bottleneck.
- The time required to restore data is the critical aspect taking into account the possibility of consecutive failures.

Thus, in order to be efficient in a distributed system with frequent disk failures, a system of error-correcting coding should have one or more of the following properties:

- efficient use of the network and disk.
- load balancing, capability to parallelize the restore process, and to decrease the time required to restore data.
- capability to restore several errors.

Let us consider one of the possible approaches to error-correcting coding optimized for restoring data frequently that is based on the Reed-Solomon codes.

Locally Repairable Codes (LRC) [16] are error-correcting codes converting k input symbols into n code symbols so that in order to restore any one of n code symbols just r of other code symbols are enough.

Let us consider one of the ways to obtain locally repairable code. Assume that we have Reed–Solomon code with parameters $(m = n\frac{r}{r+1},\ k)$. Let's split m code symbols into $\frac{m}{r}$ groups of r symbols so that they would not overlap. Then, we apply error-correcting coding with the parameters $(r + 1, r)$. to each of these groups. The array of resulting code symbols will represent the code that can be repaired locally allowing you to restore any code symbol from r code symbols.

Use of locally repairable codes allows you to limit the network and disk bandwidth that are needed for the data restore in a distributed system and to reduce the average time required to restore a block of data by increasing the redundancy of data storage.

6. Overview of Methods to Increase Reliability of Data Storage in Actual Systems

Reliability of data storage is one of the key problems for modern distributed systems. It is solved differently depending on the tasks the system performs. Let us briefly consider approaches used in some modern systems.

Ceph, an open distributed file system, supports both data replication and error-correcting coding using user selectable advanced algorithms (different variants of Reed–Solomon codes, Cauchy code, Blaum-Roth, Liberation, Liber8tion) [18]. Ceph positions the storage using error-correcting coding as "cold" storage with high latency and low data access speed designed to store large (~1 GB) objects that remain unchanged after writing.

Wuala [19], a social distributed cryptographic file system, uses Reed–Solomon code based error-correcting coding and georeplication to provide high reliability and availability of storage.

Facebook has developed and is using a proprietary solution for error-correcting coding, locally repairable codes (LRC), allowing you to reduce the volume of disk and network traffic compared to typical solutions based on Reed–Solomon codes.

7. Plans for Future

As one of the possible directions for further work, it is proposed to develop and implement a computer model to study the performance of distributed systems. Depending on different parameters, the model should use various solutions based on error-correcting coding algorithms. It looks interesting to investigate the dependence of such system performance on the following parameters:

- data transfer speed in the network
- average latency of data transfer
- network load
- network topology

Development of such models will allow you to predict the most efficient approaches to be used in various typical scenarios of the data storage operation.

8. Conclusion

This work considers the typical approaches that are used to provide reliability and availability of data storage in modern distributed systems. It has been shown that the typical approaches like replication and RAIDs are losing their relevance in view of the exponential growth of the stored data. Thus, RAIDs, for example, provide relatively low reliability due to the long time needed to restore and the small number of simultaneous disk failures that a RAID can survive without data loss.

Error-correcting coding allows you to achieve higher data storage reliability compared to standard approaches, but its performance is lower. The selection of an encoding algorithm for a certain system directly depends on the system configuration, requirements, and dominant patterns of the system operation.

Among the erasure codes reviewed in this article the Reed Solomon codes and their modifications are the most universal and suitable for the wide range of problems. Reed–Solomon codes, for example, can be effectively used in distributed and peer-to-peer systems where data transfer between servers over the network is the bottleneck and the overhead of encoding algorithm implementation

can be disregarded. In systems without any constant configuration where machines can randomly become unavailable or connected to the system, flexibility of Reed–Solomon codes, i.e., the possibility to freely change the number of the coding fragments without the complete re-encoding of the object is the major advantage. The performance problems of the Reed-Solomon codes can be solved by using the modified variants of the Reed-Solomon codes, such as Cauchy codes, that combine all the advantages of the Reed-Solomon codes with the relatively high performance.
On the other hand, in RAIDs, where it is not required to flexibly change parameters and the system configuration stays the same, it is convenient to use parity-array code based schemes that can be efficiently implemented at the level of RAID controller hardware.
Locally repairable codes and other codes optimized for data restore should be considered in case of frequent disk failures in the system, where network and disk traffic required to restore data generates a significant load on the system. Under these conditions, the use of locally repairable codes allows you to limit the load on the system by way of a certain increase of data storage redundancy.

Acknowledgments. This article is published within the applied scientific research program financed by Ministry of Education of Russian Federation. Agreement № 14.579.21.0010. Universal identifier of the agreement RFMEFI57914X0010.

References

[1] C. F. Gantz J., Reinsel D., The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East. – 2012.
[2] Eduardo Pinheiro, Wolf-Dietrich Weber, Luiz Andre Barroso, Failure Trends in a Large Disk Drive Population // Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST’07). – 2007.
[3] Hakim Weatherspoon, John D. Kubiatowicz, Erasure Coding vs. Replication: A Quantitative Comparison // Proceedings of the First International Workshop on Peer-to-Peer Systems. – 2002. http://dx.doi.org/10.1007/3-540-45748-8_31
[4] Frederique Oggier, Anwitaman Datta, Coding Techniques for Repairability in Networked Distributed Storage Systems // Now Publishers Inc. Hanover, MA, USA. – 2013.
[5] James S. Plank, Erasure Codes for Storage Applications // FAST-2005: 4th Usenix Conference on File and Storage Technologies. – 2005.
[6] James S. Plank, Erasure Codes for Storage Systems. A Brief Primer // ;login: the Usenix Magazine. – 2013. – V.38. – N.6.
[7] Anvin H. P., The Mathematics of RAID-6. The Linux Kernel Archives. 2011. http://kernel.org/pub/linux/kernel/people/hpa/raid6.pdf

[8] James S. Plank, A tutorial on Reed-Solomon coding for fault-tolerance in RAID-like systems // Journal Software—Practice & Experience. – 1997. – V.27. – I.9. – P.995–1012.
http://dx.doi.org/10.1002/(sici)1097-024x(199709)27:9<995::aid-spe111>3.3.co;2-y

[9] Adam Leventhal, Triple-Parity RAID and Beyond // Magazine Queue – Development. – 2009. – V.7. – I.11.

[10] M. Blaum, J. Brady, J. Bruck, J. Menon, EVENODD: An Efficient Scheme for Tolerating Double Disk Failures in RAID Architectures // IEEE Transactions on Computing. - 1995. - V.44. - I.2. - P.192-202.
http://dx.doi.org/10.1109/12.364531

[11] James Lee Hafner, WEAVER codes: highly fault tolerant erasure codes for storage systems // FAST'05 Proceedings of the 4th conference on USENIX Conference on File and Storage Technologies. – V.4.

[12] Blaum M., Roth R.M., New array codes for multiple phased burst correction // IEEE Transactions on Information Theory. – 1993. – V.39. – I.1. – P.66–77. http://dx.doi.org/10.1109/18.179343

[13] J. S. Plank, M. G. Thomason, A Practical Analysis of Low-Density Parity-Check Erasure Codes for Wide-Area Storage Applications," DSN-2004: The International Conference on Dependable Systems and Networks, IEEE. – 2004. – P.115-124. http://dx.doi.org/10.1109/dsn.2004.1311882

[14] Shokrollahi, A. Raptor Codes // IEEE Transactions on Information Theory. – 2006. – V.52. – I.6. – P.2552-2567.http://dx.doi.org/10.1109/tit.2006.874390

[15] Michael Luby, Tornado Codes: Practical Erasure Codes Based on Random Irregular Graphs // Randomization and Approximation Techniques in Computer Science. Lecture Notes in Computer Science. – 1998. – V.1518. – P.171. http://dx.doi.org/10.1007/3-540-49543-6_14

[16] Papailiopoulos, D.S., Dimakis, A.G., Locally repairable codes // IEEE International Symposium on Information Theory Proceedings (ISIT). – 2012. – P.2771-2775. http://dx.doi.org/10.1109/isit.2012.6284027

[17] Cheng Huang, Minghua Chen, Jin Li, Pyramid Codes: Flexible Schemes to Trade Space for Access Efficiency in Reliable Data Storage Systems // Sixth IEEE International Symposium on Network Computing and Applications. – 2007. – P.79-86. http://dx.doi.org/10.1109/nca.2007.37

[18] [On the Web]
http://docs.ceph.com/docs/master/dev/osd_internals/erasure_coding

[19] [On the Web] http://www.wuala.com

Received: November 18, 2014; Published: December 2, 2014