Comparative Performance Analysis of Information Dispersal Methods

Maxim Deryabin, Nikolai Chervyakov
North Caucasus Federal University,
Stavropol, Russian Federation
{maderiabin, ncherviakov}@ncfu.ru

Andrei Tchernykh
CICESE Research Center, Ensenada, Mexico
South Ural State University, Chelyabinsk, Russia
Institute for System Programming of the RAS, Moscow, Russia
chernykh@cicese.mx

Viktor Berezhnoy, Anvar Djurabaev, Anton Nazarov, Mikhail Babenko
North Caucasus Federal University, Stavropol, Russian Federation
beregnoj@yandex.ru, anvar19971403@gmail.com, kapitoshking@mail.ru, mgbabenko@ncfu.ru

***Abstract*—In this paper, we present an analysis of information dispersal methods for using in distributed storage systems, processing, and transmission of data. We provide a comparative study of the methods most widely used in practice considering performance, reliability and cryptographic security. There are three main approaches to the information dispersal: Information Dispersal Algorithm by Rabin, Residue Number System (RNS) and Polynomial Residue Number System. We propose an efficient data recovery algorithm based on data representation in the RNS. Comprehensive experimental analysis shows that the most productive approach for bit length up to 256 bits is the use of the RNS with our developed algorithm. We show that the use of the RNS for the design of distributed storage systems, data transmission, and data processing, can significantly reduce the time of information processing.**

I. Introduction

The significant increase of the stored and processed information and the introduction of digital devices in all fields of human activity lead to problems we are all faced with [1]. Considering the tendency of increasing the use of smart edge devices (wearable devices, sensors, digital tags, etc.), many modern processing algorithms are designed as distributed.

As an example, let us consider the actively used distributed file systems and storage networks, which allow getting access to a variety of physical media [2]. Many works are devoted to the creation of multi-cloud storage systems, which allow increasing the availability of data [3] and level of confidentiality [4], [5]. These characteristics are especially important in the case of storing sensitive data (medical data, user personal and corporative information, etc.).

Besides, the development of the concepts of the Internet of Things, sensor networks, and 5G networks leads to the emergence of new challenges in the theory of networking [6] and storing. The number of devices, as well as the amount of transmitted data, are increasing. At the same time, devices for such networks are subjected to strict requirements for performance, energy consumption, size, etc. It affects user functions and wide distribution.

It promotes the development of a new types of network (i.e., MANET [7]), in which the transfer is carried out from device to device not through the central node, which can be a bottleneck, but through adjacent available devices. To this end, it is necessary to solve a complex of problems associated with the probability of message delivery, integrity, and confidentiality of the transmitted data.

The steady growth of the variety, volume, and speed of data transmission have led to the emergence of methods for distributed storage, processing and transmission of data.

Depending on the problem being solved, different requirements are made for dispersal information. The most important ones include:

- high performance,
- low redundancy, which is critical for big data;
- cryptographic security, which is important for public infrastructure.

The most popular approach to distributed data representation is Rabin's Information Dispersal Algorithm (IDA) [8]. Based on the simple idea of matrix-vector multiplication in the Galois field, this method has gained popularity and has been used in a number of projects both for multi-cloud storage [5] and local distributed data storage systems [9], [10].

The main idea of the Rabin's scheme is the redundant presentation of information, with additional parts of each data chunk. Based on redundant parts and properties of a distributed system, information can be recovered in case of loss of some of its fragments. The structural similarity of the Rabin's scheme with Reed-Solomon's codes [11] extends the application areas. Some modifications allow not only to recover data in case of parts loss, but also to diagnose damaged pieces of data and, in some cases, to correct errors.

An alternative to the Rabin's scheme is the using of the Residue Number System (RNS), which is a universal instrument for solving several problems at once. The Residue

Number System is a modular non-positional number system [12]. Like the Rabin's scheme, the RNS can be used for redundant distributed data representation. On the other hand, on the basis of RNS, it is possible to construct effective error correction codes [14]. An important feature of the RNS is the arithmetic coding of information, which can perform arithmetic operations with parts of the encoded data independently from each other. It leads to the possibility of using RNS for the implementation of homomorphic distributed information encryption [14], [15], which is important for distributed data processing, for example, in cloud computing systems.

However, complex operations of the transfer to the RNS and restoration of information from parts of the RNS, require comprehensive analysis. In this paper, we propose modifications of the algorithms for the conversion of data in the RNS and data recovery, optimized for software implementation based on processor systems with traditional architecture. Besides, we conduct a comparative analysis of the performance of data distribution and recovery algorithms.

This paper is organized as follows. Section II reviews the data dispersal methods and their applications. Section III describes the features of the basic data dispersal methods, which include the Rabin scheme and schemes based on RNS. Section IV introduces an optimized data recovery algorithm from RNS, which improves the performance of such a representation. Section V presents the performance analysis of the methods specified in Sections III and IV. The conclusions and future work are discussed in the last Section VI.

II. Related Works

Data dispersal is an important algorithmic primitive which is the basis of many methods connected with ensuring reliability and security in various spheres of distributed technologies. Secret sharing schemes, erasure codes, and error correction codes can be referred to such methods. The choice of method depends on the requirements for the system.

An important place in the practice of reliable storage and transmission of information is the replication – parallel copies storage of data on various media or transmission through various channels. Replication is used in many distributed data storage and processing systems, for example, in the Google File System [16]. The main advantage of replication is that there is no necessity to process data before saving – replication only requires time to copy of data. It is obvious to have disadvantages of this approach which are concluded in high redundancy, unacceptable in the conditions of storing and processing large amounts of data [17], on the one hand, and the necessity to apply additional data protection methods on the other hand.

Erasure codes [18] are a generalization of methods based on redundant coding intended to data recovery in case of their parts loss. The data must be encoded in such a way that it can be restored in case of several parts loss.

Error correction codes are intended to recover corrupted data. These codes are more difficult in an organization, because unlike the case of data loss, it is necessary, along with recovery, to realize the determination and localization of data errors [13], [19]. This leads to greater redundancy compared to erasure codes, as it requires a certain structure of redundant parts. Besides, correction algorithms are often very complicated and slow down the system. Nevertheless, the use of such approaches makes it possible to give a distributed system high reliability and durability of storage.

If it is necessary to ensure the confidentiality of data at the level of a distributed system, secret sharing schemes are used [20]. Threshold secret sharing schemes allow to dispersal data in such a way that they can only be recovered by combining a certain number of parts. Moreover, if the number of parts is less than a certain threshold, then, when the parts are combined, it should be impossible or difficult to restore the original data. There can be distinguished perfect secret sharing schemes [21], which differ by maximum security, which is achieved by increasing the redundancy of data representation. Such schemes are very important for distributed storage of small amounts of sensitive information (for example, encryption keys or information for authentication). However, in the case of storing large amounts of data, this approach is not effective. For confidential distributed storage of large amounts of data, computationally secure schemes are suitable [4], [22], in which cryptographic requirements are reduced, but the redundancy compared with perfect schemes is much lower.

The Rabin's scheme can serve as the basis for erasure codes [23], error correction codes [19], and computationally secure secret sharing schemes [22]. However, an important issue is the performance of this scheme. During the implementation, it is necessary to consider that the Rabin's scheme leads to a matrix-vector product over the Galois field $GF(2^n)$. Such an operation may be difficult for computers with traditional architecture.

As an alternative to the Rabin's scheme, a Residue Number System (RNS) can be used. RNS, being a number system, allows representing data in the form of non-positional numbers' sets used as a set of the dispersed part. Thanks to this feature, RNS is a universal tool for distributed data presentation. So Redundant RNS (RRNS) can be used as erasure codes. Error correction codes based on an RNS are widely used [13]. Besides, on the basis of the RNS, a number of both perfect [24], [25], [26] and computationally secure [4], [27] secret sharing schemes have been developed. The advantage of the RNS is the ability to perform operations with encoded data, which makes it a unique tool for designing homomorphic information distribution schemes [14], [15]. In addition, RNS is well used in various distributed systems, such as Multi-Cloud storage systems [4], [26], [28] and reliable storage for Internet of Things [29].

An important modification of the RNS is the Polynomial Residue Number System (PRNS), which is based on irreducible polynomials moduli over the Galois field $GF(2)$ [30], [31]. Such an approach presumes less redundancy compared to RNS over residue classes, but it is impossible to construct homomorphic schemes which will be based on it.

For the practical implementation of the methods of dispersal data coding, they must be represented in the form of corresponding algorithms. To assess the performance, it is

necessary to analyze the time spending on forward and reverse data conversion operations for each method.

In the next section, algorithms of dispersal data representation are considered. Rabin's scheme considered the main of them. As an alternative, algorithms for encoding and decoding data in RNS and polynomial RNS are considered.

III. Dispersal Coding Algorithms

A. Information Dispersal Algorithm by Rabin

Coding according to the Rabin's scheme is a process of conversion of the input data represented as a vector of parts $v = \{v_1, v_2, \dots, v_n\}^T$ in a new vector $w = \{w_1, w_2, \dots, w_n, w_{n+1}, \dots, w_{n+r}\}^T$ which contains $r \geq 0$ additional parts of data. According to the scheme, the conversion of vector v to vector w occurs by multiplication a matrix A size $(n + r) \times n$ by vector v: $Av = w$:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{(n+r),1} & a_{(n+r),2} & \cdots & a_{(n+r),n} \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{n+r} \end{bmatrix}$$

The main condition for choosing a matrix A is the invisibility of all its square submatrices with dimensions $n \times n$. Vandermonde and Cauchy's matrices possess a similar property [8], [23], which are most often used in IDA. For a more efficient implementation of erasure codes, we can use the matrix A, in which the upper submatrix is an identity. The remaining rectangular submatrix is a Cauchy or Vandermonde's matrix [32]. This approach can significantly improve the performance of encoding, but is not suitable for applications with high requirements for cryptographic security [33].

Decoding the data based on an existing subvector with n elements of vector w is multiplication the inverse matrix to the corresponding submatrix of matrix A by this subvector. Hence, both encoding and decoding in the Rabin's scheme are performed by multiplying the constant matrix by a vector.

It should be taken into account that the most efficient is the matrix-vector multiplication in the Galois field over polynomial with elements from $GF(2)$. This follows from the fact that the addition in such fields is very simple and can be executed with binary encoding as bitwise "XOR" (exclusive or). Moreover, operations in a Galois field with a modulus, equal to an irreducible polynomial, do not lead to overflow.

However, matrix-vector multiplication requires a large number of additions and multiplications modulo an irreducible polynomial. The following sections describe how to represent data in RNS, for the implementation of encoding and decoding which requires fewer operations.

B. Redundant Residue Number System

The Residue Number System is a non-positional number system, in which numbers are represented as reminders of division by pre-selected moduli from moduli set $m_1 < m_2 < \cdots < m_n < m_{n+1} < \cdots < m_{n+r}$, which must be pairwise co-prime numbers. This moduli set is divided into two parts: the first n moduli are dynamic and specify the number itself, in turn, the second remaining higher r moduli are control and produce the redundant part.

Let the data block be represented as the number S in the interval $0 \leq S < M$, where $M = m_1 \cdot m_2 \cdot \ldots \cdot m_n$ is a range of data representation in Redundant RNS. In RRNS, the number S is represented as a set of residues $S = \{s_1, s_2, \dots, s_n, s_{n+1}, \dots, s_{n+r}\}$, where $s_i = S \bmod m_i$ for all $i = 1,2, \dots, n + r$.

Each of the digits s_i in RRNS independent from others. Such representation allows to perform addition and multiplication without carry propagation between digits, which makes it possible to perform calculations in parallel and as independently as possible.

For the inverse conversion (decoding), it suffices to choose any n moduli, which underlies both the erase codes based on RRNS and the error correction codes (with the only difference is that, in the case of error correction, n moduli might be selected, which respective residues do not contain an error). The reverse conversion from RNS is based on the theory of solving systems of congruences and the Chinese Remainder Theorem (CRT). Hence, recovering the number S by n arbitrary residues $\{s_1^*, s_2^*, \dots, s_n^*\}$ (converting from RNS to weighed number system (WNS)) is calculated by the following equation:

$$S = \left| \sum_{i=1}^{n} \left| M_i^{*-1} \right|_{m_i^*} M_i^* s_i^* \right|_{M^*} \quad (1)$$

where $M^* = m_1^* \cdot m_2^* \cdot \ldots \cdot m_n^*$ is an RNS range corresponding to the obtained subset of residuals. $M_i^* = M^*/m_i^*$, $\left| M_i^{*-1} \right|_{m_i^*}$ is the multiplicative inversion M_i^* by modulus m_i^*.

The total number of operations in RNS is less than in Rabin's scheme. However, most operations here are modular, which is much more complicated than additions and multiplications in the Galois field. Also, RNS over congruence classes leads to greater redundancy compared to the Rabin's scheme due to the difference in the values of the moduli. In practice, however, the overhead equal to a few bits can be neglected.

C. Polynomial Residue Number System

The difference of polynomial RNS is that it uses polynomial Galois fields over various irreducible polynomials over the $GF(2)$ instead of congruence classes to represent digits (residues). The data chunk represented as a certain number S can be interpreted as a polynomial $S(x)$ over the Galois field modulo 2. Each binary digit of S is a coefficient in the polynomial $S(x)$ at x to a degree equal to the binary digit number. Let several different irreducible polynomials $m_1(x), m_2(x), \dots, m_n(x), m_{n+1}(x), \dots, m_{n+r}(x)$ are selected. Then polynomial $S(x)$ is represented as a set $\{s_1(x), s_2(x), \dots, s_n(x), s_{n+1}(x), \dots, s_{n+r}(x)\}$, where $s_i(x) = S(x) \bmod m_i(x)$.

Such representation has the same advantages as traditional RNS. On the basis of a polynomial RNS, erasure codes, error

correction codes, and secret sharing schemes are built by introducing redundant coding. To recover the data, you can collect any n residues and use the polynomial analogue of equation (1) for the reverse conversion.

Polynomial RNS requires the organization of calculations in the Galois field, but the redundancy of this approach is less than the redundancy of the traditional RNS and corresponds to the redundancy of the Rabin's scheme.

IV. Efficient Algorithms for Implementation of Reverse Conversion from RNS

The major problematic and complex operation in RNS is reverse conversion, which is difficult to do in parallel in computers with traditional architecture. The algorithm described by equation (1) is not efficient as it requires the calculation of the remainder of the division by a large integer M. It is important to modify the algorithm described by this equation (called CRT algorithm), in which the calculation of the remainder of dividing by M is replaced by simpler operations modulo 2^N.

The main goal of most researches of the reverse conversion is to simplify this operation as much as possible, reducing the computations along a large modulus M to simpler operations. One of the most efficient approaches is the modification of the CRT (1) proposed in [34]. This algorithm consists in the approximation of the relative position of numbers on the number line. The idea of the algorithm consists in the modification of equation (1) in such a way that it can be used to estimate the position of a number on a number line, while getting rid of complex arithmetic operations.

Let a number $S = \{s_1, s_2, \dots, s_n\}$ is given in some RNS with moduli $m_1, m_2, \dots, m_n$. To simplify the calculations, it was proposed in [34] to estimate the relative value $\tilde{S} = S/M$ based on (1), which allows to restore the value S by multiplying $\tilde{S}$ by M. For this modification, all constants $|M_i^{-1}|_{m_i} M_i$ must be divided by M. Then calculations need to be made not by modulus M, a but using the operation of discarding the integer part, which is much simpler. To simplify this algorithm, we use an approach that transforms the computations into an integer arithmetic with a fixed numbers size N by multiplying both parts of the equation by 2^N.

For a given N, we calculate the following constants:

$$k_i = \left\lfloor \frac{2^N |M_i^{-1}|_{m_i} M_i}{M} \right\rfloor$$

Using the constants k_i, e find the relative estimate of the value of S as follows:

$$\tilde{S} = \left\lfloor \frac{2^N S}{M} \right\rfloor = \left| \sum_{i=1}^{n} k_i s_i \right|_{2^N} \quad (2)$$

The sum in formula (2) requires only n multiplications and $n-1$ addition. At the same time, taking into account that operations are performed modulo 2^N, carry propagation to binary digits with numbers higher than N are ignored. According to [34] for exact recovering of value S it is necessary to choose N defined by the expression:

$$N = \lceil \log_2 M\mu \rceil - 1,$$

where $\mu = m_1 + m_2 + \cdots + m_n - n$. Further, to calculate the final value of S it is necessary to multiply the approximate value of $\tilde{S}$ by the value of M. In this case, the result of the algorithm is the high-order bits starting at bit number $N + 1$. Hence

$$S = \frac{\tilde{S}M}{2^N} \quad (3)$$

In the last expression, the division operation in software implementation is equivalent to shifting by N bits to the right. Thus, the algorithm of conversion the numbers from RNS to WNS, based on the equations (2) and (3) avoids the operation of calculating the remainder of the division into a large modulus by replacing by the multiplication, which is simpler in software implementation. Note, that in this conversion increases the size of the constants, which, however, does not significantly affect performance due to changes in the structure of calculations.

Algorithm 1 reflects the features of the software implementation of the algorithm reflected in formulas (2) and (3). The temporary variable $\tilde{S}$ accumulates the results of multiplying the residues of s_i by the constants k_i, truncated by N bit (can be implemented by limiting the computation to fixed only with N bits). Note, that in some cases (i.e. calculations with numbers with arbitrary length), it is more advantageous to perform truncation immediately after loop in line 1, rather than at each iteration. Next, in line 2 of the algorithm multiplication the number $\tilde{S}$ truncated to N lower bits by M performs. the result is placed in the variable S_{temp}. At the end, the value S is calculated as a shift N bits to the right of the value S_{temp}.

Algorithm 1 Efficient algorithm for conversion numbers from RNS to WNS

input data: number in RNS $\{s_1, s_2, \dots, s_n\}$, pre-calculated constants N, $k_1, k_2, \dots, k_n$ and M
0: **set** $\tilde{S} = 0$
1: **for** i **from** 1 **to** n **do**: $\tilde{S} = \mathbf{trunc}_N(\tilde{S} + k_i s_i)$
2: $S_{temp} = \tilde{S}M$
3: $S = \mathbf{rshift}_N(S_{temp})$
result: S

The proposed algorithm requires only addition, multiplication and shift operations and can be effectively used to recover the data represented in RNS. The performance of this algorithm in comparison with the algorithm implementing equation (1) according to the corollary of the Chinese Remainder Theorem (CRT) is presented in Fig. 1. Note, the significantly better performance of algorithm 1.

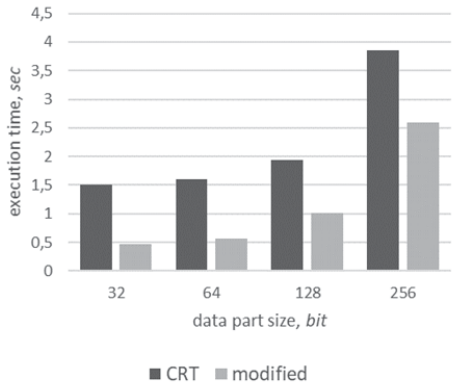

Fig. 1. Performance evaluation of RNS reverse conversion algorithms by Chines Reminder Theorem (1) and by Algorithm 1 (1 000 000 iterations each)

Considering the polynomial RNS, we note that the algorithm for reverse conversion, similar to the equations (2) and (3), can be obtained in the same way, taking into account the peculiarities of operations in polynomial arithmetic over $GF(2)$.

Note that any number S in binary arithmetic is equivalent to a polynomial $S(x)$ over a $GF(2)$, whose coefficients at degrees from 0 to the highest are binary digits of a number S. For example, $1011101_2 \leftrightarrow x^6 + x^4 + x^3 + x^2 + 1$. The main difference lies in the execution of the addition operation with polynomials, since the coefficients of the resulting polynomial are grouped by the degrees of the indeterminates, where addition modulo 2 is performed. Thus, the addition of two polynomials is equivalent to a bitwise exclusive or of their binary equivalents, which is much simpler than positional addition. Multiplication can also be done in the same way as for regular binary numbers, given only the way the addition is performed on polynomials.

Modular calculations on polynomials require the operation of calculating the remainder of division by some polynomial $m(x)$. By analogy with the binary division, it can be implemented using a series of subtractions, which are equivalent to the addition of polynomials over $GF(2)$.

During selection of the moduli set for a polynomial RNS, it is necessary to take into account that the polynomials contain a small number of low-degree non-zero coefficients and one highest-degree non-zero coefficients are most effective for implementation. For example $x^{128} + x^7 + x^2 + x + 1$ for representation of 128-bit numbers. We know in advance that there are zeros in the digits with numbers from 8 to 127, which allows us to significantly speed up the calculations [35].

V. Performance Evaluation

To provide sufficient cryptographic security, it is necessary that the dispersed parts have a sufficiently large size. Therefore, different sizes of dispersed parts are considered for this study. For cases when cryptographic properties are not crucial for the developed system, the dispersal by 32 and 64 bit was considered. More sensitive data must be dispersed into parts with a large size, so the bit sizes from 96 to 1024 bits were analyzed.

Calculations in RNS and in Galois fields with arbitrary precision numbers are used for simulations, with number-theoretic algorithms NTL by Victor Shoup [36], which effectively implements a number of specific operations, including operations in Galois fields, modular arithmetic, matrix calculus. An important advantage when running algorithms on Unix-like systems is the ability to combine NTL with the built-in library GMP for multi-digit calculations, which allows achieving higher performance.

All experiments are performed on a computer with Intel Xeon CPU E5-2690 v4 with 2.60 GHz and Linux Ubuntu Server 18.04.1 in a sequential mode. The virtual machine for experiments is allocated in the Data Center of the North Caucasus Federal University. The programs are implemented using the NTL 11.3.2 library and GMP with GNU compiler version 7.3.

Three cases are evaluated dispersing data into 4, 6 and 8 parts. For the Rabin scheme, in order to evaluate the procedure of data dispensation, we simulate the operation of matrix multiplication by a vector in the Galois field over an irreducible polynomial. The length corresponds to the size of the parts that should be dispersed. Hence, it corresponds to the sizes of the constant matrix and the input data vector elements. The multiplication is implemented in the NTL library.

For RNS, moduli sets of 4, 6, and 8 co-prime numbers were selected where each modulus was larger than the size of the part of the dispersed data. In such a case, a sufficient range is guaranteed to present the source data. In turn, for PRNS, we choose 4, 6, and 8 irreducible polynomials, which are used as the corresponding moduli sets.

The degree of each polynomial is determined by the size of the part be dispersed.

It should be noted that minimal irreducible polynomials for a specific degree are used for simulations. It means that the binary representation of such polynomials contains the minimum number of units located in the least bits, not counting the element with the highest degree. For such polynomials, there are optimized algorithms for calculating the remainder of division and optimized approaches to performing arithmetic operations.

Fig. 2 shows the results of modeling serial versions of data dispersal and recovery algorithms based on the approaches described above. Each operation for averaging the results was performed 1,000,000 times. The best performance among all considered operations for sizes until to 256 bits is achieved by the using of developed Algorithm 1 for reverse converting numbers from RNS to the weighted number system. Other operations, namely the forward conversion to RNS, the matrix-vector product for Rabin conversions and the reverse conversion from PRNS, using the proposed algorithm, are performed approximately equal time.

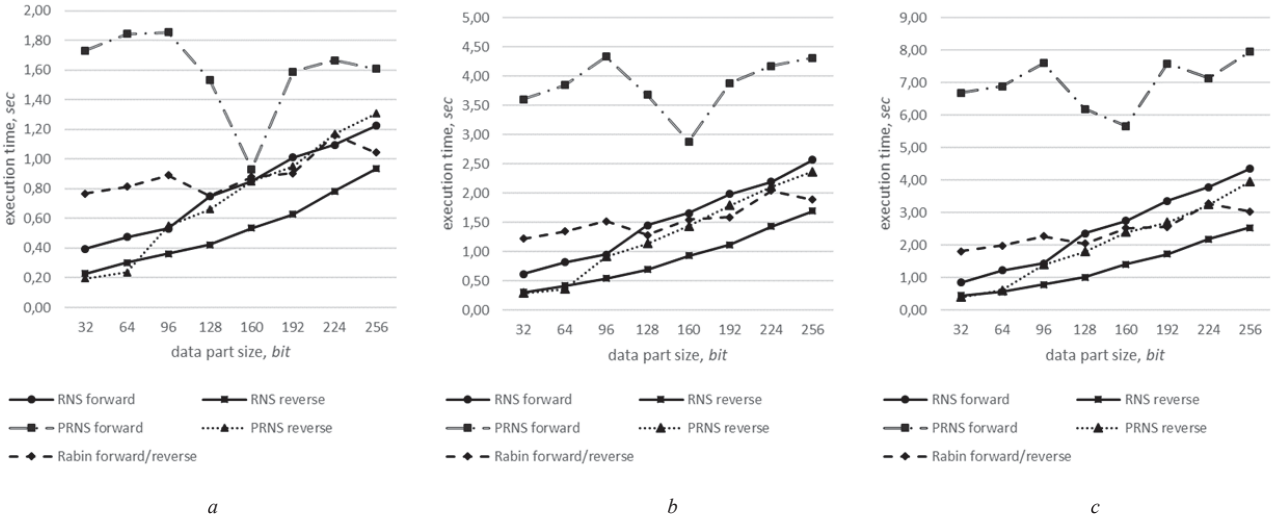

Fig. 2. Comparison of the performance of different algorithms for coding and decoding for dispersal of data: *a* – into 4 parts, *b* – into 6 parts, *c* – into 8 parts

Note that in practice the application of each type of conversions occurs in pairs (forward and reverse) depending on the chosen data representation method. A visual comparison of the total time of the forward and reverse data conversions is illustrated in Fig. 3. The graphs demonstrate the performance (the sum of the time of the forward and reverse conversions) depending on the chosen method of data presentation for dispersing them into 8 parts.

Fig. 3 shows that the forward and reverse conversion of data to RNS requires less CPU time compared to IDA and PRNS.

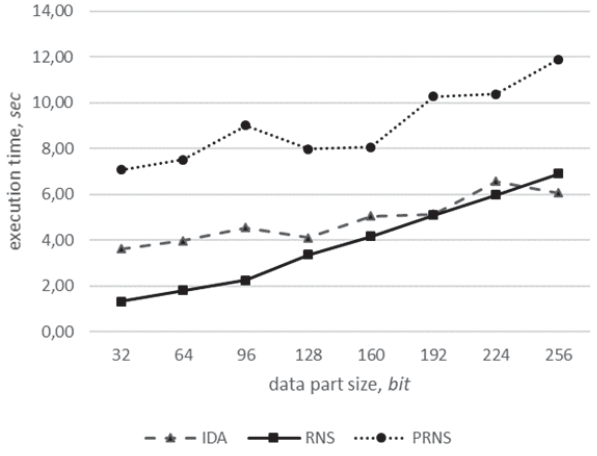

Fig. 3. Performance evaluation of both forward and reverse conversions of the different approaches to the data dispersal

In the graphs at Fig. 2 and Fig. 3, there are noticeable differences in the speed of operations for algorithms based on the polynomial data representation (the Rabin's scheme and PRNS). These algorithms are very sensitive to the choice of irreducible polynomials.

For example, minimal irreducible polynomials which are chosen as PRNS moduli sets for 4 moduli with 96-bit length are:

$$m_1(x) = x^{96} + x^6 + x^5 + x^3 + x^2 + x + 1,$$
$$m_2(x) = x^{96} + x^7 + x^6 + x^4 + x^3 + x^2 + 1,$$
$$m_3(x) = x^{96} + x^8 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1,$$
$$m_4(x) = x^{96} + x^8 + x^7 + x^6 + x^4 + x^3 + 1.$$

On the other hand, for 160-bit size, the moduli set are selected as follows:

$$m_1(x) = x^{160} + x^5 + x^3 + x^2 + 1,$$
$$m_2(x) = x^{160} + x^5 + x^4 + x^3 + 1,$$
$$m_3(x) = x^{160} + x^7 + x^3 + x + 1,$$
$$m_4(x) = x^{160} + x^8 + x^7 + x^5 + x^4 + x^3 + x^2 + x + 1.$$

From the polynomial record, we see that the selected polynomials of degree 160 require less significant digits (except for the last one). Experiments have shown that this fact significantly affects the performance of the algorithms used. Conversion in PRNS with 160-bit moduli is faster than conversion with 96-bit moduli (Fig. 2). Such an effect can be seen in detail in Fig. 4. It shows the results of modeling separately by the operations of calculating the remainder of division by polynomials (the smallest and largest for each of the digits), the regular reminder of division by a number, and the multiplication of a vector by a vector.

Fig. 4 also shows the potential for parallelization of forward conversion algorithms using various algorithms for the considered bit lengths. In Rabin's scheme, the matrix-vector multiplication operation can be divided into several parallel scalar multiplications of vectors. On the other hand, the conversion to RNS and to PRNS can be implemented in

parallel for each modulus. However, a parallel organization of calculations for these algorithms requires further study.

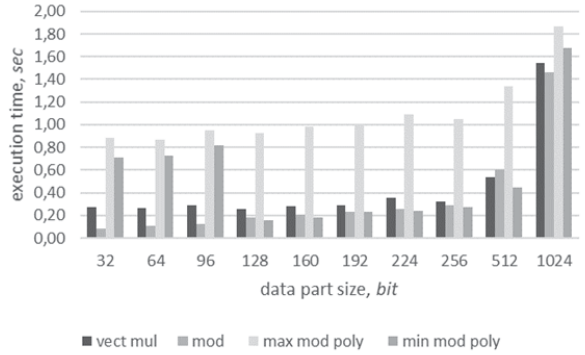

Fig. 4. Performance evaluation of single operations of vectors multiplication, residue calculation (*mod*) and polynomial residue calculations (*mod poly* for minimal and maximal polynomials used for PRNS moduli sets).

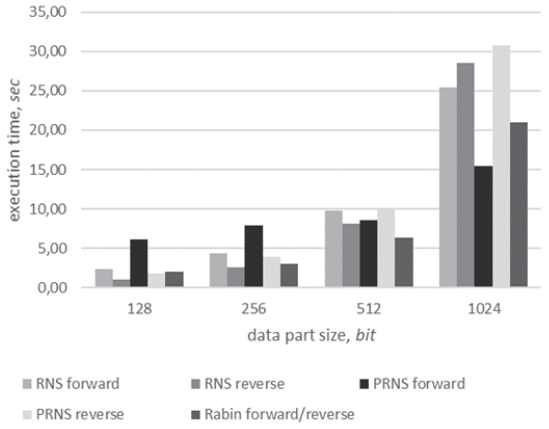

Fig. 5. Performance evaluation of dispersal into 8 parts for high bit lengths of parts

Besides, for numbers with a larger bit length, the proposed methods show lower performance. This is primarily due to the fact that for the input numbers with the large size the value N required for an estimation of the number of digits in the calculations by equations (2) and (3) for RNS and PRNS increases significantly. This fact is demonstrated in Fig. 5.

On the other hand, forward conversion to PRNS slows down slightly, showing better performance for very large bit sizes relative to other methods.

TABLE I. PROPERTIES OF EVALUATED METHODS (FOR THE PART SIZES BELOW 256 BIT)

Method	Algorithms	Performance	Redundancy
RNS	forward/encoding	medium	medium
	reverse/decoding	height	
PRNS	forward/encoding	low	low
	reverse/decoding	medium	
Rabin's IDA	forward/encoding	medium	low
	reverse/decoding	medium	

The comparison of the properties of the analyzed information dispersal methods is presented in Table I. We noted that RNS has medium redundancy since using this method leads to a small redundancy required to represent data as residues from dividing by numbers not equal to 2^t. The other two methods considered is devoid of this disadvantage.

The simulation showed that the performance of the Rabin's method and the conversion to PRNS depend significantly on the choice of irreducible polynomials on the basis of which the calculations are made. Fig. 2-5 present simulation results that support this conclusion. On the other hand, it can be noted that the methods of forward and reverse conversion to RNS for numbers up to 256 bits are devoid of such a disadvantage. Hence, they are faster, without depending on the number of parts into which the data are dispersed.

VI. CONCLUSION AND FUTHER WORK

Information dispersal methods are an important instrument for organizing modern distributed systems. Such methods underlie reliable and secure data storage, transmission and processing. However, in the face of increasing data volumes, an important issue is the performance of information distribution and reconstruction algorithms.

In this paper, we analyze and conduct a comprehensive performance evaluation study of information distribution algorithms based on three methods: Rabin scheme, RNS, and polynomial RNS. Rabin's IDA is the main approach used in practice. We demonstrate that Rabin's IDA performance strongly depends on the selection of an irreducible polynomial for the Galois field.

Our experimental analysis shows that despite the effective implementation of the reverse conversion from PRNS, this representation is inefficient due to the low performance of the forward conversion algorithms to PRNS.

On the other hand, we demonstrate that RNS has several important advantages, including the possibility to perform parallel arithmetic operations with encoded data. The scientific novelty of the conducted research consists in the proposed algorithm of reverse conversion from RNS to the weighted number system. Using the proposed algorithm, we achieve a high data recovery rate.

We detect main drawbacks of the reverse conversion of numbers from RNS with the high bit length of the operands. The length is much higher than the bit length of the data to be dispersed.

However, further study is required to assess their actual efficiency, effectiveness, and scalability on a different distributed infrastructures. This will be subject of future work requiring a better understanding of recovery algorithms based on the RNS coding with higher processing speed, which would be suitable for the dispersal of data into parts of greater bit lengths.

Besides, an important issue to be addressed is the parallel processing for each of the modules. It allows developing

approaches for implementing complex distributed systems that effectively use their available resources.

ACKNOWLEDGMENT

The work is partially supported by Russian Foundation for Basic Research (RFBR), grants numbers 18-07-00109, 19-07-00130, 19-07-00856, 18-07-01224, and Russian Federation President Grants MK-6294.2018.9.

REFERENCES

[1] J. Gantz and D. Reinsel, "The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East," 2012.

[2] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, Sep. 2010, pp. 4539–4551.

[3] J. L. Gonzalez, J. C. Perez, V. J. Sosa-Sosa, L. M. Sanchez, and B. Bergua, "SkyCDS: A resilient content delivery service based on diversified cloud storage," *Simul. Model. Pract. Theory*, vol. 54, May 2015, pp. 64–85.

[4] M. Deryabin, N. Chervyakov, A. Tchernykh, M. Babenko, N. Kucherov, V. Miranda-Lopez, and A. Avetisyan, "Secure Verifiable Secret Short Sharing Scheme for Multi-Cloud Storage," in *2018 International Conference on High Performance Computing & Simulation (HPCS)*, 2018, pp. 700–706.

[5] P. Junghanns, B. Fabian, and T. Ermakova, "Engineering of secure multi-cloud storage," *Comput. Ind.*, vol. 83, Dec. 2016, pp. 108–120.

[6] S. Sen, J. A. Clark, and J. E. Tapiador, "Security threats in mobile ad hoc networks," *Secur. Self-Organizing Networks MANET, WSN, WMN, VANET*, 2010, pp. 127–147.

[7] M. Conti and S. Giordano, "Mobile ad hoc networking: milestones, challenges, and new research directions," *IEEE Commun. Mag.*, vol. 52, no. 1, Jan. 2014, pp. 85–96.

[8] M. O. Rabin and M. O., "Efficient dispersal of information for security, load balancing, and fault tolerance," J. ACM, vol. 36, no. 2, Apr. 1989, pp. 335–348.

[9] S. C. Gladwin, M. M. England, Z. J. Mark, V. T. Thornton, J. J. Mullin, and S. K. Modi, "Billing system for information dispersal system." US Patent 7574570, 2009.

[10] S. C. Gladwin, G. Dhuse, V. Thornton, M. Motwani, I. Volvovski, W. Leggette, J. Bellanca, S. Toledano, L. Foster, Z. Mark, and others, "Virtualized data storage vaults on a dispersed data storage network." US Patent 7904475, 2011.

[11] A. G. Dimakis, K. Ramchandran, Yunnan Wu, and Changho Suh, "A Survey on Network Codes for Distributed Storage," *Proc. IEEE*, vol. 99, no. 3, Mar. 2011, pp. 476–489.

[12] N. S. Szabó and R. L. Tanaka, *Residue arithmetic and its applications to computer technology*. 1967.

[13] V. T. Goh and M. U. Siddiqi, "Multiple error detection and correction based on redundant residue number systems," *IEEE Trans. Commun.*, vol. 56, no. 3, Mar. 2008, pp. 325–330.

[14] M. Babenko, N. Chervyakov, A. Tchernykh, N. Kucherov, M. Deryabin, G. Radchenko, P. O. A. Navaux, and V. Svyatkin, "Security analysis of homomorphic encryption scheme for cloud computing: Known-plaintext attack," in *2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 2018, pp. 270–274.

[15] M. Gomathisankaran, A. Tyagi, and K. Namuduri, "HORNS: A homomorphic encryption scheme for Cloud Computing using Residue Number System," in *2011 45th Annual Conference on Information Sciences and Systems*, 2011, pp. 1–5.

[16] S. Ghemawat, H. Gobioff, S.-T. Leung, "The Google file system," in *Proceedings of the nineteenth ACM symposium on Operating systems principles - SOSP '03*, 2003, vol. 37, no. 5, p. 29-43.

[17] R. Nachiappan, B. Javadi, R. N. Calheiros, and K. M. Matawie, "Cloud storage reliability for Big Data applications: A state of the art survey," *J. Netw. Comput. Appl.*, vol. 97, Nov. 2017, pp. 35–47.

[18] J. Li and B. Li, "Erasure coding for cloud storage systems: A survey," *Tsinghua Sci. Technol.*, vol. 18, no. 3, Jun. 2013, pp. 259–272.

[19] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields," *J. Soc. Ind. Appl. Math.*, vol. 8, no. 2, Jun. 1960, pp. 300–304.

[20] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, Nov. 1979, pp. 612–613.

[21] E. Brickell and D. Davenport, "On the classification of ideal secret sharing schemes," *J. Cryptol.*, vol. 4, no. 2, 1991, pp. 123–134.

[22] H. Krawczyk, "Secret Sharing Made Short," in *Advances in Cryptology — CRYPTO' 93*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 136–146.

[23] J. S. Plank, "Erasure codes for storage systems: A brief primer," *Login: The USENIX Magzine*, vol. 38, no. 6, 2013, pp. 44–50.

[24] C. Asmuth and J. Bloom, "A modular approach to key safeguarding," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, Mar. 1983, pp. 208–210.

[25] M. Quisquater, B. Preneel, and J. Vandewalle, "On the security of the threshold scheme based on the chinese remainder theorem," in *Lecture Notes in Computer Science*, vol. 2274, 2002, pp. 199–210.

[26] A. Tchernykh, V. Miranda-López, M. Babenko, F. Armenta-Cano, G. Radchenko, A. Y. Drozdov, and A. Avetisyan, "Performance evaluation of secret sharing schemes with data recovery in secured and reliable heterogeneous multi-cloud storage," *Cluster Comput.*, Jan. 2019, pp. 1–13.

[27] A. Tchernykh, M. Babenko, N. Chervyakov, V. Miranda-López, V. Kuchukov, J. M. Cortés-Mendoza, M. Deryabin, N. Kucherov, G. Radchenko, and A. Avetisyan, "AC-RRNS: Anti-collusion secured data sharing scheme for cloud storage," *Int. J. Approx. Reason.*, vol. 102, Nov. 2018, pp. 60–73.

[28] A. Tchernykh, M. Babenko, V. Miranda-Lopez, A. Y. Drozdov, and A. Avetisyan, "WA-RRNS: Reliable Data Storage System Based on Multi-cloud," in *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2018, pp. 666–673.

[29] N. Chervyakov, M. Babenko, A. Tchernykh, N. Kucherov, V. Miranda-López, and J. M. Cortés-Mendoza, "AR-RRNS: Configurable reliable distributed data storage systems for Internet of Things to ensure security," *Futur. Gener. Comput. Syst.*, vol. 92, pp. 1080–1092, Mar. 2019.

[30] C. Ding, D. Pei, and A. Salomaa, *Chinese remainder theorem: applications in computing, coding, cryptography*, World Scientific, 1996.

[31] J. Chu and M. Benaissa, "Error detecting AES using polynomial residue number systems," *Microprocess. Microsyst.*, vol. 37, no. 2, Mar. 2013, pp. 228–234.

[32] L. X. James S. Plank, "Optimizing Cauchy Reed-Solomon Codes for Fault-Tolerant Network Storage Applications," in *Fifth IEEE International Symposium on Network Computing and Applications (NCA'06)*, 2006, pp. 173–180.

[33] Alvarez, Gonzalo, and Shujun Li. "Some basic cryptographic requirements for chaos-based cryptosystems." *International journal of bifurcation and chaos*, vol. 16, no. 8, 2006, pp. 2129-2151.

[34] N. I. Chervyakov, A. S. Molahosseini, P. A. Lyakhov, M. G. Babenko, and M. A. Deryabin, "Residue-to-binary conversion for general moduli sets based on approximate Chinese remainder theorem," *Int. J. Comput. Math.*, vol. 94, no. 9, Sep. 2017, pp. 1833–1849.

[35] V. Shoup, "Efficient computation of minimal polynomials in algebraic extensions of finite fields," in *Proceedings of the 1999 international symposium on Symbolic and algebraic computation - ISSAC '99*, 1999, pp. 53–58.

[36] NTL: A library for doing number theory by V. Shoup, Web: http://www. shoup. net/ntl/