High speed method of conversion numbers from residue number system to positional notation

Vladislav Lutsenko
North-Caucasus Federal University
Stavropol, Russia
https://orcid.org/0000-0003-4648-8286

Mikhail Babenko
North-Caucasus Federal University
Stavropol, Russia
https://orcid.org/0000-0001-7066-0061

***Abstract*—The Residue Number System is a widely used non-positional number system. Residue Number System can be effectively used in applications and systems with a predominant share of addition, subtraction and multiplication operations, due to the parallel bit-by-bit execution of operations and the absence of inter-bit carries. The reverse conversion of a number from Residue Number System to positional notation requires the use of special algorithms. The main focus of this article lies in introducing the new conversion method, which incorporates Chinese Remainder Theorem, Akushsky Core Function and rank of number. The step-by-step procedure of the conversion process is detailed, accompanied by numerical examples to demonstrate the effectiveness of the proposed method. The proof of the relationship between the ranks of positional characteristics using the Chinese Remainder Theorem is presented. Through careful analysis and comparison with existing transformation methods, it is concluded that the presented approach takes on average 8% less time than the Approximate Method.**

***Index Terms*—residue number system, Chinese remainder theorem, approximate method, Akushsky core functions, non-modular operations**

I. INTRODUCTION

In today's world, where computational systems play an increasingly significant role in various fields of activity, the question of efficiently converting numbers between different numeral systems becomes particularly relevant. One such system is the Residue Number System (RNS), which provides unique capabilities for handling large numbers through parallel computations [1]. RNS is applied in the following areas: blockchain [2], homomorphic encryption [3], digital signal and image processing [4], neural networks [5].

However, there are cases where it is necessary to translate numbers from RNS to positional notation, which is commonly used in most computational devices [6]. Efficient methods for converting numbers from RNS to positional notation must be developed.

In [7] the authors introduced a technique based on the Chinese Remainder Theorem (CRT) and employed optimized modular arithmetic operations to achieve faster conversions. The algorithm was evaluated on a variety of RNS moduli sets and demonstrated significant improvements in conversion time compared to previously known methods.

The research was supported by the Russian Science Foundation Grant No. 19-71-10033, https://rscf.ru/en/project/19-71-10033/.

Chervyakov et al. [8] focused on developing a hybrid conversion method that combines RNS and binary arithmetic to achieve more efficient conversions. The proposed method utilized operand scanning techniques to identify patterns in the RNS representation and optimize the conversion process. The authors demonstrated that their hybrid approach outperforms conventional conversion methods in terms of both speed and hardware resource utilization.

The article [9] focuses on hardware acceleration for RNS-to-decimal conversion using Field-Programmable Gate Arrays (FPGAs). The authors designed a specialized hardware accelerator capable of handling large-scale RNS numbers and converting them efficiently to decimal format. The proposed FPGA-based implementation demonstrated substantial speedup compared to software-based conversion methods, making it suitable for real-time applications.

In [10], [11] proposed energy efficient conversion algorithms which minimizes the energy consumption in the process of number conversion and number sign determination. By optimizing the use of resources and considering the power constraints of the base equipment, the proposed methods provide significant energy savings compared to conventional conversion methods.

Advances in this area have paved the way for improved performance, reduced power consumption and increased fault tolerance, making RNS a more attractive option in various domains [12]. However, further research is still warranted to explore new techniques and optimizations that can further enhance the conversion process and maximize the potential of RNS in modern computing systems. This paper researches methods of converting numbers from RNS to the positional notation. The main methods are the CRT based method, the Interval Method, the Mixed Radix Conversion (MRC) method, the Diagonal Function (DF) method and the Approximate Method.

The purpose of this paper is to present a high-speed method for converting numbers from RNS to positional notation based on the use of Akushsky core function and number rank.

The paper is organised as follows. In Section 2, we give a brief overview of the Residue Number System. In Section 3, various techniques for converting numbers from RNS to positional notation are described. Section 4 deals with performance evaluation. Finally, Section 5 concludes with some final thoughts and considerations, including possible directions for future research.

II. Residue Number System

In RNS, numbers are expressed as sets of residues obtained by performing modular arithmetic operations on those numbers with respect to a set of coprime moduli. The use of coprime moduli ensures that there is no overlap or interference between the residues, allowing for parallel computation of operations on individual residues [12].

The numerical representation in RNS utilises the Chinese Remainder Theorem. Let $\{p_1, p_2, ..., p_n\}$ be mutually prime moduli, and $P = p_1 \cdot p_2 \cdot ... \cdot p_n$ be their product. For each number x, there exists a set of remainders $x_1, x_2, ..., x_n$, where $0 \le x_i < p_i$, and these remainders form the RNS representation of X. Put differently, X exhibits congruence with the residues x_i modulo p_i.

Mathematically, this can be expressed as:

$$x_i \equiv X \pmod{p_i} \tag{1}$$

Thus, the number X is written in the RNS in the following form:

$$X = (x_1, x_2, \ldots, x_n). \tag{2}$$

The computations for the reductions x_i can be derived through the application of the following equation:

$$x_i = X - \left\lceil \frac{X}{p_i} \right\rceil \cdot p_i. \tag{3}$$

To perform operations on numbers in RNS, such as addition and multiplication, operations are carried out independently on the remainders of each modulo. For example, calculations in RNS are performed according to equation:

$$A * B = (a_1 * b_1, a_1 * b_1, ..., a_n * b_n).$$

Here, the symbol $*$ represents arithmetic operations, encompassing addition $(+)$, subtraction $(-)$, or multiplication $(\cdot)$. Note that each modulo within the RNS is coprime with every other modulo, satisfying the condition: $gcd(p_i, p_j) = 1$, where $i \ne j$.

III. Methods for Conversion Numbers from RNS to Positional Notation

A. Chinese Remainder Theorem

If the number X is given as residues $(x_1, x_2, ..., x_n)$ from division by moduli $\{p_1, p_2, ..., p_n\}$, the number X can be obtained from the equation based on the CRT [9]:

$$\begin{aligned} X = \left| \sum_{i=1}^{n} P_i \cdot x_i \cdot \left| P_i^{-1} \right|_{p_i} \right|_P = \\ = \sum_{i=1}^{n} P_i \cdot x_i \cdot \left| P_i^{-1} \right|_{p_i} - r(X) \cdot P. \end{aligned} \tag{4}$$

where P is the dynamic range, $P_i = \frac{P}{p_i}$, $\left| P_i^{-1} \right|_{p_i}$ is the multiplicative inversion of P_i modulo p_i, and the operator $|X|_{p_i}$ denotes the remainder of division X by p_i, that is $X \mod p_i$ and $r(X)$ is the rank of the number indicating how many times the range value must be subtracted from the resulting number to bring it back into the range. Let us consider the process of number reconstruction as an example.

Example 1. Given a system of bases $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, p_5 = 11$ the volume of the dynamic range $P = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310$. Convert the number $X = (1, 2, 1, 4, 7)$ to a positional system.

For this purpose, find the values of P_i:

$$P_1 = \frac{P}{p_1} = 1155, P_2 = \frac{P}{p_2} = 770, P_3 = \frac{P}{p_3} = 462,$$

$$P_4 = \frac{P}{p_4} = 330, P_5 = \frac{P}{p_5} = 210.$$

Subsequently, our focus turns to the computation of multiplicative inversion, a process entailing the determination of α such that $\alpha \cdot P_i \equiv 1 \mod p_i$. Thus:

$$\left| P_1^{-1} \right|_{p_1} = 1, \left| P_2^{-1} \right|_{p_2} = 2, \left| P_3^{-1} \right|_{p_3} = 3,$$

$$\left| P_4^{-1} \right|_{p_4} = 1, \left| P_5^{-1} \right|_{p_5} = 1.$$

With these values, we can calculate the value of the number X, according to the (4):

$$X = |8411|_{2310} = 1481.$$

B. Approximate Method Based on CRT

In [10], [13] a fractional, approximate representation of numbers based on CRT is proposed. Let us divide (4) by P and obtain

$$\frac{X}{P} = \left| \sum_{i=1}^{n} x_i \cdot \frac{\left| P_i^{-1} \right|_{p_i}}{p_i} \right|_1 = \left| \sum_{i=1}^{n} x_i \cdot \ k_i \right|_1. \tag{5}$$

where $k_i = \frac{\left| P_i^{-1} \right|_{p_i}}{p_i}$ constants of the chosen system, and the (5) gives a result within the interval $[0, 1)$. In this context, the process of determining the remainder with a larger modulo is replaced by simply discarding the integer part, a simple operation to implement. To get the exact value, the fractional part is multiplied by P. Consider a similar example.

Example 2. Given a system of bases $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, p_5 = 11$ and the number $X = (1, 2, 1, 4, 7)$. Find the constants k_i:

$$k_1 = \frac{1}{2}, k_2 = \frac{2}{3}, k_3 = \frac{3}{5}, k_4 = \frac{1}{7}, k_5 = \frac{1}{11}.$$

Then by (5) it is easy to find:

$$\frac{X}{P} = \left| 1 \cdot \frac{1}{2} + 2 \cdot \frac{2}{3} + 1 \cdot \frac{3}{5} + 4 \cdot \frac{1}{7} + 7 \cdot \frac{1}{11} \right|_1 =$$

$$= \left| 1 \frac{52}{105} \right|_1 = \frac{52}{105},$$

Hence

$$X = \frac{52}{105} \cdot 2310 = 1481.$$

Obviously, these calculations are simpler than in the CRT-based method, but in hardware calculations the fractional coefficients k_i can rarely be represented as finite fractions, so there is a question of rounding accuracy. To perform approximate calculations the fractional coefficients k_i are multiplied by 2^N, where N signifies the count of binary digits located beyond the decimal point, which provides the required level of calculation accuracy, each resulting number is rounded up to the next integer and then all calculations are performed modulo 2^N.

C. Mixed Radix Conversion Method

The Mixed Radix Conversion technique involves systematically translating a numerical representation from RNS to Weighted Number System (WNS) through a sequential process [14].This method involves subtracting moduli and multiplying by the multiplicative inversion of a modulo. In WNS the translated number has the following form:

$$X = d_1 + x_1 p_1 + d_2 p_1 p_2 + ... + d_n p_1 p_2 ... p_{n-1}, \tag{6}$$

where $0 \leq d_i \leq (p_{i+1} - 1)$. The parameters d_i are known as WNS digits.

The WNS digits can be obtained from the ratios:

$$\begin{aligned} & d_1 = X - X_1 \cdot p_1, X_1 = \left\lceil \frac{X}{p_1} \right\rceil \\ & d_2 = X_1 - X_2 \cdot p_2, X_2 = \left\lceil \frac{X_1}{p_2} \right\rceil \\ & \vdots \\ & d_n = X_{n-1} - X_n \cdot p_n, X_n = \left\lceil \frac{X_{n-1}}{p_n} \right\rceil . \end{aligned} \tag{7}$$

The conversion carried out according to the algorithm (7) contains $2(n-1)$ only residual arithmetic operations of subtraction and division without remainder, where is the number of moduli of the system. Some modification of the considered algorithm can be proposed in the sense that the division operation is replaced by the multiplication operation. For this purpose we pre-calculate constants τ_{kj} that satisfy the condition

$$\tau_{kj} p_k \equiv 1 \pmod{p_j}, (1 \leq k < j \leq n). \tag{8}$$

It is noteworthy to highlight that the constants τ_{kj} are entirely dictated by the selected system of bases, rendering them computable beforehand and amenable to storage in a designated table.

If the constants τ_{kj} are calculated, the calculation of the digits d_i WNS by the algorithm (6) can be rewritten in the form:

$$\begin{aligned} & d_1 \equiv x_1 \pmod{p_1}, \\ & d_2 \equiv (x_2 - d_1)\tau_{12} \pmod{p_2}, \\ & d_3 \equiv ((x_3 - d_1)\tau_{13} - d_2)\tau_{23} \pmod{p_3}, \\ & \quad \vdots \\ & d_n \equiv ((...(x_n - d_1)\tau_{1n} - ...d_{n-1})\tau_{n-1n} \pmod{p_n}). \end{aligned} \tag{9}$$

The constants τ_{kj} are multiplication inverses for the numbers p_k modulo p_j.

Consider the algorithm (9) with an example.

Example 3. Let a system of bases $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, p_5 = 11$ be given. The volume of the dynamic range $P = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310$. Convert the number $X = (1, 2, 1, 4, 7)$ to WNS.

We first find the constants τ_{kj}. For convenience, we write the constants τ_{kj} as a matrix $k \times j$:

$$\begin{pmatrix} 0 & 2 & 3 & 4 & 6 \\ 0 & 0 & 2 & 5 & 4 \\ 0 & 0 & 0 & 3 & 9 \\ 0 & 0 & 0 & 0 & 8 \end{pmatrix}$$

Now run the algorithm (9) and write the results in Tab. I. Thus,

$$X = d_5 p_1 p_2 p_3 p_4 + d_4 p_1 p_2 p_3 + d_3 p_1 p_2 + d_2 p_1 + d_1 =$$

$$= 7 \cdot 2 \cdot 3 \cdot 5 \cdot 7 + 0 \cdot 2 \cdot 3 \cdot 5 + 1 \cdot 2 \cdot 3 + 2 \cdot 2 + 1 = 1481.$$

D. Interval Method

Sufficiently effective methods of converting numbers from RNS to positional representation is the interval method, based on the interval characteristics of numbers. One of these characteristics is the interval number [15].

Let RNS is given by a system of bases $p_1, p_2, ..., p_n$, with the volume of the range $P = \prod_{i=1}^{n} p_i$. Choose a splitting modulo p_i and split the given range into intervals by dividing P by the modulo p_i. Then the number of intervals is $m = P_i = \frac{P}{p_i}$, and the length of an interval is determined by the modulo value. As a result, the value of any number X given in RNS on the chosen bases can be determined by the interval number:

$$l_X = \left\lceil \frac{X}{p_i} \right\rceil, \tag{10}$$

which contains the number X and by digit x_i of the number X in the RNS modulo p_i, i.e.

$$X = p_i l_X + x_i. \tag{11}$$

Since $gcd(p_i, P_i) = 1$, by Euler's theorem:

$$P_i^{\varphi(p_i)} \equiv 1 \pmod{p_i}, \tag{12}$$

where $\varphi(p_i)$ is an Euler function. If p_i is a prime number, then $\varphi(p_i) = p_i - 1$.

Substituting (12) into (4) the number X can be written as

$$X = \left| \sum_{i=1}^{n} P_i^{\varphi(p_i)} x_i \right|_P . \tag{13}$$

To determine the interval number l_X, substitute (13) into (10):

$$l_X = \left\lceil \frac{\sum_{i=1}^{n} P_i^{\varphi(p_i)} x_i - r(X) P}{p_i} \right\rceil . \tag{14}$$

TABLE I: Algorithm of the MRC method

Actions	Moduli					Digits
	$p_1=2$	$p_2=3$	$p_3=5$	$p_4=7$	$p_5=11$	
$X-d_1$	1 1	2 1	1 1	4 1	7 1	$d_1=1$
$(X-d_1)\cdot\tau_{1j}$	0	1 2	0 3	3 4	6 6	
X_1-d_2		2 2	0 2	5 2	3 2	$d_2=2$
$(X_1-d_2)\cdot\tau_{2j}$		0	3 2	3 5	1 4	
X_2-d_3			1 1	1 1	4 1	$d_3=1$
$(X_2-d_3)\cdot\tau_{3j}$			0	0 3	3 9	
X_3-d_4				0 0	5 0	$d_4=0$
$(X_3-d_4)\cdot\tau_{4j}$				0	5 8	
X_4					7	$d_5=7$

Since p_i is a divisor of the numbers $P_j^{\varphi(p_j)} (i \neq j)$, $P_i^{\varphi(p_i)}-1$, P then

$$l_X = l_{X_1}x_1 + l_{X_2}x_2 + ... + l_{X_n}x_n - r_X P, \qquad (15)$$

where $l_{X_j} = \frac{P_j^{\varphi(p_j)}}{p_i}, i \neq j$ and $l_{X_j} = \frac{P_i^{\varphi(p_i)}-1}{p_i}$ are constant coefficients defined by the base system.

Thus we have,

$$l_X = \left| \sum_{i=1}^{n} \left| l_{X_i} x_i \right|_{P_i}^{+} \right|_{P_i}^{+}. \qquad (16)$$

Substituting (16) into (11), we obtain a positional notation of the number X:

$$X = \left(\left| \sum_{i=1}^{n} \left| l_{X_i} x_i \right|_{P_i}^{+} \right|_{P_i}^{+} \right) p_i + x_i. \qquad (17)$$

It may be noted here that it is more appropriate to choose the largest modulo in the system as the split modulo. In this case, modular operations are performed with a smaller modulo value.

We will illustrate this method with an example.

Example 4. Let a system of bases $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, p_5 = 11$ be given. Convert the number $X = (1, 2, 1, 4, 7)$ to a positional notation. Let us choose $p_5 = 11$ as the splitting modulo, then $P_5 = \frac{P}{p_5} = 210$, the interval number

$$l_X = \left| \sum_{i=1}^{5} \left| l_{X_i} x_i \right|_{210}^{+} \right|_{210}^{+}.$$

and the number $X = p_5 l_X + x_5$. Define l_{X_i}. Since $\varphi(p_1) = 2-1 = 1, \varphi(p_2) = 3-1 = 2, \varphi(p_3) = 5-1 = 4, \varphi(p_4) = 7-1 = 6, \varphi(p_4) = 11-1 = 10$, then

$$l_{X_1} = \left| \frac{1155}{11} \right|_{210}^{+} = 105, l_{X_2} = \left| \frac{770^2}{11} \right|_{210}^{+} = 140,$$

$$l_{X_3} = \left| \frac{462^4}{11} \right|_{210}^{+} = 126, l_{X_4} = \left| \frac{330^6}{11} \right|_{210}^{+} = 30,$$

$$l_{X_5} = \left| \frac{210^{10}-1}{11} \right|_{210}^{+} = 19.$$

Then $l_X = |764|_{210}^{+} = 134$.
Thus, $X = 134 \cdot 11 + 7 = 1481$.

E. Diagonal Function

There is another way of reconstructing the numbers in the literature [16], [17]. For RNS $\{p_1, p_2, ..., p_n\}$ define the Sum of Quotients (SQ) parameter as

$$SQ = P_1 + P_2 + ... + P_n, \qquad (18)$$

and the constants

$$k_i = \left| -p_i^{-1} \right|_{SQ}. \qquad (19)$$

The diagonal function for a given number $X = (x_1, x_2, ..., x_n)$ is defined as

$$D(X) = \left| x_1 k_1 + x_2 k_2 + ... + x_n k_n \right|_{SQ}. \qquad (20)$$

If (4) is multiplied by $\frac{SQ}{P}$, we get the scaled value of X:

$$\frac{X \cdot SQ}{P} = \left| \sum_{i=1}^{n} SQ \cdot \frac{x_i}{p_i} \cdot \left| {P_i}^{-1} \right|_{p_i} \right|_{SQ}. \qquad (21)$$

From the definition of k_i (19) we can derive $\beta_i \cdot SQ - k_i p_i = 1$, where $\beta_i = \left| SQ^{-1} \right|_{p_i}$, which is equivalent to $\beta_i = \left| {P_i}^{-1} \right|_{p_i}$. Thus, $k_i = \frac{SQ}{p_i} \cdot \left| P_i \right|_{p_i} - \frac{1}{p_i}$, where $\frac{SQ}{p_i} \cdot \left| P_i^{-1} \right|_{p_i} = k_i + \frac{1}{p_i}$. Then substituting $k_i + \frac{1}{p_i}$ in (20) instead of k_i we get the scaled value of $D'(X)$. Thus, to obtain the value of X, substitute the calculated values in (21) and multiply by $\frac{P}{SQ}$.

$$X = \frac{P}{SQ} \cdot \left| \sum_{i=1}^{n} x_i \left(k_i + \frac{1}{p_i} \right) \right|_{SQ}. \qquad (22)$$

Consider this method with an example.

Example 5. Similarly, we are given RNS $(2,3,5,7,11)$ and a number $X = 1481 = (1,2,1,4,7)$. From the previous examples we know $P = 2310, P_1 = 1155, P_2 = 770, P_3 = 462, P_4 = 330, P_5 = 210$. Then $SQ = 2927$ and from (18) $k_1 = 1463, k_2 = 1951, k_3 = 1756, k_4 = 418, k_5 = 266$. Find the diagonal function

$$D(X) = |10655|_{2927} = 1874,$$

From (22) find the required value:

$$X = \frac{4334887}{2927} = 1481.$$

IV. The Akushsky Core Function Method Based on the Rank of Number

We present a fast technique for conversion numerical values from the RNS to positional notation. This approach involves using the Akushsky Core Function to find the rank of a number. The Akushsky Core Function [18] is defined by the following equation

$$C(X) = \sum_{i=1}^{n} w_i \left\lfloor \frac{X}{p_i} \right\rfloor. \tag{23}$$

where integers w_i are constants determined by the choice of the interpolation point. The numbers w_i in equation (23) can be arbitrary in a certain sense. It is they that define each particular core function and can vary depending on the problem to be solved. An algorithm for determining the optimal weights for the Akushsky core function is presented in [19].

Core function range value is calculated as

$$C(P) = C_P = \sum_{i=1}^{n} w_i P_i. \tag{24}$$

We define the so-called orthogonal bases B_i as

$$B_i = P_i \cdot \left|P_i^{-1}\right|_{p_i},$$

We also define the coefficients c_i as

$$c_i = C(B_i),$$

Rewrite (23) as

$$C(X) = \left|\sum_{i=1}^{n} c_i x_i\right|_{C_P} = \sum_{i=1}^{n} c_i \cdot x_i - \check{r}(X) \cdot C_P, \tag{25}$$

Then the rank of the Akushsky core function number can be defined as

$$\check{r}(X) = \left\lfloor \frac{\sum_{i=1}^{n} c_i \cdot x_i}{C_P} \right\rfloor. \tag{26}$$

There are three forms of representation of the CRT, each of them corresponds to a positional characteristic of the number represented in RNS.

The first form was represented in (4), the rank of a number in this representation can be calculated as follows

$$r(X) = \left\lfloor \sum_{i=1}^{n} \frac{1}{p_i} \cdot \left|P_i^{-1}\right|_{p_i} \cdot x_i \right\rfloor. \tag{27}$$

Second form

$$X = \left|\sum_{i=1}^{n} P_i \cdot \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i}\right|_P = \sum_{i=1}^{n} P_i \cdot \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} - \hat{r}(X) \cdot P, \tag{28}$$

where $\hat{r}(X)$ is the normalised rank of the number, which can be calculated as

$$\hat{r}(X) = \left\lfloor \sum_{i=1}^{n} \frac{1}{p_i} \cdot \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} \right\rfloor. \tag{29}$$

The third form is proposed and its rank is represented respectively in (25) and (26).

Consider the following properties.

Theorem 1: $\hat{r}(X) = -\frac{X}{P} + \sum_{i=1}^{n} \frac{\left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i}}{p_i}$.

Proof: According to the definition

$$\hat{r}(X) = \left\lfloor \sum_{i=1}^{n} \frac{\left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i}}{p_i} \right\rfloor = \left\lfloor \frac{1}{P} \sum_{i=1}^{n} \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} \cdot P_i \right\rfloor. \tag{30}$$

Since $\left\lfloor \frac{X}{P} \right\rfloor = \frac{X}{P} - \frac{|X|_P}{P}$, then

$$\hat{r}(X) = \frac{1}{P} \sum_{i=1}^{n} \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} \cdot P_i - \frac{1}{P} \cdot \left|\sum_{i=1}^{n} \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} \cdot P_i\right|_P.$$

According to the CRT, $\left|\sum_{i=1}^{n} \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} \cdot P_i\right|_P = X$, consequently,

$$\hat{r}(X) = \frac{1}{P} \sum_{i=1}^{n} \left|\left|P_i^{-1}\right|_{p_i} \cdot x_i\right|_{p_i} \cdot P_i - \frac{X}{P}.$$

The theorem is proved. ∎

Theorem 2: $\hat{r}(1) = -\frac{1}{P} + \sum_{i=1}^{n} \frac{\left|P_i^{-1}\right|_{p_i}}{p_i}$.

Proof: It follows directly from Theorem 1 that $\hat{r}(1) = -\frac{1}{P} + \sum_{i=1}^{n} \frac{\left|P_i^{-1}\right|_{p_i}}{p_i}$.

The theorem is proved. ∎

Let us examine the correlation between the ranks of positional characteristics.

Theorem 3: Let $p_1 < p_2 < \ldots < p_n$, the number $X \xrightarrow{RNS} (x_1, x_2, \ldots, x_n)$ and the weights of the Akushsky core function $w_1, w_2, \ldots, w_n$ satisfying the condition $0 \leq X < P$, then

$$\check{r}(X) = r(X) + \left\lfloor \frac{C(X)}{C_P} \right\rfloor. \tag{31}$$

Proof: Let us calculate c_i, we get

$$c_i = C\left(B_i\right) = \sum_{j=1}^{n} w_j \left\lfloor \frac{\left|P_i^{-1}\right|_{p_i} \cdot P_i}{p_j} \right\rfloor. \tag{32}$$

Since $\forall i \neq j$: $\left|P_i^{-1}\right|_{p_i} \cdot P_i \equiv 0 \bmod p_j$ and $\forall i$: $\left|P_i^{-1}\right|_{p_i} \cdot P_i \equiv 1 \bmod p_i$, then for $i \neq j$: $\left\lfloor \left|P_i^{-1}\right|_{p_i} \cdot P_i / p_i \right\rfloor = \frac{\left|P_i^{-1}\right|_{p_i} \cdot P_i}{p_i}$, and for $i = j$: $\left\lfloor \left|P_i^{-1}\right|_{p_i} \cdot P_i / p_i \right\rfloor = \frac{\left|P_i^{-1}\right|_{p_i} \cdot P_i - 1}{p_i}$, hence the coefficient c_i can be represented as follows

$$c_i = \left|P_i^{-1}\right|_{p_i} \cdot P_i \cdot \sum_{j=1}^{n} \frac{w_j}{p_j} - \frac{w_i}{p_i}. \tag{33}$$

Given that $\sum_{j=1}^{n} \frac{w_j}{p_j} = \frac{C_P}{P}$, then (33) is transformed to the form

$$\begin{aligned} \check{r}\left(X\right) &= \left\lfloor \frac{\sum_{i=1}^{n} c_i \cdot x_i}{C_P} \right\rfloor = \\ &= \left\lfloor \frac{1}{P} \cdot \sum_{i=1}^{n} \left|P_i^{-1}\right|_{p_i} \cdot P_i \cdot x_i - \frac{1}{C_P} \cdot \sum_{i=1}^{n} \frac{x_i \cdot w_i}{p_i} \right\rfloor. \end{aligned} \tag{34}$$

Substituting (34) into (28), we obtain

$$\check{r}\left(X\right) = \left\lfloor r\left(X\right) + \frac{X}{P} - \frac{1}{C_P} \cdot \sum_{i=1}^{n} \frac{x_i \cdot\ w_i}{p_i} \right\rfloor. \tag{35}$$

Considering that

$$\begin{aligned} \sum_{i=1}^{n} \frac{x_i \cdot\ w_i}{p_i} &= \sum_{i=1}^{n} \frac{\left(X - p_i \cdot \left\lfloor \frac{X}{p_i} \right\rfloor\right) \cdot w_i}{p_i} \\ &= X \cdot \sum_{i=1}^{n} \frac{w_i}{p_i} - \sum_{i=1}^{n} \left\lfloor \frac{X}{p_i} \right\rfloor \cdot\ w_i \\ &= X \cdot \frac{C_P}{P} - C\left(X\right). \end{aligned} \tag{36}$$

Substituting (36) into (35), we obtain

$$\check{r}\left(X\right) = \left\lfloor r\left(X\right) + \frac{C\left(X\right)}{C_P} \right\rfloor. \tag{37}$$

Since as $r\left(X\right) \in \mathbb{Z}$, and $\forall a \in \mathbb{R}, n \in \mathbb{Z}$: $\lfloor a + n \rfloor = \lfloor a \rfloor + n$, then

$$\check{r}\left(X\right) = r\left(X\right) + \left\lfloor \frac{C\left(X\right)}{C_P} \right\rfloor. \tag{38}$$

The theorem is proved. ■

Theorem 4: Let $p_1 < p_2 < \cdots < p_n$, a number $X \in \mathbb{Z}_P$ and an Akushsky core function with with all positive weights w_i be given, then $\check{r}\left(X\right) = r\left(X\right)$.

Proof: According to Theorem 3, $\check{r}\left(X\right) = r\left(X\right) + \left\lfloor \frac{C(X)}{C_P} \right\rfloor$. Given that the Akushsky core function contains no critical cores, $\forall X \in [0, P)$: $0 \leq C\left(X\right) < C_P$. Hence $\left\lfloor \frac{C(X)}{C_P} \right\rfloor = 0$, and hence $\check{r}\left(X\right) = r\left(X\right)$.

The theorem is proved. ■

Let us consider our proposed method with an example.

Example 6. Similarly, we are given RNS $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, p_5 = 11$ and a number $X = 1481 = (1, 2, 1, 4, 7)$. $P = 2310, P_1 = 1155, P_2 = 770, P_3 = 462, P_4 = 330, P_5 = 210$. Let us use a set of weights $w_1 = 0, w_2 = 0, w_3 = 0, w_4 = 0, w_5 = 1$.

Let us calculate the values of B_i:

$$B_1 = P_1 \cdot \left|P_1^{-1}\right| = 1155, B_2 = P_2 \cdot \left|P_2^{-1}\right| = 1540,$$
$$B_3 = P_3 \cdot \left|P_3^{-1}\right| = 1386, B_4 = P_4 \cdot \left|P_4^{-1}\right| = 330,$$
$$B_5 = P_5 \cdot \left|P_5^{-1}\right| = 210.$$

Then we find the value of the core function range by (24)

$$C\left(P\right) = C_P = 210.$$

Find the value of coefficients c_i:

$$c_1 = 105, c_2 = 140, c_3 = 126, c_4 = 30, c_5 = 19.$$

Then the rank of the number is

$$\check{r}\left(X\right) = \left\lfloor \frac{105 \cdot 1 + 140 \cdot 2 + 126 \cdot 1 + 30 \cdot 4 + 19 \cdot 7}{210} \right\rfloor = 3.$$

Thus,

$$X = 1155 \cdot 1 + 1540 \cdot 2 + 1386 \cdot 1 + 330 \cdot 4 + 210 \cdot 7 - 3 \cdot 2310 = 1481.$$

V. Performance Evaluation

The methodology expounded in Section 4 evinces an indisputable advantage over the approaches outlined in Section 3.

To validate the properties of each approach, every algorithm was carefully implemented in Python, and a comprehensive performance analysis was executed on a computer equipped with an Intel Core i7-7700HQ processor running at 2.80 GHz, 8 GB DDR4 RAM at 1196 MHz, and a 512 GB SSD, operating on Windows 10 Home Edition.

The study involves two significant phases:

Stage A examines the performance of three moduli by processing data sets of 50000, 100000, 200000, 350000, and 500000 using each of the proposed methods.

In Stage B, we expanded our analysis to cover 19 sets, varying from 3 to 21 moduli, with each modulo having an 8-bit dimensionality. We processed a data set of 100000 numbers.

Throughout the two-stage simulation, we measured the time characteristics of each method with attention to detail. To guarantee precision and dependability, we reiterated each measurement one hundred times and recorded the average time for evaluation. The findings of these experiments are presented concisely in Tables II and III, with time values depicted in seconds.

Let us conduct a detailed examination of the ensuing tables, delving deeper into the tabulated data with a scientific scrutiny. The provided information discusses two stages: Stage A and Stage B, focusing on their time characteristics and importance. Stage A is crucial for tracking method behavior with increasing data size. Analysis of the data shows a linear growth, which indicates the stability of the obtained method using the core function. To enhance understanding, graphs will be presented.

Table II provides insights into the time-related features observed during Stage B, underscoring the significance of this phase akin to Stage A. In a practical system comprising the control system may encompass various configurations, such as two, four, six, or more moduli. Consequently, exploring the behavior of methods in relation to the number of moduli within the system becomes imperative. The acquired data not only facilitates an understanding of methodological performance but also allows for inferences regarding the stability of the methods.

The data from the given table were utilized to create visual representations in the form of figures. In addition, a more comprehensive analysis was enabled by extrapolating the acquired values through polynomial methods, extending the perspective on the outcomes.

Upon scrutinizing the acquired outcomes, we can extrapolate the following deductions. Examining the graphical representation in Fig. 1, it becomes apparent that conventional methodologies demonstrate efficacy particularly when handling a limited quantity of numerical inputs. However, starting from the processing of two hundred thousand numbers, MRC method and interval method begin to lose.

A similar situation is apparent in the graph presented in Fig. 2. On average, our approach displays a time efficiency that is roughly 8% superior to that of the Approximate Method.

The comparative analysis conducted on methods for translating numbers from RNS to positional notation revealed that the method utilizing Akushsky core function and number rank offers certain advantages. This is due to the performance of addition and multiplication operations in positional notation within the mentioned approach. When performing calculations using MRC, each RNS modulo corresponds to a separate channel in which calculations are completed using modular arithmetic. However, these calculations are not performed in parallel. When using the interval method, it is necessary to complete operations such as addition, multiplication, and degree expansion in the positional system. Degree expansion can result in rather large values. One positive aspect of the interval method is the ability to process data in a conveyor-like manner.

VI. Conclusion

In this paper, we have presented a high speed method for converting numbers from RNS to positional notation. The proposed method offers a novel approach to achieve rapid and accurate conversions. By leveraging the inherent properties of the RNS and optimizing algorithms, our method streamlines the conversion process, minimizing computational complexities, and significantly reducing conversion times. Experiments demonstrate its superiority over conventional methods, showcasing notable improvements in speed.

While our proposed method represents a significant advancement, there is still room for further exploration and optimization. Future studies may investigate hybrid conversion techniques that combine the strengths of different algorithms, aiming to achieve even greater efficiency. Additionally, evaluating the proposed method's performance in large-scale systems and exploring its potential application in emerging technologies will be exciting avenues for future research.

Acknowledgment

The research was supported by the Russian Science Foundation Grant No. 19-71-10033, https://rscf.ru/en/project/19-71-10033/.

References

[1] Mohan P.A., Ananda Mohan P.V. Error Detection, Correction and Fault Tolerance in RNS-Based Designs. *Residue Number Systems: Theory and Applications*, Birkhäuser: Cham, Switzerland, 2016, pp. 163–175.

[2] Guo Z., Gao Z., Mei H., Zhao M., Yang J. Design and optimization for storage mechanism of the public blockchain based on redundant residual number system. *IEEE Access*, 2019, vol. 7, pp. 98546–98554.

[3] Al Badawi A., Polyakov Y., Aung K. M. M., Veeravalli B., Rohloff K. Implementation and performance evaluation of RNS variants of the BFV homomorphic encryption scheme. *IEEE Transactions on Emerging Topics in Computing*, 2019, vol. 9, no. 2, pp. 941–956.

[4] Molahosseini A. S., De Sousa L. S., Chang C. H. Embedded systems design with special arithmetic and number systems. *Cham, Switzerland: Springer*, 2017, p. 390.

[5] Valueva M. V., Nagornov N. N., Lyakhov P. A., Valuev G. V., Chervyakov N. I. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. *Mathematics and computers in simulation*, 2020, vol. 177, pp. 232–243.

[6] Dong X. A multi-secret sharing scheme based on the CRT and RSA. *International Journal of Electronics and Information Engineering*, 2015, vol. 2, no. 1., pp. 47–51.

[7] Van Vu T. Efficient implementations of the Chinese remainder theorem for sign detection and residue decoding. *IEEE Transactions on Computers*, 1985, vol. 100, no. 7, pp. 645–651.

[8] Chervyakov N. I., Molahosseini A. S., Lyakhov P. A., Babenko M. G., Deryabin M. A. Residue-to-binary conversion for general moduli sets based on approximate Chinese remainder theorem. *International journal of computer mathematics*, 2017, vol. 94, no. 9, pp. 1833–1849.

[9] Chervyakov N. I., Babenko M. G., Kuchukov V. A. Research of effective methods of conversion from positional notation to RNS on FPGA. *In Proceedings of the 2011 45th Annual Conference on Information Sciences and Systems*, 1–3 February 2017, IEEE: St. Petersburg/Moscow, Russia, 2017, pp. 371–375.

[10] Soderstrand M., Vernia C., Chang J. H. An improved residue number system digital-to-analog converter. *IEEE transactions on circuits and systems*, 1983, vol. 30, no. 12, pp. 903–907.

[11] Babenko M., Golimblevskaia E. About One Property of Number Rank in RNS. *In Proceedings of the 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering*, 26–29 January 2021, St. Petersburg, FL, USA, 2021, pp. 212–216.

[12] Isupov K. High-Performance Computation in Residue Number System Using Floating-Point Arithmetic. *Computation*, 2021, vol. 9, no. 2, 9.

[13] Chervyakov N.I., Averbukh V.M., Babenko M.G., Lyakhov P.A., Gladkov A.V., Gapochkin A.V. An Approximate Method for Performing Non-Modular Operations in a System of Residual Classes. *Fundam. Res.*, 2012, vol. 6, pp. 189–193.

[14] Yassine H. M., Moore W. R. Improved mixed-radix conversion for residue number system architectures. *IEE Proceedings G (Circuits, Devices and Systems)*, 1991, vol. 138, no. 1, pp. 120–124.

[15] Gladkov A., Gladkova N., Kucherov N. Analytical Review of Methods for Detection, Localization and Error Correction in the Residue Number System. *In Proceedings of the International Conference on Mathematics and its Applications in New Computer Systems*, 25–28 October 2022, Munich, Germany, Springer: Berlin/Heidelberg, 2022, pp. 507–514.

[16] Babenko M., Piestrak S. J., Chervyakov N., Deryabin M. The study of monotonic core functions and their use to build RNS number comparators. *Electronics*, 2021, vol. 10, no. 9, p. 1041.

[17] Piestrak, S. J. A note on RNS architectures for the implementation of the diagonal function. *Information Processing Letters*, 2015, vol. 115, no. 4, pp. 453–457.

TABLE II: The result of the study of stage A

Amount	CRT Method	Approximate Method	MRC Method	Interval Method	DF Method	Rank Core Method
50000	0.17184758	0.15310092	0.17178512	0.87428927	0.28112435	0.13308518
100000	0.37491608	0.33490013	1.61959763	1.9496377	0.48421788	0.29072099
200000	0.70988417	0.57174363	2.60662079	3.31276298	1.19526315	0.45420003
350000	1.19558525	1.14163585	4.49329138	5.65635467	1.68907547	0.87942809
500000	1.68847227	1.59833269	6.57562566	8.14383984	2.54651117	1.25240469

TABLE III: The result of stage B study: dimension of modulo set p[n] where n represents modulo count in the ensemble

p[n]	CRT Method	Approximate Method	MRC Method	Interval Method	DF Method	Rank Core Method
3	0.04886961	0.04188609	0.16806006	0.19650173	0.07034159	0.0388873
4	0.04986429	0.04387736	0.21841407	0.25733018	0.08476949	0.03990845
5	0.05085826	0.04582379	0.34164977	0.30221629	0.09973574	0.04392817
6	0.06984472	0.07378912	0.34810019	0.33629251	0.10273242	0.0588873
7	0.07682538	0.08676386	0.41370296	0.39549708	0.11066651	0.06990845
8	0.07836747	0.09275365	0.51267076	0.43252301	0.12862134	0.07392817
9	0.08676624	0.09826112	0.59272242	0.45488119	0.13066811	0.09067698
10	0.09473872	0.10372066	0.62236333	0.53865314	0.13461476	0.09264201
11	0.11070347	0.11968732	0.72314477	0.55988812	0.15419126	0.09558553
12	0.11466908	0.12268424	0.77975607	0.6223812	0.17807412	0.11655969
13	0.12469697	0.12665558	0.88087988	0.63008047	0.19151998	0.12052173
14	0.13267827	0.12510133	1.01868820	0.66010213	0.19850206	0.12311763
15	0.13463926	0.12766194	1.04511428	0.75057304	0.20049644	0.12251919
16	0.16458368	0.12769699	1.15422750	0.81782241	0.20647359	0.11738696
17	0.16755462	0.13862944	1.29933691	0.86782241	0.23633909	0.12436595
18	0.16951680	0.14361358	1.35722113	0.87273455	0.24734974	0.13237681
19	0.17810869	0.15760803	1.40875983	0.93827939	0.26701593	0.13935819
20	0.18051696	0.15960505	1.51713409	0.98166609	0.26928353	0.15541186
21	0.18350887	0.16758013	1.63444066	1.02923965	0.27526116	0.16631331

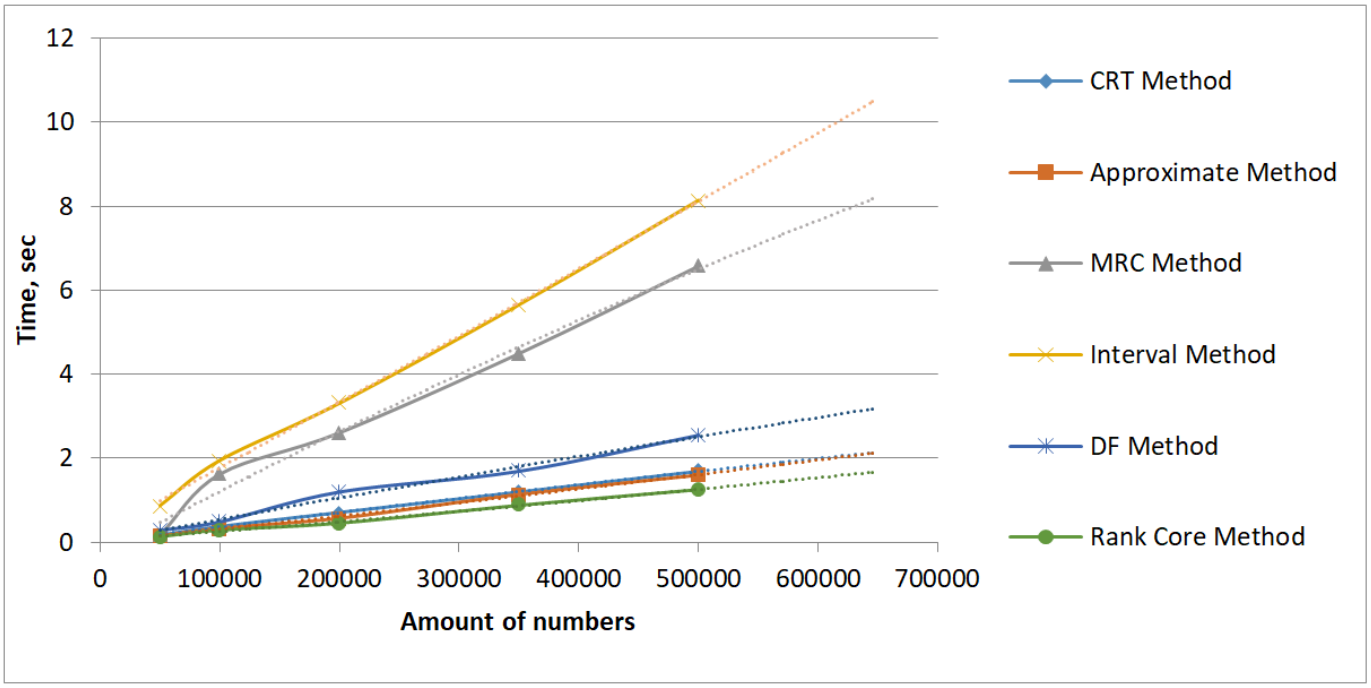

Fig. 1: Findings from stage A analysis.

[18] Akushsky I.Y., Burtsev V.M., Pak I.T. O novoj pozicionnoj harakteristike nepozicionnogo koda i ee prilozhenii [About the New Positional Characteristic of the Non-Positional Code and Its Application]. *Theory of Coding and Optimization of Complex Systems*, Nauka: Alma-Ata, Kazakhstan, 1977, pp. 8–16 (in Russian).

[19] Shiriaev E., Kucherov N., Babenko M., Lutsenko V., Al-Galda S. Algorithm for Determining the Optimal Weights for the Akushsky Core Function with an Approximate Rank. *Applied Sciences*, 2023, vol. 13, no. 18, p. 10495.

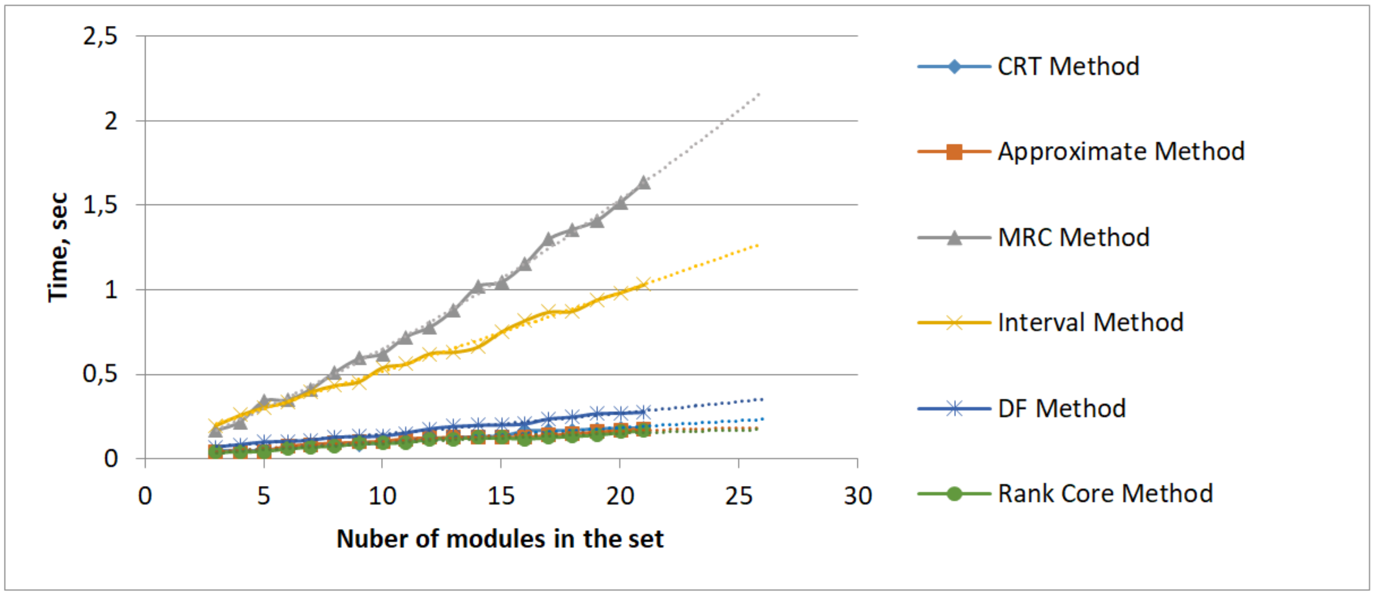

Fig. 2: Findings from stage B analysis.