Theory of Computing Systems (2021) 65:1117–1140
https://doi.org/10.1007/s00224-021-10035-y

Computationally Efficient Approach to Implementation of the Chinese Remainder Theorem Algorithm in Minimally Redundant Residue Number System

Mikhail Selianinau[1]

Accepted: 25 January 2021 / Published online: 20 April 2021

Abstract
In this paper, we deal with the critical problem of performing non-modular operations in the Residue Number System (RNS). The Chinese Remainder Theorem (CRT) is widely used in many modern computer applications. Throughout the article, an efficient approach for implementing the CRT algorithm is described. The structure of the rank of an RNS number, a principal positional characteristic of the residue code, is investigated. It is shown that the rank of a number can be represented by a sum of an inexact rank and a two-valued correction to it. We propose a new variant of minimally redundant RNS, which provides low computational complexity for the rank calculation, and its effectiveness analyzed concerning conventional non-redundant RNS. Owing to the extension of the residue code, by adding the excess residue modulo 2, the complexity of the rank calculation goes down from $O\left(k^2\right)$ to $O\left(k\right)$ with respect to required modular addition operations and lookup tables, where k equals the number of non-redundant RNS moduli.

Keywords Residue number system · Chinese remainder theorem · Non-modular operations · Rank of a number · Mixed-radix representation

1 Introduction

At present, the field of high-performance computing is developing extremely rapidly. That leads to qualitatively new requirements imposed on numerical methods and computing algorithms. Practically, all the well-known approaches to the development

✉ Mikhail Selianinau
m.selianinau@ujd.edu.pl

[1] Jan Dlugosz University in Czestochowa, Armii Krajowej 13/15, 42-200 Czestochowa, Poland

of high-performance computing have one trait in common, that essence consists in the application of certain parallel forms of data representation and processing.

One of the most promising avenues to significantly improve the performance and reliability of computing facilities is the implementation of non-conventional methods of number representation, as well as corresponding variants of computer arithmetic. Within the framework of this approach, the non-positional number systems with a parallel structure, in particular, the RNS, occupy a place of special importance [3, 5, 18–20, 27, 28].

Since its inception in the mid-1950s till the present day, the RNS has attracted the continuous attention of researchers in computer technology, communications, numerical methods, cryptography, and other fields. An RNS, whose ideological roots date back to the standard topics of number theory, has natural internal parallelism. The main advantage of an RNS is a unique ability to decompose the large word-length numbers up into the set of independent short word-length residues, which can be processed in parallel. The carry-free property and speed-up provided by the parallelism of RNS allows carrying out the sequences of modular operations quickly and easily.

Thus, it becomes possible to parallelize the computations at the level of arithmetic operations, which is fundamentally crucial for modern high-performance computing systems. Unlike traditional weighted number systems (WNS), an RNS provides an entirely different approach for increasing the speed and reliability of digital information processing, as well as obtaining new and more advanced computational structures. In this regard, RNS arithmetic is the cornerstone of many modern algorithms of parallel computer algebra, as well as it is in demand in many fundamental applications of science and technology.

Due to the inherent code parallelism, an RNS possesses a few essential advantages over the conventional number system in the field of high-speed computing. A broad class of fast algorithms can be implemented based on modular arithmetic in areas such as digital signal and image processing, distributed information and communication systems, computer networks, information security systems, cloud computing, etc. [19, 20, 27, 30]. Moreover, these RNS algorithms can effectively be applied even to processor platforms functioning according to the conventional information-processing approach based on the weighted binary number system. The parallel and carry-free RNS arithmetic corresponds with high-performance modern embedded systems requirements [19].

However, the numerical solution of a wide range of science and technology problems is often required to perform complicated and heterogeneous processes of computation. In RNS, this results in the need to carry out a set of so-called non-modular operations, such as magnitude comparison, sign determination, overflow detection, general division, scaling, residue-to-binary conversion, etc. The problem of effective implementation of non-modular operations is constantly receiving considerable attention after the formalization of the processing principles in RNS arithmetic [3, 5, 18–20].

To perform in RNS the non-modular operations, it is not enough to take into account only the individual residues of the modular code. Furthermore, it is necessary to estimate the value of the total number, which, in general, is hampered by the non-positional nature of RNS.

Unfortunately, up until now, the underlying unresolved problem for creating a high-speed RNS arithmetic consists of the computational complexity of non-modular operations. Precisely because of the lack of fast algorithms of non-modular operations, the use of RNS is useful in cases only when the modular addition and multiplication operations make up the bulk of required amounts of computation, and the number of non-modular operations is relatively small. This circumstance restricts the scope of practical applications of RNS to a narrow class of specific tasks, for example, digital filtering, fast Fourier transform, cryptographic transformations, etc.

Therefore, the design of high-performance algorithms for non-modular operations is an urgent problem at the current stage of RNS arithmetic development and its application in computer sciences. To solve this problem is necessary to design efficient approaches and methods for the fast weighted representation of RNS numbers by their residue codes. That will make it possible for the extensive use of RNS arithmetic for high-speed computing in many priority areas of science and technology.

2 The Basic Principles of RNS Arithmetic

The abstract algebra and number theory are formed the theoretical basis of residue arithmetic [6, 11].

In the conventional non-redundant RNS with the set $\{m_1, m_2, \ldots, m_k\}$ of k pairwise relatively prime odd moduli ($m_i > 2, i = 1, 2, ..., k$), the integer number $X \in \mathbb{Z}_{M_k}$ is represented by an ordered set of residues $(\chi_1, \chi_2, \ldots, \chi_k)$ that is generally called a residue (modular) code, where $\mathbb{Z}_{M_k} = \{0, 1, \ldots, M_k - 1\}$, $M_k = \prod_{i=1}^{k} m_i$, $\chi_i = |X|_{m_i}$ $(i = 1, 2, \ldots, k)$; $|Y|_m$ denotes the least non-negative residue of the integer Y modulo m, i.e., $|Y|_m \in \mathbb{Z}_m = \{0, 1, \ldots, m - 1\}$.

As is known, the main fundamental advantage of RNS arithmetic as compared with the arithmetic of WNS consists of the performance of addition, subtraction and multiplication operations in parallel at the level of small word-length residues. In other words, these operations decompose into independent components with respect to basic moduli $m_1, m_2, \ldots, m_k$. The modular operation $\circ \in \{+, -, \times\}$ on the integers $A = (\alpha_1, \alpha_2, \ldots, \alpha_k)$ and $B = (\beta_1, \beta_2, \ldots, \beta_k)$ $(\alpha_i = |A|_{m_i}, \beta_i = |B|_{m_i}, i = 1, 2, \ldots, k)$ is carried out independently for each of the RNS moduli, i.e., by the rule

$$\begin{aligned} A \circ B &= (\alpha_1, \alpha_2, \ldots, \alpha_k) \circ (\beta_1, \beta_2, \ldots, \beta_k) \\ &= \left(|\alpha_1 \circ \beta_1|_{m_1}, |\alpha_2 \circ \beta_2|_{m_2}, \ldots, |\alpha_k \circ \beta_k|_{m_k}\right). \end{aligned}$$

The efficiency of RNS arithmetic is primarily determined by the complexity and performance of non-modular procedures, which are used as essential elementary procedures for implementing more complex computational algorithms. The fundamental principles of designing high-speed variants of RNS arithmetic consist in carrying out the following crucial conditions imposed on non-modular operations: parallelism, high modularity, and simplicity of pipelining at the level of small word-length residue operations.

The root problem in RNS arithmetic is that the integer value of the number $X = (\chi_1, \chi_2, \ldots, \chi_k)$ depends on all its residues together. In the RNS, its evaluation underlies all non-modular operations. As is known, for performing non-modular operations, the so-called positional characteristics of the residue code are used [3, 5, 18–20]. In general, all these characteristics depend on part or all residues of the number X. They can be defined as a specific mathematical function that, by some method or other, allows estimation of the positional value of X. It is quite clear that the computational complexity of the applied positional characteristics eventually determines the efficiency of RNS arithmetic constructed on their basis.

The traditional implementations of non-modular operations in RNS arithmetic are based on the reverse conversion from residues back to an integer. There are two canonical techniques: the straightforward conversion method based on the CRT algorithm, and the conversion of the residue code to a weighted representation in Mixed-Radix System (MRS) [2, 12, 18–20, 31]. In general, all other conversion methods are variants of these two main methods.

At the same time, in recent years, the CRT has been intensively studied with its applications in high-performance computing. That led to the development of a sufficiently wide class of specific methods based on the calculation of positional characteristics of RNS numbers, which support effective implementations of non-modular operations. Some of the generally accepted characteristics are the core function, rank function, interval index, parity, diagonal function, and quotient function [1, 3–5, 7–10, 15–17, 19, 21, 22, 24].

Nevertheless, in conventional non-redundant RNS, the calculation of all these characteristics is complicated and quite time-consuming. That is why the non-modular operations limit the applications of RNS arithmetic and restrict its general usage.

3 The Rank of an RNS Number

As is known, in the case of pairwise relatively prime moduli $m_1, m_2, \ldots, m_k$ the system of simultaneous linear congruences

$$\begin{cases} X \equiv \chi_1 \pmod{m_1}, \\ X \equiv \chi_2 \pmod{m_2}, \\ \qquad \cdots \\ X \equiv \chi_k \pmod{m_k} \end{cases}$$

has a unique solution that is the residue class modulo M_k specified by the congruence

$$X \equiv \left(\sum_{i=1}^{k} M_{i,k} \mu_{i,k} \chi_i \right) \pmod{M_k}, \tag{1}$$

where $M_{i,k} = M_k / m_i$, $\mu_{i,k} = \left| M_{i,k}^{-1} \right|_{m_i}$ $(i = 1, 2, \ldots, k)$; $\left| Y^{-1} \right|_m$ denotes the multiplicative inverse of the integer Y modulo m.

In essence, the equation (1) represents the CRT [14, 20, 26].

In the RNS with the set $m_1, m_2, \ldots, m_k$ of k pairwise relatively prime moduli it is possible to represent at most M_k integers. Therefore, the set $\mathbb{Z}_{M_k}$ usually used as a dynamic range of the RNS.

Thus, modular coding is defined as a mapping $\varphi_{RNS}\colon \mathbb{Z}_{M_k} \to \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_k}$ that assigns the residue code $(\chi_1, \chi_2, \ldots, \chi_k)$ to each $X \in \mathbb{Z}_{M_k}$. At the same time, regarding the decoding mapping $\varphi_{RNS}^{-1}\colon \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_k} \to \mathbb{Z}_{M_k}$ based on the relationship (1), the following rule is valid:

$$X = \left| \sum_{i=1}^{k} M_{i,k} \mu_{i,k} \chi_i \right|_{M_k} = \left| \sum_{i=1}^{k} B_i \chi_i \right|_{M_k} \left(X \in \mathbb{Z}_{M_k} \right), \tag{2}$$

where the integer numbers $B_1, B_2, \ldots, B_k$ are the basic primitive constants in the given RNS; $B_i = M_{i,k} \mu_{i,k}$, $i = 1, 2, \ldots, k$ [3].

As can be seen, the reverse conversion from RNS to WNS by using the straightforward application of (2) requires $O(k)$ of large word-lengths addition operations modulo M_k, which is the product of all moduli $m_1, m_2, \ldots, m_k$. If we assume that the processing of such long L-bit numbers ($L = \lceil \log_2 M_k \rceil$) is comparable in time with k operations on the small residues, then the complexity of this method is equal to $O\left(k^2\right)$ Because of this fact, a given approach is practically unacceptable for high-speed computing, especially in cryptographic applications.

Here and further $\lceil x \rceil$ denotes the smallest integer greater than or equal to x.

To circumvent the problem of slow addition operations modulo M_k, the equation (2) can be written as an exact integer equality

$$X = \sum_{i=1}^{k} B_i \chi_i - r(X) M_k, \tag{3}$$

where $r(X)$ is a non-negative integer called the true rank of the number X [3, 5].

From (3) it follows that the upper bound of $r(X)$ depends on weighting factors $\mu_{1,k}, \mu_{2,k}, \ldots, \mu_{k,k}$ and it is a rather large value for moduli-sets $\{m_1, m_2, \ldots, m_k\}$ suitable for practical application.

By using Euclid's Division Lemma, we have

$$\mu_{i,k} \chi_i = \left| \mu_{i,k} \chi_i \right|_{m_i} + \left\lfloor \frac{\mu_{i,k} \chi_i}{m_i} \right\rfloor m_i = \chi_{i,k} + \left\lfloor \frac{\mu_{i,k} \chi_i}{m_i} \right\rfloor m_i, \tag{4}$$

where $\chi_{i,k}$ is a normalized residue modulo m_i:

$$\chi_{i,k} = \left| \mu_{i,k} \chi_i \right|_{m_i} = \left| M_{i,k}^{-1} \chi_i \right|_{m_i} \ (i = 1, 2, \ldots, k), \tag{5}$$

$\lfloor x \rfloor$ denotes the largest integer less than or equal to x.

Substituting (4) into (2) and taking into consideration (5), we have

$$X = \left| \sum_{i=1}^{k} M_{i,k} \chi_{i,k} + M_k \sum_{i=1}^{k} \left\lfloor \frac{\mu_{i,k} \chi_i}{m_i} \right\rfloor \right|_{M_k}$$

that is equivalent to

$$X = \left| \sum_{i=1}^{k} M_{i,k} \chi_{i,k} \right|_{M_k} = \left| \sum_{i=1}^{k} M_{i,k} \left| M_{i,k}^{-1} \chi_i \right|_{m_i} \right|_{M_k}. \tag{6}$$

Similarly to (3), according to (6), the number X can also be written as

$$X = \sum_{i=1}^{k} M_{i,k} \chi_{i,k} - \rho_k (X) M_k, \tag{7}$$

where the integer $\rho_k (X)$ is the normalized rank (or, briefly, rank) of the number X.

Equation (7) is called the rank form (or CRT-form) of the number X. In essence, the rank $\rho_k (X)$ is the CRT reconstruction coefficient that is indicated how many times the RNS dynamic range exceeded when converting the residue code $(\chi_1, \chi_2, \ldots, \chi_k)$ of the number X to its integer value according to (7).

Let us now estimate the upper bound of the rank $\rho_k (X)$. Dividing (7) by M_k with taking the integer part of both sides of the obtained equality and taking into account the fact that $\lfloor X/M_k \rfloor = 0$, we have

$$\rho_k (X) = \left\lfloor \frac{1}{M_k} \sum_{i=1}^{k} M_{i,k} \chi_{i,k} \right\rfloor = \left\lfloor \sum_{i=1}^{k} \frac{\chi_{i,k}}{m_i} \right\rfloor. \tag{8}$$

Hence, owing to $\chi_{i,k} \in \mathbb{Z}_{m_i}$ $(i = 1, 2, \ldots, k)$ we get the following estimation

$$\begin{aligned} 0 \le \rho_k (X) \le \left\lfloor \sum_{i=1}^{k} \frac{m_i - 1}{m_i} \right\rfloor &= k + \left\lfloor -\sum_{i=1}^{k} \frac{1}{m_i} \right\rfloor \\ &= k - \left\lceil \sum_{i=1}^{k} \frac{1}{m_i} \right\rceil = k - \left\lceil \frac{1}{M_k} \sum_{i=1}^{k} M_{i,k} \right\rceil. \end{aligned} \tag{9}$$

Let us consider the number $Y = (\gamma_1, \gamma_2, \ldots, \gamma_k)$, where $\gamma_i = \left| M_{i,k} \right|_{m_i}$, $i = 1, 2, \ldots, k$. Then, according to (7), we have

$$Y = \sum_{i=1}^{k} M_{i,k} \gamma_{i,k} - \rho_k (Y) M_k = \sum_{i=1}^{k} M_{i,k} - \rho_k (Y) M_k$$

because of $\gamma_{i,k} = \left| M_{i,k}^{-1} \gamma_i \right|_{m_i} = \left| M_{i,k}^{-1} M_{i,k} \right|_{m_i} = 1$.

Thus,

$$\sum_{i=1}^{k} M_{i,k} = Y + \rho_k (Y) M_k.$$

Therefore, because $0 < Y < M_k$, we have

$$\left\lceil \frac{1}{M_k} \sum_{i=1}^{k} M_{i,k} \right\rceil = \left\lceil \frac{1}{M_k} (Y + \rho_k (Y) M_k) \right\rceil = \rho_k (Y) + \left\lceil \frac{Y}{M_k} \right\rceil = \rho_k (Y) + 1.$$

Since $\rho_k(Y) \geq 0$, regardless of the selected moduli-set $\{m_1, m_2, \ldots, m_k\}$, then, according to (9), the following estimation is valid

$$0 \leq \rho_k(X) \leq k - 1 - \rho_k(Y) \leq k - 1.$$

Hence, $\rho_k(X) \in \mathbb{Z}_k = \{0, 1, \ldots, k-1\}$.

The summands $M_{i,k}\chi_{i,k}$ in the equation (6) have smaller values than $M_{i,k}\mu_{i,k}\chi_i$ in (2) ($i = 1, 2, \ldots, k$), so $\rho_k(X)$ is significantly less than $r(X)$. Therefore, the CRT-form (7) in comparison with (3) is a more suitable basis for performing the non-modular operations in RNS arithmetic.

The structure of the rank characteristic and the methods of its calculation in the non-redundant RNS were studied in detail in [3, 5]. At the same time, the main drawback of known approaches is the difficulty in the calculation of the rank and straightforward implementation of the CRT. Consequently, these algorithms are not suitable for designing efficient variants of RNS arithmetic, especially for large word-length numbers.

4 A Novel Method for Calculating the Rank in Non-redundant RNS

In the RNS, the rank $\rho_k(X)$ is a positional characteristic of the residue code of primary importance since on its basis all non-modular operations can be implemented. Therefore, the development of effective methods and algorithms for calculating the rank $\rho_k(X)$ occupies an essential place in the practice of applying RNS to construct high-performance modular computational structures.

Knowledge of the rank $\rho_k(X)$ allows estimation of the integer value of the RNS-number X. It is quite clear that the level of complexity of rank calculation ultimately determines the efficiency of modular arithmetic constructed on its basis.

From the relationship for the CRT-form (7) of the number X it follows that

$$X_k = X + \rho_k(X) M_k, \tag{10}$$

where

$$X_k = \sum_{i=1}^{k} M_{i,k}\chi_{i,k}. \tag{11}$$

We will call this number the CRT-number of X.

Then, by using the following notations: $M_{k-1} = \prod_{i=1}^{k-1} m_i$ and $M_{i,k-1} = M_{k-1}/m_i$ ($i = 1, 2, \ldots, k-1$), as well as taking into account that $M_{i,k} = M_k/m_i = M_{k-1}m_k/m_i = M_{i,k-1}m_k$ and $M_{k,k} = M_k/m_k = M_{k-1}$, we can write the CRT-number X_k (see (11)) as

$$X_k = \sum_{i=1}^{k-1} M_{i,k}\chi_{i,k} + M_{k,k}\chi_{k,k} = \sum_{i=1}^{k-1} M_{i,k-1}m_k\chi_{i,k} + M_{k-1}\chi_{k,k}. \tag{12}$$

The number $m_k\chi_{i,k}$ in equation (12), according to Euclid's Division Lemma, can be represented as

$$m_k\chi_{i,k} = \left|m_k\chi_{i,k}\right|_{m_i} + \left\lfloor \frac{m_k\chi_{i,k}}{m_i} \right\rfloor m_i = \chi_{i,k-1} + \left\lfloor \frac{m_k\chi_{i,k}}{m_i} \right\rfloor m_i, \tag{13}$$

taking into account that

$$\left|m_k\chi_{i,k}\right|_{m_i} = \left|m_k\left|M_{i,k}^{-1}\chi_i\right|_{m_i}\right|_{m_i} = \left|m_kM_{i,k}^{-1}\chi_i\right|_{m_i} = \left|M_{i,k-1}^{-1}\chi_i\right|_{m_i} = \chi_{i,k-1}.$$

Thus,

$$X_k = \sum_{i=1}^{k-1} M_{i,k-1}\chi_{i,k-1} + M_{k-1}\sum_{i=1}^{k-1}\left\lfloor \frac{m_k\chi_{i,k}}{m_i} \right\rfloor + M_{k-1}\chi_{k,k}.$$

Finally, we have

$$X_k = X_{k-1} + M_{k-1}S_k\left(X\right), \tag{14}$$

where

$$X_{k-1} = \sum_{i=1}^{k-1} M_{i,k-1}\chi_{i,k-1}, \tag{15}$$

$$S_k\left(X\right) = \sum_{i=1}^{k} R_{i,k}\left(\chi_i\right), \tag{16}$$

$$R_{i,k}\left(\chi_i\right) = \left\lfloor \frac{m_k\chi_{i,k}}{m_i} \right\rfloor \ \left(i = 1, 2, \ldots, k\right). \tag{17}$$

Remark 1 Taking into account (13), we have

$$R_{i,k}\left(\chi_i\right) = \frac{m_k\chi_{i,k} - \chi_{i,k-1}}{m_i}.$$

As far as $R_{i,k}\left(\chi_i\right) \in \mathbb{Z}_{m_k}$, we can then obtain a residue modulo m_k on both sides of this equality. Hence, it then follows that

$$R_{i,k}\left(\chi_i\right) = \left|\frac{m_k\chi_{i,k} - \chi_{i,k-1}}{m_i}\right|_{m_k} = \left|-\frac{\chi_{i,k-1}}{m_i}\right|_{m_k} = \left|-\frac{\left|M_{i,k-1}^{-1}\chi_i\right|_{m_i}}{m_i}\right|_{m_k} \tag{18}$$

for $i = 1, 2, \ldots, k-1$. At the same time, according to (17),

$$R_{k,k}\left(\chi_k\right) = \chi_{k,k} = \left|M_{k,k}^{-1}\chi_k\right|_{m_k} = \left|M_{k-1}^{-1}\chi_k\right|_{m_k}. \tag{19}$$

Similarly, as described above for the number X_k (see (14)–(16)), the numbers X_i $(i = k-1, k-2, \ldots, 1)$ can be written as

$$\begin{aligned} X_{k-1} &= X_{k-2} + M_{k-2} S_{k-1}(X), \\ X_{k-2} &= X_{k-3} + M_{k-3} S_{k-2}(X), \\ &\cdots \\ X_2 &= X_1 + M_1 S_2(X), \\ X_1 &= M_0 S_1(X), \end{aligned}$$

where $M_0 = 1$, $S_1(X) = \chi_1$.

Finally, the CRT-number X_k can be represented as

$$X_k = \sum_{l=1}^{k} M_{l-1} S_l(X), \tag{20}$$

where $M_{l-1} = \prod_{i=1}^{l-1} m_i$.

The integer $S_l(X)$ for $l = 2, 3, \ldots, k$ using Euclid's Division Lemma can be written as

$$S_l(X) = R_l(X) + m_l Q_l(X). \tag{21}$$

At the same time,

$$R_l(X) = |S_l(X)|_{m_l} = \left| \sum_{i=1}^{l} R_{i,l}(\chi_i) \right|_{m_l}, \tag{22}$$

$$Q_l(X) = \left\lfloor \frac{1}{m_l} S_l(X) \right\rfloor = \left\lfloor \frac{1}{m_l} \sum_{i=1}^{l} R_{i,l}(\chi_i) \right\rfloor, \tag{23}$$

where

$$R_{i,l}(\chi_i) = \left| -\frac{\left| M_{i,l-1}^{-1} \chi_i \right|_{m_i}}{m_i} \right|_{m_l} \quad (i \neq l), \tag{24}$$

$$R_{l,l}(\chi_l) = \chi_{l,l} = \left| M_{l-1}^{-1} \chi_l \right|_{m_l}, \tag{25}$$

$M_{i,l-1} = M_{l-1}/m_i$.

It is evident that $Q_l(X)$ is equal to the number of overflows that occurred when calculating the sum $R_l(X)$ of l residues $R_{1,l}(\chi_1), R_{2,l}(\chi_2), \ldots, R_{l,l}(\chi_l)$ modulo m_l $(l = 2, 3, \ldots, k)$.

It can be noted that $R_1(X) = \chi_1$ and $Q_1(X) = 0$ since $S_1(X) = \chi_1$.
Taking into account (20) and (21), we have

$$\begin{aligned} X_k &= \sum_{l=1}^{k} M_{l-1}\left(R_l(X) + m_l Q_l(X)\right) \\ &= \sum_{l=1}^{k} M_{l-1} R_l(X) + \sum_{l=1}^{k} M_l Q_l(X) \\ &= \sum_{l=1}^{k} M_{l-1} R_l(X) + \sum_{l=1}^{k-1} M_l Q_l(X) + M_k Q_k(X). \end{aligned}$$

Thus,

$$X_k = X_k^{(R)} + X_{k-1}^{(Q)} + M_k Q_k(X), \tag{26}$$

where

$$X_k^{(R)} = \sum_{l=1}^{k} M_{l-1} R_l(X), \tag{27}$$

$$X_{k-1}^{(Q)} = \sum_{l=1}^{k-1} M_l Q_l(X). \tag{28}$$

As is known, in the MRS based on the moduli-set $\{m_1, m_2, \ldots, m_k\}$ the integer $X \in \mathbb{Z}_{M_k}$ is represented by the k-tuple $\langle x_k, x_{k-1}, \ldots, x_1\rangle$ of mixed-radix digits, resulting in

$$X = x_1 + x_2 M_1 + \cdots + x_k M_{k-1} = \sum_{i=1}^{k} x_i M_{i-1},$$

where $x_i \in \mathbb{Z}_{m_i} = \{0, 1, \ldots, m_i - 1\}$, $i = 1, 2, \ldots, k$ [18–20].

As follows from equation (27), the number $X_k^{(R)}$ is represented by the k-tuple $\left\langle x_k^{(R)}, x_{k-1}^{(R)}, \ldots, x_1^{(R)}\right\rangle$ of mixed-radix digits $x_l^{(R)} = R_l(X)$, $R_l(X) \in \mathbb{Z}_{m_l}$, $l = 1, 2, \ldots, k$. It is evident that $X_k^{(R)} < M_k$.

As for the number $X_{k-1}^{(Q)}$ (see (28)), it can be written as

$$X_{k-1}^{(Q)} = \sum_{l=2}^{k} M_{l-1} Q_{l-1}(X) = \sum_{l=1}^{k} M_{l-1} \widehat{Q}_l(X),$$

where $\widehat{Q}_1(X) = 0$, $\widehat{Q}_2(X) = Q_1(X) = 0$, $\widehat{Q}_l(X) = Q_{l-1}(X)$ $(l = 3, 4, \ldots, k)$.

At the same time, as follows from (23) in the case when the subscript l is substituted by $l-1$,

$$\widehat{Q}_l(X) = Q_{l-1}(X) = \left\lfloor \frac{1}{m_{l-1}} \sum_{i=1}^{l-1} R_{i,l-1}(\chi_i) \right\rfloor < l-1$$

because of the residue $R_{i,l-1}(\chi_i) \le m_{l-1}-1$ $(l=3,4,\ldots,k)$.

Thus, $\widehat{Q}_l(X) \in \mathbb{Z}_{l-1} = \{0,1,\ldots,l-2\}$. Hence, $X_{k-1}^{(Q)}$ can be considered as a mixed-radix number $\left\langle x_k^{(Q)}, x_{k-1}^{(Q)}, \ldots, x_1^{(Q)} \right\rangle$ $\left(x_1^{(Q)} = x_2^{(Q)} = 0;\ x_l^{(Q)} = Q_{l-1}(X), l=3,4,\ldots,k\right)$ in the case if $x_l^{(Q)} \in \mathbb{Z}_{m_l}$. Therefore, the following condition must be met: $\mathbb{Z}_{l-1} \subseteq \mathbb{Z}_{m_l}$ for all l. This leads to inequality

$$m_l \ge l-1 \ (l=1,2,\ldots,k).$$

If the moduli-set $\{m_1, m_2, \ldots, m_k\}$ is chosen according to this condition, which as a rule always holds for the RNS used in practical applications, then $X_{k-1}^{(Q)} \in \mathbb{Z}_{M_k}$, that is $X_{k-1}^{(Q)} < M_k$.

Thus,

$$0 \le X_k^{(R)} + X_{k-1}^{(Q)} \le 2(M_k - 1). \tag{29}$$

According to Euclid's Division Lemma,

$$X_k^{(R)} + X_{k-1}^{(Q)} = \left| X_k^{(R)} + X_{k-1}^{(Q)} \right|_{M_k} + M_k \left\lfloor \frac{X_k^{(R)} + X_{k-1}^{(Q)}}{M_k} \right\rfloor.$$

Therefore, from (26) we have

$$X_k = \left| X_k^{(R)} + X_{k-1}^{(Q)} \right|_{M_k} + M_k \left(Q_k(X) + \left\lfloor \frac{X_k^{(R)} + X_{k-1}^{(Q)}}{M_k} \right\rfloor \right). \tag{30}$$

From equation (30), according to (10), it follows that

$$X = \left| X_k^{(R)} + X_{k-1}^{(Q)} \right|_{M_k}, \tag{31}$$

$$\rho_k(X) = \widehat{\rho}_k(X) + \Delta_k(X), \tag{32}$$

where

$$\widehat{\rho}_k(X) = Q_k(X), \tag{33}$$

$$\Delta_k(X) = \left\lfloor \frac{X_k^{(R)} + X_{k-1}^{(Q)}}{M_k} \right\rfloor. \tag{34}$$

Taking into account (29), we have that $\Delta_k(X) \le 1$, that is $\Delta_k(X) \in \mathbb{Z}_2 = \{0,1\}$.

We will call the integers $\widehat{\rho}_k(X)$ and $\Delta_k(X)$ the inexact rank and the rank correction, respectively.

The reasons stated above allow us to formulate the following theorem.

Theorem 1 (**About the rank of an RNS number**) *Let an arbitrary RNS be defined as an ordered set of k pairwise relatively prime odd moduli* $m_1, m_2, \ldots, m_k$

($m_l \geq l-1$, $l = 1, 2, \ldots, k$, $k \geq 2$). Then the rank $\rho_k(X)$ of the integer $X = (\chi_1, \chi_2, \ldots, \chi_k)$ $\left(X \in \mathbb{Z}_{M_k}\right)$ can be computed as follows:

$$\rho_k(X) = \widehat{\rho}_k(X) + \Delta_k(X),$$

where

$$\widehat{\rho}_k(X) = \left\lfloor \frac{1}{m_k} \sum_{i=1}^{k} R_{i,k}(\chi_i) \right\rfloor$$

and

$$R_{i,k}(\chi_i) = \left\lfloor \frac{m_k \left| M_{i,k}^{-1} \chi_i \right|_{m_i}}{m_i} \right\rfloor = \left| -\frac{\left| M_{i,k-1}^{-1} \chi_i \right|_{m_i}}{m_i} \right|_{m_k} \quad (i \neq k),$$

$$R_{k,k}(\chi_k) = \chi_{k,k} = \left| M_{k-1}^{-1} \chi_k \right|_{m_k},$$

the correction $\Delta_k(X)$ is a two-valued number that only takes on the values 0 *or* 1.

As it follows from Theorem 1, the inexact rank $\widehat{\rho}_k(X) \in \mathbb{Z}_k$ is equal to the number of overflows, which occur when computing the sum of k residues $R_{1,k}(\chi_1)$, $R_{2,k}(\chi_2)$, $\ldots$, $R_{k,k}(\chi_k)$ with respect to the kth modulus m_k. Thus, the inexact rank $\widehat{\rho}_k(X)$ can be computed quickly and easily.

A different situation takes place in the case of the rank correction $\Delta_k(X)$ as far as the calculation of its value has a more massive computational complexity than the estimation of the inexact rank $\widehat{\rho}_k(X)$.

As follows from the equations (29) and (34), the rank correction $\Delta_k(X)$ is, in essence, an overflow flag

$$\Delta_k(X) = \begin{cases} 0, \text{ if } X_k^{(R)} + X_{k-1}^{(Q)} < M_k \\ \\ 1, \text{ if } X_k^{(R)} + X_{k-1}^{(Q)} \geq M_k. \end{cases} \tag{35}$$

To get $\Delta_k(X)$, one must first calculate the k-tuple mixed-radix representations $\left\langle x_k^{(R)}, x_{k-1}^{(R)}, \ldots, x_1^{(R)} \right\rangle$ and $\left\langle x_k^{(Q)}, x_{k-1}^{(Q)}, \ldots, x_1^{(Q)} \right\rangle$ of the integers $X_k^{(R)}$ and $X_{k-1}^{(Q)}$, respectively. The digits $x_l^{(R)}$ and $x_l^{(Q)}$ $(l = 2, 3, \ldots, k)$ are calculated by taking the sum of l residues $R_{1,l}(\chi_1)$, $R_{2,l}(\chi_2)$, $\ldots$, $R_{l,l}(\chi_l)$ modulo m_l according to (22)–(25), followed by the necessity to determine whether the sum of two MRS-numbers $X_k^{(R)}$ and $X_{k-1}^{(Q)}$ overruns the RNS dynamic range $\mathbb{Z}_{M_k}$.

To illustrate the calculation of the rank in conventional non-redundant RNS based on the mentioned above, we present a first numerical example.

Example 1 Let us consider an RNS with the moduli $m_1 = 5, m_2 = 7, m_3 = 9$ and $m_4 = 11$. Suppose we wish to calculate the rank $\rho_k(X)$ of the number $X = 2$ having the residue code $(\chi_1, \chi_2, \chi_3, \chi_4) = (2, 2, 2, 2)$.

The primitive constants in the given RNS are

$$M_4 = 3465,\ M_3 = 315,\ M_2 = 35,\ M_1 = 5,\ M_0 = 1.$$
$$M_{1,4} = 693,\ M_{2,4} = 495,\ M_{3,4} = 385,\ M_{4,4} = M_3 = 315,$$
$$\left|M_{1,4}^{-1}\right|_5 = \left|693^{-1}\right|_5 = 2,\ \left|M_{2,4}^{-1}\right|_7 = \left|495^{-1}\right|_7 = 3,$$
$$\left|M_{3,4}^{-1}\right|_9 = \left|385^{-1}\right|_9 = 4,\ \left|M_{4,4}^{-1}\right|_{11} = \left|315^{-1}\right|_{11} = 8,$$
$$\left|5^{-1}\right|_{11} = 9,\ \left|7^{-1}\right|_{11} = 8,\ \left|9^{-1}\right|_{11} = 5,\ \left|M_3^{-1}\right|_{11} = \left|315^{-1}\right|_{11} = 8.$$
$$M_{1,3} = 63,\ M_{2,3} = 45,\ M_{3,3} = M_2 = 35,$$
$$\left|M_{1,3}^{-1}\right|_5 = \left|63^{-1}\right|_5 = 2,\ \left|M_{2,3}^{-1}\right|_7 = \left|45^{-1}\right|_7 = 5,\ \left|M_{3,3}^{-1}\right|_9 = \left|35^{-1}\right|_9 = 8,$$
$$\left|5^{-1}\right|_9 = 2,\ \left|7^{-1}\right|_9 = 4;\ \left|M_2^{-1}\right|_9 = \left|35^{-1}\right|_9 = 8.$$
$$M_{1,2} = 7,\ M_{2,2} = M_1 = 5,$$
$$\left|M_{1,2}^{-1}\right|_5 = \left|7^{-1}\right|_5 = 3,\ \left|M_{2,2}^{-1}\right|_7 = \left|5^{-1}\right|_7 = 3.$$
$$M_{1,1} = 1.$$

First, according to (24) and (25), we calculate the residue $R_{1,1}(\chi_1)$ and the sets of residues $\langle R_{1,l}(\chi_1), R_{2,l}(\chi_2), \ldots, R_{l,l}(\chi_l)\rangle$ $(l = 2, 3, 4)$:

$$R_{1,1}(\chi_1) = 2;$$
$$R_{1,2}(\chi_1) = |-|1 \cdot 2|_7 \cdot 3|_7 = 1,$$
$$R_{2,2}(\chi_2) = |3 \cdot 2|_7 = 6.$$
$$R_{1,3}(\chi_1) = \left|-|3 \cdot 2|_5 \cdot 2\right|_9 = 7,$$
$$R_{2,3}(\chi_2) = |-|3 \cdot 2|_7 \cdot 4|_9 = 3,$$
$$R_{3,3}(\chi_3) = |8 \cdot 2|_9 = 7.$$
$$R_{1,4}(\chi_1) = \left|-|2 \cdot 2|_5 \cdot 9\right|_{11} = 8,$$
$$R_{2,4}(\chi_2) = |-|5 \cdot 2|_7 \cdot 8|_{11} = 9,$$
$$R_{3,4}(\chi_3) = \left|-|8 \cdot 2|_9 \cdot 5\right|_{11} = 9,$$
$$R_{4,4}(\chi_4) = |8 \cdot 2|_{11} = 5.$$

Thus, as a result, we have

$$\langle R_{1,1}(\chi_1)\rangle = \langle 2\rangle,$$
$$\langle R_{1,2}(\chi_1), R_{2,2}(\chi_2)\rangle = \langle 1, 6\rangle,$$
$$\langle R_{1,3}(\chi_1), R_{2,3}(\chi_2), R_{3,3}(\chi_3)\rangle = \langle 7, 3, 7\rangle,$$
$$\langle R_{1,4}(\chi_1), R_{2,4}(\chi_2), R_{3,4}(\chi_3), R_{4,4}(\chi_4)\rangle = \langle 8, 9, 9, 5\rangle.$$

Further, we compute the sum of the corresponding set of residues modulo m_l and the number of occurred overflows according to (22) and (23) for $l = 2, 3, 4$. Taking

into account that $R_1(X) = R_{1,1}(\chi_1)$ and $Q_1(X) = 0$, we have

$$\begin{aligned} R_1(X) &= 2, \\ R_2(X) &= |1+6|_7 = |7|_7 = 0, \\ R_3(X) &= |7+3+7|_9 = |17|_9 = 8, \\ R_4(X) &= |8+9+9+5|_{11} = |31|_{11} = 9 \end{aligned}$$

and

$$\begin{aligned} Q_1(X) &= 0, \\ Q_2(X) &= \lfloor (1+6)/7 \rfloor = \lfloor 7/7 \rfloor = 1; \\ Q_3(X) &= \lfloor (7+3+7)/9 \rfloor = \lfloor 17/9 \rfloor = 1; \\ Q_4(X) &= \lfloor (8+9+9+5)/11 \rfloor = \lfloor 31/11 \rfloor = 2. \end{aligned}$$

Therefore, according to the equations (26)–(28), we obtain the inexact rank

$$\widehat{\rho}_4(X) = Q_4(X) = 2$$

as well as the MRS-integers

$$X_4^{(R)} = \langle 9, 8, 0, 2 \rangle$$

and

$$X_3^{(Q)} = \langle 1, 1, 0, 0 \rangle.$$

Then, by taking the sum of the numbers $X_4^{(R)}$ and $X_3^{(Q)}$ in the MRS, we get the positional mixed-radix representation of the number X:

$$X = |X_4|_{M_4} = \left| X_4^{(R)} + X_3^{(Q)} \right|_{M_4} = \langle 0, 0, 0, 2 \rangle$$

as well as the correction $\Delta_4(X) = 1$ (see (35)).

Finally, we have

$$\rho_4(X) = \widehat{\rho}_4(X) + \Delta_4(X) = 2 + 1 = 3.$$

To verify the obtained result, taking into account that the normalized residues $\chi_{l,4}$ $(l = 1, 2, 3, 4)$ (see (5)) in the CRT-form (7) take on the following values

$$\begin{aligned} \chi_{1,4} &= \left| M_{1,4}^{-1} \chi_1 \right|_5 = |2 \cdot 2|_5 = 4, \\ \chi_{2,4} &= \left| M_{2,4}^{-1} \chi_2 \right|_7 = |3 \cdot 2|_7 = 6, \\ \chi_{3,4} &= \left| M_{3,4}^{-1} \chi_3 \right|_9 = |4 \cdot 2|_9 = 8; \\ \chi_{4,4} &= \left| M_3^{-1} \chi_4 \right|_{11} = |8 \cdot 2|_{11} = 5 \end{aligned}$$

we find

$$\begin{aligned} X &= \sum_{i=1}^{4} M_{i,4}\chi_{i,4} - \rho_4(X) M_4 \\ &= 693 \cdot 4 + 495 \cdot 6 + 385 \cdot 8 + 315 \cdot 5 - 3 \cdot 3465 \\ &= 10397 - 10395 = 2. \end{aligned}$$

Equations (21)–(28) show that the calculation of the rank $\rho_k(X)$ in the conventional non-redundant RNS is reduced to a summation of the sets of small residues with respect to the moduli $m_1, m_2, \ldots, m_k$. These calculations can be implemented within the framework of tabular computing structures. Also, along with the calculation of the rank $\rho_k(X)$, we obtain the positional representation of the number X in the MRS. Thus, the computational complexity of calculating the correction $\Delta_k(X)$ (and, hence, the rank $\rho_k(X)$) coincides with the computational complexity of converting the residue code into the MRS representation.

From the above, it follows that the main computational cost is the calculation of the rank correction $\Delta_k(X)$. As follows from the equations (21)–(34), during the calculation of $\Delta_k(X)$ we can perform the independent and concurrent summations of the sets of residues $\langle R_{1,l}(\chi_1), R_{2,l}(\chi_2), \ldots, R_{l,l}(\chi_l)\rangle$ with respect to the corresponding modulus m_l $(l = 2, 3, \ldots, k)$.

Now let us evaluate the computational costs of the rank calculation. The number of required single one-input lookup tables to store the set of residues $R_{1,l}(\chi_1), R_{2,l}(\chi_2), \ldots, R_{l,l}(\chi_l)$ (see (24) and (25)) is equal to $n_{LUT}(l) = l$, while the bit length of recorded constants is $\lceil \log_2 m_l \rceil$ $(l = 2, 3, \ldots, k)$. It should be noted that $n_{LUT}(1) = 0$ since in the case of the modulus m_1 we have $S_1(X) = \chi_1$. Then, the total number of required lookup tables

$$N_{LUT} = \sum_{l=2}^{k} n_{LUT}(l) = \frac{k^2 + k - 2}{2}. \tag{36}$$

The summation of the set of residues $\langle R_{1,l}(\chi_1), R_{2,l}(\chi_2), \ldots, R_{l,l}(\chi_l)\rangle$ modulo m_l along with the counting of the number of occurred overflows requires $n_{MO}(l) = l-1$ modular operations $(l = 2, 3, \ldots, k)$. Taking into account that $x_1^{(Q)} = x_2^{(Q)} = 0$, the summation of MRS-numbers $X_k^{(R)}$ and $X_{k-1}^{(Q)}$ along with the detection of overflow flag $\Delta_k(X)$ (see (35)) and the final correction of inexact rank $\widehat{\rho_k}(X)$ requires $2(k-2)$ modular addition operations. Hence, the total number of required modular operations

$$N_{MO} = \sum_{l=2}^{k} n_{MO}(l) + 2(k-2) = \frac{k^2 + 5k - 10}{2}. \tag{37}$$

Therefore, the process of calculating the rank $\rho_k(X)$ requires $O(k^2)$ modular operations so that it can become computationally expensive for large values of k. Thus, for effective implementation of non-modular operations based on the CRT, one needs to speed up and optimize the calculation of the rank correction $\Delta_k(X)$.

5 The Relationship Between the Rank Correction and the Parity of an RNS Number

The fact that $\Delta_k(X) \in \{0, 1\}$ (see Theorem 1) allows us to consider it as the residue modulo 2.

According to the CRT-form (7), taking into account (32), we have

$$X = \sum_{i=1}^{k} M_{i,k}\chi_{i,k} - (\widehat{\rho}_k(X) + \Delta_k(X))\, M_k = \widehat{X} - \Delta_k(X)\, M_k, \tag{38}$$

where

$$\widehat{X} = \sum_{i=1}^{k} M_{i,k}\chi_{i,k} - \widehat{\rho}_k(X)\, M_k. \tag{39}$$

Since the RNS moduli $m_1, m_2, \ldots, m_k$ are relatively prime odd integers, then $|m_i|_2 = 1$ $(i = 1, 2, \ldots, k)$, and correspondingly, $|M_k|_2 = 1$. Therefore,

$$|X|_2 = \left|\widehat{X} - \Delta_k(X)\, M_k\right|_2 = \left|\left|\widehat{X}\right|_2 - \Delta_k(X)\right|_2. \tag{40}$$

Thus, for the rank correction $\Delta_k(X)$ the following equality is true

$$\Delta_k(X) = \left|\left|\widehat{X}\right|_2 - |X|_2\right|_2. \tag{41}$$

Hence, the calculation of the rank correction $\Delta_k(X) \in \{0, 1\}$ can be reduced to a parity comparison of the numbers X and $\widehat{X}$, that is:

$$\Delta_k(X) = \begin{cases} 0, \text{ if } |X|_2 = |\widehat{X}|_2, \\ \\ 1, \text{ if } |X|_2 \neq |\widehat{X}|_2. \end{cases} \tag{42}$$

Taking into account the fact that $|M_{i,k}|_2 = 1$ $(i = 1, 2, \ldots, k)$, the calculation of the parity of a number $\widehat{X}$ according to (39) is reduced to performing trivial operations modulo 2, such as

$$\left|\widehat{X}\right|_2 = \left|\sum_{i=1}^{k} \left|\chi_{i,k}\right|_2 - |\widehat{\rho}_k(X)|_2\right|_2 = \left|\sum_{i=1}^{k} \chi_{i,k}^{(0)} - \widehat{\rho}_k^{(0)}\right|_2, \tag{43}$$

where

$$\chi_{i,k}^{(0)} = \left|\chi_{i,k}\right|_2 = \left|\left|M_{i,k-1}^{-1}\chi_i\right|_{m_i}\right|_2 \tag{44}$$

and

$$\widehat{\rho}_k^{(0)} = |\widehat{\rho}_k(X)|_2 \tag{45}$$

are the least-significant bits of the binary representation of the normalized residue $\chi_{i,k}$ $(i = 1, 2, \ldots, k)$ and the inexact rank $\widehat{\rho}_k(X)$, respectively.

As follows from Theorem 1, the inexact rank $\widehat{\rho}_k(X)$, and hence, its parity $\widehat{\rho}_k^{(0)}$ is calculated quickly during the summation of corresponding residues modulo m_k. Thus, the calculation of the parity of the number $\widehat{X}$ according to (43) does not cause difficulties.

Regarding the parity check of the number X, this operation in RNS arithmetic refers to complicated non-modular operations requiring high computational costs. In

conventional non-redundant RNS, the computational complexity of this operation is comparable to the computational complexity of RNS to MRS conversion.

Therefore, for fast calculation of the correction $\Delta_k(X)$ according to (42), the complexity of computing parity $|X|_2$ can be overcome by adding the redundant residue modulo 2 into conventional non-redundant residue code $(\chi_1, \chi_2, \ldots, \chi_k)$ of the number X. Thus, the case in question is the use of redundant RNS, in which only one extra bit is added to the primary residue code.

6 Fast Calculation of the Rank in Minimally Redundant RNS

As is known, the use of code redundancy often allows improving the arithmetic and other properties of numerical systems, including an RNS. In the first place, the redundant RNS are of interest concerning error checking and failure recovery properties [18–20]. On the other hand, the use of RNS representations with redundant residues is an established method that allows gaining speed and cost benefits when performing various non-modular operations [13, 23, 25, 29].

The proposed redundant residue coding assumes the extension of the modular code $(\chi_1, \chi_2, \ldots, \chi_k)$ of the number X in the conventional non-redundant RNS with the k-moduli set $\{m_1, m_2, \ldots, m_k\}$ and the dynamic range $\mathbb{Z}_{M_k}$ by the redundant residue $\chi_0 = |X|_{m_0}$ with respect to the extra modulus $m_0 = 2$, i.e., by adding the parity of the number X to its residue representation. Thus, in the redundant RNS the number $X \in \mathbb{Z}_{M_k}$ is represented by its redundant residue code $(\chi_0, \chi_1, \ldots, \chi_k)$.

This modular code is, of course, minimally redundant since the total code length increases by only one bit. The redundancy is estimated by the following index

$$R_{RRNS} = 1 - \frac{\log_2 M_k}{\log_2 (m_0 M_k)} = \frac{1}{1 + \log_2 M_k}. \tag{46}$$

From (46), it follows that the code redundancy decreases as the number k of RNS moduli, and, consequently, the dynamic range $\mathbb{Z}_{M_k}$ increases.

In this case, the main advantage of minimally redundant RNS in comparison with non-redundant RNS consists of a significantly simplified calculation of the rank correction $\Delta_k(X)$ and, accordingly, the rank $\rho_k(X)$. That leads in turn to optimization and speed-up of non-modular operations in terms of the CRT. The following statement reveals the essence of such an approach.

Theorem 2 (**About the rank of a number in minimally redundant RNS**) *Let a minimally redundant RNS be defined by an ordered set of k pairwise relatively prime odd moduli* $m_1, m_2, \ldots, m_k$ *and excess modulus* m_0, $(m_0 = 2;\ m_l \geq l - 1, l = 1, 2, \ldots, k;\ k \geq 2)$. *Then the rank* $\rho_k(X)$ *of the integer* $X = (\chi_0, \chi_1, \ldots, \chi_k)$ *from the dynamic range* $\mathbb{Z}_{M_k}$ *can be computed as follows:*

$$\rho_k(X) = \widehat{\rho}_k(X) + \delta_k(X), \tag{47}$$

where $\widehat{\rho}_k(X)$ *is calculated according to Theorem 1,*

$$\delta_k(X) = |\chi_0 + \widehat{\chi}_0|_2, \tag{48}$$

while

$$\widehat{\chi}_0 = \left| \sum_{i=1}^{k} \chi_{i,k}^{(0)} + \widehat{\rho}_k^{(0)} \right|_2,$$

$\chi_{i,k}^{(0)}$ *and* $\widehat{\rho}_k^{(0)}$ *are calculated according to* (44) *and* (45)*, respectively.*

The proof of this theorem follows directly from equations (38)–(45), taking into account the fact that $|a-b|_2 = |a+b|_2$ $(a, b \in \mathbb{Z})$.

Following Theorem 2, the transition from non-redundant to minimally redundant modular coding allows replacing in (32) the rank correction $\Delta_k(X)$ requiring time-consuming calculations by a trivially calculated two-value attribute $\delta_k(X) \in \{0, 1\}$.

At the same time, as can be easily seen, the calculations according to (47) are reduced to a set of quickly implemented modular operations modulo m_k (the calculation of the inexact rank $\widehat{\rho}_k(X)$) and modulo m_0 (the calculation of the correction $\delta_k(X)$). Also, as follows from the equation (48), the calculation of $\widehat{\rho}_k(X)$ can be performed in parallel with the calculation of the value

$$\delta_0 = \left| \chi_0 + \sum_{i=1}^{k} \chi_{i,k}^{(0)} \right|_2,$$

whose correction by using $\widehat{\rho}_k^{(0)}$ in the final stage of the calculation gives us the resulting value of $\delta_k(X)$.

Thus, in the minimally redundant RNS the calculation of the rank $\rho_k(X)$ is carried out exceptionally merely within the scope of the modular computational process, which can be easily implemented, for example, by using high-performance lookup-table methods in $T_k = \lceil \log_2 k \rceil$ modular clock cycles.

Because of the use of minimum-redundancy residue code, the complexity of calculating the rank $\rho_k(X)$ is significantly reduced in comparison with non-redundant analogs. First of all, this is due to the need to perform the modular addition operations only with respect to the one kth modulus m_k instead of the k-moduli set $\{m_1, m_2, \ldots, m_k\}$ in the case of conventional non-redundant RNS. Therefore, the computational complexity decreases from $O\left(k^2\right)$ to $O(k)$ modular operations.

Accordingly, the number of required lookup tables to store the sets of residues is also reduced from $O\left(k^2\right)$ to $O(k)$.

The corresponding costs for calculating the rank $\rho_k(X)$ in minimally redundant RNS are

$$N_{LUT}^{(R)} = k \tag{49}$$

lookup tables and

$$N_{MO}^{(R)} = k \tag{50}$$

modular addition operations, including the correction of the inexact rank $\widehat{\rho}_k(X)$ by two-valued correction $\delta_k(X)$.

Here it is assumed that to the ith lookup table one records a pair of residues $\left\langle R_{i,k}(\chi_i), \chi_{i,k}^{(0)} \right\rangle$, which calculated according to equations (18), (19), and (44), $R_{i,k}(\chi_i) \in \mathbb{Z}_{m_k}$, $\chi_{i,k}^{(0)} \in \mathbb{Z}_2$ $(i = 1, 2, \ldots, k)$. Thus, the bit-width of the used lookup tables is $l = \lceil \log_2 m_k \rceil + 1$.

As can be seen, the use of minimally redundant RNS results in a significant reduction of computational costs both in terms of required modular addition operations and lookup tables relative to the non-redundant RNS. The reduction factors of the computational complexity of calculating the rank $\rho_k(X)$ in minimally redundant RNS compared to conventional non-redundant RNS are represented by the following fractions

$$C_{LUT} = \frac{N_{LUT}}{N_{LUT}^{(R)}} = \frac{k^2 + k - 2}{2k} \tag{51}$$

for the number of required lookup tables (see (36) and (49)), and

$$C_{MO} = \frac{N_{MO}}{N_{MO}^{(R)}} = \frac{k^2 + 5k - 10}{2k} \tag{52}$$

for the number of modular addition operations (see (37) and (50)).

Equations (51) and (52) show that the reduction factors C_{LUT} and C_{MO} increase with the number k of non-redundant moduli $m_1, m_2, \ldots, m_k$ asymptotically approaching the threshold $k/2$.

For example, take the number k of non-redundant RNS moduli as a multiple of 5:

$$k = 5,\ 10,\ 15,\ 20,\ 25,\ 30.$$

Then for the reduction factors (51) and (52), we have the following values depending on the number of modules k, respectively:

$$C_{LUT} = 2.8,\ 5.4,\ 7.93,\ 10.45,\ 12.96,\ 15.47$$

and

$$C_{MO} = 4.0,\ 7.0,\ 9.67,\ 12.25,\ 14.8,\ 17.33.$$

Thus, the proposed approach for the rank calculation in minimally redundant RNS with the length k of primary residue code from 5 to 30 digits compared to conventional non-redundant RNS allows us to reduce the computational costs by 2.8 – 15.47 times in terms of lookup table memory and by 4.0 – 17.33 times in terms of required modular addition operations.

Below we give the numerical examples, which demonstrate the calculation of the integer value of the number $X = (\chi_1, \chi_2, \ldots, \chi_k)$ based on the CRT-form (7) in the proposed minimally redundant RNS.

Let us consider the RNS with the moduli $m_1 = 5, m_2 = 7, m_3 = 9$, and $m_4 = 11$ from Example 1, taking into account the excess modulus $m_0 = 2$.

Example 2 Suppose we wish to calculate the rank $\rho_k(X)$ of the number $X = 2$ having the minimally redundant residue code (0, 2, 2, 2, 2).

First, the residues $R_{i,4}(\chi_i)$ and $\chi_{i,4}^{(0)}$ $(i = 1, 2, 3, 4)$ are calculated according to (18), (19), and (44), respectively. As a result, we have

$$\begin{aligned} R_{1,4}(\chi_1) &= \left|-|2\cdot 2|_5\cdot 9\right|_{11} = 8,\\ R_{2,4}(\chi_2) &= |-|5\cdot 2|_7\cdot 8|_{11} = 9,\\ R_{3,4}(\chi_3) &= \left|-|8\cdot 2|_9\cdot 5\right|_{11} = 9,\\ R_{4,4}(\chi_4) &= |2\cdot 8|_{11} = 5 \end{aligned}$$

and

$$\begin{aligned} \chi_{1,4}^{(0)} &= \left||2\cdot 2|_5\right|_2 = 0,\\ \chi_{2,4}^{(0)} &= ||3\cdot 2|_7|_2 = 0,\\ \chi_{3,4}^{(0)} &= \left||4\cdot 2|_9\right|_2 = 0,\\ \chi_{4,4}^{(0)} &= ||8\cdot 2|_{11}|_2 = 1. \end{aligned}$$

It can be noted that the normalized residues $\chi_{l,4}$ $(l = 1, 2, 3, 4)$ (see (5)) take on the values 4, 6, 8, 5, respectively.

Further, we find the number of occurred overflows when summing the residues of the set $\langle R_{1,4}(\chi_1), R_{2,4}(\chi_2), R_{3,4}(\chi_3), R_{4,4}(\chi_4)\rangle = \langle 8, 9, 9, 5\rangle$ modulo $m_4 = 11$, i.e., we calculate the inexact rank

$$\widehat{\rho}_4(X) = \lfloor(8+9+9+5)/11\rfloor = \lfloor 31/11\rfloor = 2.$$

Therefore,

$$\widehat{\rho}_4^{(0)} = |\widehat{\rho}_4(X)|_2 = 0.$$

Then, taking into account the fact that $\chi_0 = 0$, according to (48), we calculate two-valued correction

$$\delta_4(X) = |0 + (0+0+0+1) + 0|_2 = |1|_2 = 1.$$

As a result, we get the rank

$$\rho_4(X) = \widehat{\rho}_4(X) + \delta_4(X) = 2 + 1 = 3.$$

To verify the obtained result, using the CRT-form (7), by analogy with Example 1, we find

$$\begin{aligned} X &= \sum_{i=1}^{4} M_{i,4}\chi_{i,4} - \rho_4(X)\, M_4\\ &= 693\cdot 4 + 495\cdot 6 + 385\cdot 8 + 315\cdot 5 - 3\cdot 3465\\ &= 10397 - 10395 = 2. \end{aligned}$$

Example 3 Suppose we wish to calculate the rank $\rho_k(X)$ of the number $X = M_4 - 2 = 3463$, which is represented by the minimum-redundancy residue code $(1, 3, 5, 7, 9)$.

Similar to Example 2, we compute

$$\begin{aligned} R_{1,4}(\chi_1) &= \left|-|2\cdot 3|_5\cdot 9\right|_{11} = 2,\\ R_{2,4}(\chi_2) &= |-|5\cdot 5|_7\cdot 8|_{11} = 1,\\ R_{3,4}(\chi_3) &= \left|-|8\cdot 7|_9\cdot 5\right|_{11} = 1,\\ R_{4,4}(\chi_4) &= |9\cdot 8|_{11} = 6 \end{aligned}$$

and

$$\begin{aligned} \chi_{1,4}^{(0)} &= \left||2\cdot 3|_5\right|_2 = 1,\\ \chi_{2,4}^{(0)} &= ||3\cdot 5|_7|_2 = 1,\\ \chi_{3,4}^{(0)} &= \left||4\cdot 7|_9\right|_2 = 1,\\ \chi_{4,4}^{(0)} &= ||8\cdot 9|_{11}|_2 = 0. \end{aligned}$$

As can be seen, the normalized residues $\chi_{l,4}$ $(l = 1, 2, 3, 4)$ (see (5)) take on the values 1, 1, 1, 6, respectively.

We find the inexact rank $\widehat{\rho}_4(X)$ by counting the occurred overflows when summing the residues of the set $\langle R_{1,4}(\chi_1), R_{2,4}(\chi_2), R_{3,4}(\chi_3), R_{4,4}(\chi_4)\rangle = \langle 2, 1, 1, 6\rangle$ modulo $m_4 = 11$, i.e.,

$$\widehat{\rho}_4(X) = \lfloor(2+1+1+6)/11\rfloor = \lfloor 10/11\rfloor = 0.$$

Hence,

$$\widehat{\rho}_4^{(0)} = |\widehat{\rho}_4(X)|_2 = 0.$$

Since $\chi_0 = 1$, it follows from (48) that the correction

$$\delta_4(X) = |1 + (1+1+1+0) + 0|_2 = |4|_2 = 0.$$

Thus, in this case

$$\rho_4(X) = \widehat{\rho}_4(X) = 0.$$

To verify the obtained value, according to (7), we get

$$\begin{aligned} X &= \sum_{i=1}^{4} M_{i,4}\chi_{i,4} - \rho_4(X)M_4\\ &= 693\cdot 1 + 495\cdot 1 + 385\cdot 1 + 315\cdot 6 - 0\cdot 3465\\ &= 3463. \end{aligned}$$

As it follows from Example 2 and Example 3, the use of minimally redundant RNS allows us to significantly optimize the calculation of the rank $\widehat{\rho}_k(X)$, and correspondingly, the execution of non-modular procedures based on the use of the CRT. First of all, it caused by the utmost simplicity of forming the two-value characteristic $\delta_k(X)$, as well as the modular structure of the basic equation for calculating the inexact rank $\widehat{\rho}_k(X)$ (see Theorem 1). This circumstance allows us to radically simplify the calculation of the rank $\rho_k(X)$ and, consequently, to construct faster and optimal in cost variants of RNS arithmetic.

Therefore, the proposed version of minimally redundant RNS takes priority compared to non-redundant RNS concerning the optimization of the implementation of non-modular procedures on the base of the CRT algorithm.

7 Conclusions

In this paper, we have shown that the use of minimum-redundancy residue code can allow the construction of efficient RNS implementations based on the CRT due to optimizing the calculation of the rank $\rho_k(X)$, a principal positional characteristic in RNS arithmetic.

We investigated the structure of the rank $\rho_k(X)$ and proposed a novel method for calculating the correction $\Delta_k(X)$ to the inexact rank $\widehat{\rho}_k(X)$. This method is based on the fact that the correction $\Delta_k(X)$ is a two-value number ($\Delta_k(X) \in \{0, 1\}$) in the case when the RNS moduli $m_1, m_2, \ldots, m_k$ are chosen according to the rule of Theorem 1. That allows us to reduce the computational complexity of calculating the rank $\rho_k(X)$ from $O\left(k^2\right)$ to $O(k)$ owing to introducing the minimum redundancy of the residue code by adding the redundant residue modulo $m_0 = 2$, which, in essence, is the parity of the RNS number X.

The reduction factors of the computational complexity of the rank calculation in minimally redundant RNS compared to non-redundant RNS increase with the number k of RNS moduli asymptotically approaching the threshold $k/2$. For example, the use of minimally redundant RNS with the length k of primary residue code from 5 to 30 digits enables us to reduce the computational costs by 2.8–15.47 times in terms of lookup table memory and by 4.0 – 17.33 times in terms of required modular addition operations.

Therefore, the proposed minimally redundant RNS takes priority in the field of rapid calculations, especially in the case of the large RNS dynamic range, for example, for implementing various complicated algorithms of digital signal processing and cryptography. Thus, the examined approach to implementation of the CRT algorithm in minimally redundant RNS coincides with the vector of the development of modern methods and algorithms of high-performance and high-accuracy computing.

References

1. Abtahi, M.: Core function of an RNS number with no ambiguity. Comput. Math. Appl. **50**(3–4), 459–470 (2005)

2. Akkal, M., Siy, P.: A new mixed radix conversion algorithm MRC-II. J. Syst. Archit. **53**(9), 577–586 (2007)
3. Akushskii, I.Y., Juditskii, D.I.: Machine Arithmetic in Residue Classes. Sov. Radio, Moscow (1968)
4. Akushskii, I.Y., Burcev V.M., Pak, I.T.: A new positional characteristic of non-positional codes and its application. In: Coding Theory and the Optimization of Complex Systems. Nauka, Alma-Ata (1977)
5. Amerbayev, V.M.: Theoretical Foundations of Machine Arithmetic. Nauka, Alma-Ata (1976)
6. Burton, D.M.: Elementary Number Theory. Allyn and Bacon, Boston (1980)
7. Dimauro, G., Impedovo, S., Pirlo, G.: A new magnitude function for fast numbers comparison in the residue number system. Microprocess. Microprogr. **35**(1–5), 97–104 (1992)
8. Dimauro, G., Impedovo, S., Pirlo, G., Salzo, A.: RNS architectures for the implementation of the 'diagonal function'. Inf. Process. Lett. **73**(5–6), 189–198 (2000)
9. Dimauro, G., Impedovo, S., Modugno, R., et al.: Residue-to-binary conversion by the quotient function. IEEE Trans. Circ. Syst. II Analog and Digital Signal Process. **50**(8), 488–493 (2003)
10. Gonnella, J.: The application of core functions to residue number system. IEEE Trans. Signal Process. **39**(1), 69–75 (1991)
11. Hardy, G.H., Wright, E.M.: An Introduction to the Theory of Numbers, 6th edn. Oxford University Press, Ely House, London (2008)
12. Huang, C.H.: Fully parallel mixed-radix conversion algorithm for residue number applications. IEEE Trans. Comput. **32**(4), 398–402 (1983)
13. Jenkins, W.K., Etzel, M.H.: Special properties of complement codes for redundant residue number system. Proc. IEEE **69**(1), 132–133 (1981)
14. Knuth, D.E.: The art of computer programming, 3rd edn. In: Seminumerical Algorithms, vol. 2. Addison-Wesley Longman Publishing Co., Boston (1997)
15. Kolyada, A.A., Selyaninov, M.Y.: Generation of integral characteristics of symmetric-range residue codes. Cybern. Syst. Anal. **22**(4), 431–437 (1986)
16. Kong, Y., Asif, S., Khan, M.A.U.: Modular multiplication using the core function in the residue number system. Appl. Algebra Eng. Commun. Comput. **27**(1), 1–16 (2016)
17. Miller, D., Altschul, R.E., King, J.R., Polky, J.N.: Analysis of the residue class core function of Akushskii, Burcev, and Pak. In: Residue Number System Arithmetic: Modern Applications in Digital Signal Processing, pp. 390–401. IEEE Press, Piscataway (1986)
18. Mohan, P.V.A.: Residue Number Systems. Theory and Applications. Springer, Cham (2016)
19. Molahosseini, A., Sousa, L., Chang, C.H.: Embedded Systems Design with Special Arithmetic and Number Systems. Springer, Cham (2017)
20. Omondi, A.R., Premkumar, B.: Residue Number Systems: Theory and Implementation. Imperial College Press, London (2007)
21. Pirlo, G., Impedovo, D.: A new class of monotone functions of the residue number system. Int. J. Math. Models Meth. Appl. Sci. **7**(9), 802–809 (2013)
22. Rao, T.R.N., Trehan, A.K.: Binary logic for residue arithmetic using magnitude index. IEEE Trans. Comput. **19**(8), 752–757 (1970)
23. Sengupta, A., Natarajan, B.: Redundant residue number system based space-time block codes. Phys. Commun. **12**, 1–15 (2014)
24. Shenoy, M.A.P., Kumaresan, R.: A fast and accurate RNS scaling technique for high speed signal processing. IEEE Trans. Acoust. Speech Signal Process. **37**(6), 929–937 (1989)
25. Shenoy, A.P., Kumaresan, R.: Fast base extension using a redundant modulus in RNS. IEEE Trans. Comput. **38**(2), 292–297 (1989)
26. Shoup, V. A Computational Introduction to Number Theory and Algebra, 2nd edn. Cambridge University Press, Cambridge (2005)
27. Soderstrand, M.A., Jenkins, W.K., Jullien, G.A., Taylor, F.J.: Residue Number System Arithmetic: Modern Applications in Digital Signal Processing. IEEE Press, Piscataway (1986)
28. Szabo, N.S., Tanaka, R.I.: Residue Arithmetic and its Application to Computer Technology. McGraw-Hill, New York (1967)
29. Tai, L.C., Chen, C.F.: Technical note. Overflow detection in a redundant residue number system. IEE Proc. E-Comput. Digit. Tech. **131**(3), 97–98 (1984)

30. Tchernykh, A., Babenko, M., Chervyakov, N., et al.: AC-RRNS: anti-collusion secured data sharing scheme for cloud storage. Int. J. Approx. Reason. **102**, 60–73 (2018)
31. Yassine, H.M., Moore, W.R.: Improved mixed-radix conversion for residue number architectures. IEE Proc. G - Circ. Dev. Syst. **138**(1), 120–124 (1991)