Computer Science • 21(2) 2020 https://doi.org/10.7494/csci.2020.21.2.3616

Mikhail Selianinau

EFFICIENT IMPLEMENTATION OF CHINESE REMAINDER THEOREM IN MINIMALLY REDUNDANT RESIDUE NUMBER SYSTEM

Abstract *The Chinese remainder theorem is widely used in many modern computer applications. This paper presents an efficient approach to the calculation of the rank of a number, a principal positional characteristic that is used in the residue number system. The proposed method does not use large modulo addition operations as compared to a straightforward implementation of the Chinese remainder theorem algorithm. The rank of a number is equal to the sum of an inexact rank and a two-valued correction factor that only takes on values of 0 or 1. We propose a minimally redundant residue number system that provides a low computational complexity of the rank calculation. The effectiveness of the novel method is analyzed regarding a conventional non-redundant residue number system. Owing to the extension of the residue code, the complexity of the rank calculation goes down from $O(k^2)$ to $O(k)$ by adding the extra residue modulo 2 (where k equals the number of non-redundant residues).*

Keywords residue number system, Chinese remainder theorem, residue code, rank of a number, positional characteristics

Citation Computer Science 21(2) 2020: 237–252

1. Introduction

Since the mid-1950s, the residue number system (RNS) has attracted the continuous attention of researchers in computer technology, numerical methods, cryptography, communications, digital signal processing, and other fields [2, 3, 23, 24, 27, 31, 32]. An RNS, which is based on the standard topics of abstract algebra and number theory [6, 11], has natural internal parallelism. The main advantage of an RNS is its unique ability to decompose long word-length numbers into a set of independent short word-length residues (which are processed in parallel).

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

The carry-free nature of the modular operation is one of the primary and most attractive features of RNS arithmetic. Modular operation $\circ \in \{+, -, \times\}$ on integers $A = (\alpha_1, \alpha_2, \ldots, \alpha_k)$ and $B = (\beta_1, \beta_2, \ldots, \beta_k)$ is carried out independently for each modulus; i.e., by the following rule:

$$A \circ B = (\alpha_1, \alpha_2, \ldots, \alpha_k) \circ (\beta_1, \beta_2, \ldots, \beta_k) =$$

$$= \left(|\alpha_1 \circ \beta_1|_{m_1}, |\alpha_2 \circ \beta_2|_{m_2}, \ldots, |\alpha_k \circ \beta_k|_{m_k}\right)$$

where $\alpha_i = |A|_{m_i}$ $\beta_i = |B|_{m_i}, i = 1, 2, \ldots, k$.

Thus, the addition, subtraction, and multiplication operations are performed in parallel by decomposition into independent modular channels that correspond to moduli $m_1, m_2, \ldots, m_k$ in RNS arithmetic.

At the same time, the critical problem of an RNS consists of the fact that the integer value of number X depends on all of its residues together. In an RNS, the evaluation of the number value underlies all so-called non-modular operations. The problem of the effective implementation of non-modular operations is constantly receiving considerable attention [7, 19, 23, 24, 27].

Unfortunately, the yet-to-be-resolved problem concerning high-performance computing in an RNS consists of the substantial computational complexity of non-modular operations. It is precisely this fact that explains why the RNS arithmetic is for the most part only used in cases when the modular addition and multiplication operations make up the bulk of the required computation, while the number of non-modular operations is relatively small. This circumstance restricts a broad application of RNS arithmetic to a narrow class of specific tasks.

Therefore, the design of effective methods and algorithms for performing non-modular operations is an urgent problem at the current development stage of RNS arithmetic as well as its application in the field of high-performance computing. In particular within this scope, the main topical task consist of the optimization of non-modular operations from the standpoint of computational complexity, realization time, and residue code redundancy.

As is known, the so-called positional characteristics of the residue code are used for performing non-modular operations. In general, all positional characteristics represent certain numerical functions that, by some method or another, allow for the estimation of the integer value of RNS number X. These characteristics depend on part or all of the residues in k-tuple $(\chi_1, \chi_2, \ldots, \chi_k)$. At the same time, the computational complexity of the calculation of applied positional characteristics ultimately determines the efficiency of the RNS arithmetic constructed on their basis.

The non-modular operations in RNS arithmetic is based on the reverse conversion from residues back to an integer. The canonical conversion technique is a straightforward method based on the Chinese remainder theorem (CRT) [9, 23, 24, 27].

In recent decades, the CRT has been intensively studied, especially in connection with its application in high-performance computing. The applied initial efforts consisted of reducing the computational complexity of the CRT algorithm. There is a sufficiently broad class of specific methods that are allowed to replace a slow addition modulo (which is a significant product of the RNS moduli) by simpler operations. All of these methods are based on the use of various positional characteristics of the residue code. Along with the rank of a number, some of the generally accepted characteristics are core function, interval index, parity, diagonal and quotient functions, etc. [2, 3, 7, 20, 23, 24, 27].

The main drawback of the known approaches to the implementation of non-modular operations is a significant computational complexity of the calculation of the used positional characteristics. Consequently, these methods and algorithms are not commonly suitable for designing efficient variants of RNS arithmetic for high-performance computing (especially for large word-length numbers).

The rank of a number is, in essence, a CRT reconstruction coefficient; it represents a positional characteristic of primary importance in RNS arithmetic. The CRT algorithm finds its application in the most various fields of modern science and high-tech industry; for example, computing, coding, cryptography, digital signal and image processing, communication, etc. As a confirmation of the above, we can cite several recent scientific publications devoted to the use of the CRT; for example, [1, 4, 5, 8, 10, 12–15, 17, 21, 22, 25, 28, 34].

In this paper, we propose a new variant of a minimally redundant RNS as well as an efficient approach that provides a low computational complexity of the rank calculation and allows us to improve the performance of non-modular operations based on the CRT algorithm.

2. Rank of a number

As is known, 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 $M_k = \prod_{i=1}^{k} m_i$ integers according to CRT [18, 30]. At the same time, the following rule is valid:

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

where $M_{i,k} = M_k / m_i$, $\chi_{i,k} = \left| M_{i,k}^{-1} \chi_i \right|_{m_i}$ $(i = 1, 2, \ldots, k)$, $\left| Y^{-1} \right|_m$ denotes the multiplicative inverse of integer Y modulo m. The set $\mathbf{Z}_{M_k} = \{0, 1, \ldots, M_k - 1\}$ is usually used as a dynamic range of the RNS.

Equation (1) shows how the position value of number X is obtained from its residues $\chi_1, \chi_2, \ldots, \chi_k$. The arithmetic requirements for the CRT implementation include:

1) k modular multiplications of small residues $\left| M_{i,k}^{-1} \right|_{m_i}$ and χ_i for the calculation of normalized residue $\chi_{i,k}$;
2) k full non-modular multiplications of $M_{i,k}$ and $\chi_{i,k}$;
3) large word-length addition operations of summands $M_{i,k}\chi_{i,k}$ modulo M_k $(i = 1, 2, \ldots, k)$.

It is clear that, along with the large multiplications in the second step, the primary difficulty consists of the need for the modular reduction concerning potentially large modulus M_k.

Thus, the straightforward application of (1) as a primary form for integer reconstruction by a residue code is time-consuming and practically unacceptable for high-performance computing due to its computational complexity, especially in the case of a large M_k.

At the same time, the relevant form of integer representation that possesses excellent implementing properties can be obtained on the base of (1), allowing us to circumvent the problem of slow addition operations modulo M_k.

As follows from (1), difference:

$$\sum_{i=1}^{k} M_{i,k} \chi_{i,k} - X$$

is a multiple of M_k. Then, to circumvent the problem of slow addition operations modulo M_k, Equation (1) can be rewritten as an exact integer equality

$$X = \sum_{i=1}^{k} M_{i,k} \chi_{i,k} - \rho_k(X) M_k \tag{2}$$

where the unknown $\rho_k(X)$ is a positive integer called a normalized rank (or briefly, rank) of number X [2, 3].

Equation (2) is called the rank form (or CRT-form) of integer X. In essence, rank $\rho_k(X)$ is a CRT reconstruction coefficient indicating how many times the RNS dynamic range M_k was exceeded when converting k-tuple $(\chi_1, \chi_2, \ldots, \chi_k)$ to integer X according to (2).

From (2), it follows that the following inequality holds for rank $\rho_k(X)$:

$$0 \leqslant \rho_k(X) \leqslant k-1$$

since the rank $\rho_k(X)$ is an integer part of a sum of k proper fractions:

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

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

In the RNS, the rank is a positional characteristic of primary importance since all non-modular operations can be implemented on its basis. As is seen, Equation (2) does not contain addition operations modulo M_k. Therefore, the decoding mapping on the base of rank $\rho_k(X)$ is more effective than straightforward implementations of the CRT (1).

The essential content of the rank calculation in a conventional RNS consists of the following theorem, which follows from [7, 19, 20]:

Theorem 1. (About the calculation of the rank of a number) *Let an arbitrary RNS be defined as an ordered set of k pairwise relatively prime moduli $m_1, m_2, \ldots, m_k$ ($m_l \geqslant l-2$, $l = 1, 2, \ldots, k$, $k \geqslant 2$). Then, rank $\rho_k(X)$ of integer $X = (\chi_1, \chi_2, \ldots, \chi_k)$ ($X \in \mathbf{Z}_{M_k}$) can be computed as follows:*

$$\rho_k(X) = \widehat{\rho}_k(X) + \Delta_k(X) \tag{3}$$

where

$$\widehat{\rho}_k(X) = \left\lfloor \frac{1}{m_k} \sum_{i=1}^{k} R_{i,k}(\chi_i) \right\rfloor \tag{4}$$

while

$$R_{i,k}(\chi_i) = \left| -\frac{\left| M_{i,k-1}^{-1}\chi_i \right|_{m_i}}{m_i} \right|_{m_k} \quad (i \neq k) \tag{5}$$

$$R_{k,k}(\chi_k) = \chi_{k,k} = \left| M_{k-1}^{-1}\chi_k \right|_{m_k} \tag{6}$$

and integer $\Delta_k(X)$ is a two-valued number that only takes on values of 0 or 1.

Thus, rank $\rho_k(X)$ is the sum of two small numbers; namely, inexact rank $\widehat{\rho}_k(X)$ and rank correction $\Delta_k(X)$. As follows from Theorem 1 (see (4)), inexact rank $\widehat{\rho}_k(X)$ is equal to the number of occurred overflows when computing the sum of k residues $R_{1,k}(\chi_1)$, $R_{2,k}(\chi_2)$, …, $R_{k,k}(\chi_k)$ modulo m_k. Thus, inexact rank $\widehat{\rho}_k(X)$ is calculated quickly and easily.

At the same time (as follows from [7, 19, 20]), the calculation of rank correction $\Delta_k(X)$ reduced to the independent and concurrent summations of residues $R_{1,l}(\chi_1)$, $R_{2,l}(\chi_2)$, …, $R_{l,l}(\chi_l)$ modulo m_l $(l = 2, 3, \ldots, k)$ along with the counting of the modular overflows, while integers $R_{j,l}(X)$ $(j = 1, 2, \ldots, l)$ are calculated according to (5) and (6) in the case when $k = l$. These modular operations are easily pipelined within the framework of a lookup table computing technique.

The corresponding costs for rank calculation are $N_{MO} = \left(k^2 + 3k - 8\right)/2$ modular addition operations and $N_{LUT} = \left(k^2 + k - 2\right)/2$ lookup tables for storing the required residues. Therefore, the process of calculating $\rho_k(X)$ requires $O\left(k^2\right)$ modular operations so that it can become computationally expensive for large values of k, for example, in the cryptographic applications commonly encountering large word-length numbers (of a 1024 bit-length and higher).

Thus, one needs to optimize and speed up the calculation of rank $\rho_k(X)$ for the efficient implementation of the CRT algorithm as well as the non-modular operations on its basis.

3. Relationship between rank correction $\Delta_k(X)$ and parity of 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 CRT-form (2) and taking into account (3), we have:

$$X = \widehat{X} - \Delta_k(X) M_k \tag{7}$$

where:

$$\widehat{X} = \sum_{i=1}^{k} M_{i,k}\chi_{i,k} - \widehat{\rho}_k(X) M_k \tag{8}$$

Since RNS moduli $m_1, m_2, \ldots, m_k$ are prime integers, then $|M_k|_2 = 1$.

Therefore, taking into account the fact that $|a-b|_2 = |a+b|_2$ $(a, b \in \mathbf{Z})$, we have:

$$|X|_2 = \left|\widehat{X} - \Delta_k(X) M_k\right|_2 = \left|\left|\widehat{X}\right|_2 + \Delta_k(X)\right|_2 \tag{9}$$

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

$$\Delta_k(X) = \left||X|_2 - \left|\widehat{X}\right|_2\right|_2 \tag{10}$$

Hence, the calculation of $\Delta_k(X)$ is reduced to a parity comparison of numbers X and $\widehat{X}$:

$$\Delta_k(X) = \begin{cases} 0, \text{if } |X|_2 = \left|\widehat{X}\right|_2 \\ 1, \text{if } |X|_2 \neq \left|\widehat{X}\right|_2 \end{cases} \tag{11}$$

According to (8) and taking into account the fact that $|M_{i,k}|_2 = 1$ $(i = 1, 2, \ldots, k)$, the calculation of the parity of number $\widehat{X}$ is reduced to performing trivial operations modulo 2:

$$\left|\widehat{X}\right|_2 = \left|\sum_{i=1}^{k} \chi_{i,k}^{(0)} - \widehat{\rho}_k^{(0)}\right|_2 \tag{12}$$

where:

$$\chi_{i,k}^{(0)} = |\chi_{i,k}|_2 = \left|\left|M_{i,k}^{-1}\chi_i\right|_{m_i}\right|_2 \tag{13}$$

and:

$$\widehat{\rho}_k^{(0)} = |\widehat{\rho}_k(X)|_2 \tag{14}$$

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

As follows from Theorem 1, the value of inexact rank $\widehat{\rho}_k(X)$ and, hence, its parity $\widehat{\rho}_k^{(0)}$ is calculated quickly as a result of the summation of corresponding residues modulo m_k (see (4)-(6)). Thus, the calculation of the parity of number $\widehat{X}$ has no difficulties according to (12).

As for the parity check of number X, this operation in an RNS is a complicated non-modular operation requiring high computational costs of the order of $O\left(k^2\right)$ modular operations. Therefore, we must overcome the difficulty with the need to compute parity $|X|_2$ of a number X for the fast calculation of correction $\Delta_k(X)$.

Below, we proposed a computationally efficient approach for the calculation of rank correction $\Delta_k(X)$, which consists of adding redundant modular channel modulo 2 to the used k channels corresponding to primary moduli $m_1, m_2, \ldots, m_k$.

4. Rank calculation in minimally redundant RNS

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

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

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

As can be seen, the code redundancy decreases as the number k of non-redundant moduli m_1, m_2, …, m_k, and, consequently, dynamic range M_k increases.

The main advantage of a minimally redundant RNS as compared to a non-redundant RNS consists of a significantly simplified calculation of rank correction $\Delta_k(X)$ and, accordingly, rank $\rho_k(X)$.

The following statement reveals the essence of such an approach:

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

$$\rho_k(X) = \widehat{\rho}_k(X) + \delta_k(X) \tag{15}$$

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

$$\delta_k(X) = |\chi_0 + \widehat{\chi}_0|_2 \tag{16}$$

$$\widehat{\chi}_0 = \left| \sum_{i=1}^{k} \chi_{i,k}^{(0)} + \widehat{\rho}_k^{(0)} \right|_2 \tag{17}$$

$\chi_{i,k}^{(0)}$ and $\widehat{\rho}_k^{(0)}$ are calculated in accordance with (13) and (14), respectively.

The proof of this theorem follows directly from Equations (7)–(14).

According to Theorem 2, the transition from non-redundant to minimally redundant residue coding allows replacing rank correction $\Delta_k(X)$ requiring time-consuming calculations by a trivially calculated two-value attribute $\delta_k(X) \in \{0, 1\}$ in (3).

At the same time, the calculation of the rank $\rho_k(X)$ are reduced to a set of quickly implemented modular operations modulo m_k (calculation of inexact rank $\widehat{\rho}_k(X)$) and m_0 (estimation of the correction $\delta_k(X)$) according to (15). Also, the calculation of inexact rank $\widehat{\rho}_k(X)$ can be performed in parallel with the calculation of the following value (as follows from [16] and [17]):

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

whose correction using $\widehat{\rho}_k^{(0)}$ at the last stage of the calculations gives us the resulting value of $\delta_k(X)$.

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

The corresponding costs for calculating rank $\rho_k(X)$ in a minimally redundant RNS are $N_{MO}^{(R)} = k$ modular addition operations, including the correction of inexact rank $\widehat{\rho}_k(X)$, and $N_{LUT}^{(R)} = k$ lookup tables for storing the required residues. Here, it is assumed that, to the ith lookup table, one records a pair of residues $\left\langle R_{i,k}(\chi_i), \chi_{i,k}^{(0)}\right\rangle$ that, calculated according to Equations (5), (6), and (13), $R_{i,k}(\chi_i) \in \mathbf{Z}_{m_k}$, $\chi_{i,k}^{(0)} \in \mathbf{Z}_2$ $(i = 1, 2, \ldots, k)$. Thus, the bit-width of the used lookup tables is $l = \lceil \log_2 m_k \rceil + 1$, where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x.

Therefore, the reduction factors of the computational complexity of calculating rank $\rho_k(X)$ in a minimally redundant RNS as compared to a conventional non-redundant RNS are represented by the following fractions:

$$C_{MO} = \frac{N_{MO}}{N_{MO}^{(R)}} = \frac{k^2 + 3k - 8}{2k} \tag{18}$$

for the number of modular addition operations, and:

$$C_{LUT} = \frac{N_{LUT}}{N_{LUT}^{(R)}} = \frac{k^2 + k - 2}{2k} \tag{19}$$

for the number of required lookup tables.

Equations (18) and (19) show that reduction factors C_{MO} and C_{LUT} increase with the number k of non-redundant moduli $m_1, m_2, \ldots, m_k$.

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 reduction factors (18) and (19), we have the following values depending on the number of modules k, respectively:

$$C_{MO} = 3.2,\ 6.1,\ 8.73,\ 11.3,\ 13.84,\ 16.37$$

and:

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

Thus, the proposed approach for the rank calculation in a minimally redundant RNS with length k of primary residue code from 5 to 30 digits compared to a conventional non-redundant RNS allows us to reduce computational costs by 3.2–16.37 times in terms of the required modular addition operations and by 2.8–15.47 times in terms of the lookup table memory.

To illustrate the calculation of rank $\rho_k(X)$ in a minimal redundant RNS based on that which is mentioned above, we present a simple numerical example. For convenience, we consider a sample RNS with four moduli in the primary moduli set.

Example 1. *Let us consider an RNS with moduli $m_1 = 5$, $m_2 = 7$, $m_3 = 9$, and $m_4 = 11$ and excess modulus $m_0 = 2$. Suppose we wish to calculate rank $\rho_k(X)$ of number $X = 13$ having minimally redundant residue code $(1, 3, 6, 4, 2)$.*

The primitive constants in this RNS are:

$$M_4 = 3465$$

$$M_{1,4} = 693, \quad \left|M_{1,4}^{-1}\right|_{m_1} = \left|693^{-1}\right|_5 = 2$$

$$M_{2,4} = 495, \quad \left|M_{2,4}^{-1}\right|_{m_2} = \left|495^{-1}\right|_7 = 3$$

$$M_{3,4} = 385, \quad \left|M_{3,4}^{-1}\right|_{m_3} = \left|385^{-1}\right|_9 = 4$$

$$M_{4,4} = 315, \quad \left|M_{4,4}^{-1}\right|_{m_4} = \left|315^{-1}\right|_{11} = 8$$

$$\left|m_1^{-1}\right|_{m_4} = \left|5^{-1}\right|_{11} = 9$$

$$\left|m_2^{-1}\right|_{m_4} = \left|7^{-1}\right|_{11} = 8$$

$$\left|m_3^{-1}\right|_{m_4} = \left|9^{-1}\right|_{11} = 5$$

First, residues $R_{i,4}(\chi_i)$ and $\chi_{i,4}^{(0)}$ calculated according to (5), (6), and (13), respectively $(i = 1, 2, 3, 4)$. As a result, we have:

$$R_{1,4}(\chi_1) = \left|- |2 \cdot 3|_5 \cdot 9\right|_{11} = 2$$

$$R_{2,4}(\chi_2) = \left|- |5 \cdot 6|_7 \cdot 8\right|_{11} = 6$$

$$R_{3,4}(\chi_3) = \left|- |8 \cdot 4|_9 \cdot 5\right|_{11} = 8$$

$$R_{4,4}(\chi_4) = |8 \cdot 2|_{11} = 5$$

and:

$$\chi_{1,4} = |2 \cdot 3|_5 = 1, \quad \chi_{1,4}^{(0)} = |1|_2 = 1$$

$$\chi_{2,4} = |3 \cdot 6|_7 = 4, \quad \chi_{2,4}^{(0)} = |4|_2 = 0$$

$$\chi_{3,4} = |4 \cdot 4|_9 = 7, \quad \chi_{3,4}^{(0)} = |7|_2 = 1$$

$$\chi_{4,4} = |8 \cdot 2|_{11} = 5, \quad \chi_{4,4}^{(0)} = |5|_2 = 1$$

Furthermore, we find the number of overflows when summing the residues of the set:

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

modulo $m_4 = 11$; i.e., we calculate inexact rank:

$$\widehat{\rho}_4(X) = \lfloor (2 + 6 + 8 + 5)/11 \rfloor = \lfloor 21/11 \rfloor = 1$$

Therefore, $\widehat{\rho}_4^{(0)} = \left|\widehat{\rho}_4(X)\right|_2 = 1$*.*

Then, taking into account the fact that redundant residue $\chi_0 = 1$*, according to (15), we calculate two-valued correction:*

$$\delta_4(X) = \left|1 + (1 + 0 + 1 + 1) + 1\right|_2 = |5|_2 = 1$$

As a result, we get rank:

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

To verify the obtained result, using (2), we find:

$$X = \sum_{i=1}^{4} M_{i,4}\chi_{i,4} - \rho_4(X) M_4 =$$

$$= 693 \cdot 1 + 495 \cdot 4 + 385 \cdot 7 + 315 \cdot 5 - 2 \cdot 3465 = 6943 - 6930 = 13$$

5. Discussion

The known methods for reducing the complexity of the CRT implementation directed both the reduction of the complexity of the inner multiplications as well as the substitution of the additions modulo M_k by the simpler operations in (1) [2, 3, 7, 20, 23, 24, 27].

For the first time, a positional characteristic called the rank $r(X)$ of a number X was studied in [2] and later in [3]. The rank computation consists of a slow k-step iterative procedure of the sequential additions of specific constants, which are determined by the RNS moduli-set $\{m_1, m_2, \ldots, m_k\}$, modulo M_k. Moreover, the upper bound of rank $r(X)$ depends on the values of weights $\mu_{1,k}, \mu_{2,k}, \ldots, \mu_{k,k}$ (see (1)) and can be sufficiently large for most moduli-sets that are suitable for practical use.

In [29], a well-known method for obtaining a positional value of an integer number in an RNS is proposed. This is based on the calculation of an integer correction factor for the CRT implementation in the form of an exact integer equation. Thus, the slow and challenging addition modulo M_k is replaced by subtraction and multiplication. At that, the efficient calculation of the correction factor is critical. In this method, redundant modulus m_{k+1} adds to the primary RNS moduli set $\{m_1, m_2, \ldots, m_k\}$. At the same time, the redundant modulus must satisfy two conditions for the proper calculation of the correction factor: first, $m_{k+1} \geqslant k$; and second, $\gcd(m_{k+1}, M_k) = 1$. Thus, we have an extra modular channel. Nevertheless, this method cannot be used to find the correction factor of a number resulting from a subtraction. Therefore, it is not sufficient to a sign determination nor, consequently, a magnitude comparison of two numbers in the RNS, but it can be used for the base extension operation.

In [16, 26, 33], a different approach to the evaluation of the CRT reconstruction coefficient was proposed. The main idea of the considered method consists of the representation of reconstruction coefficient r (i.e., a rank $\rho_k(X)$ in our case) as an integer part of a sum of at most k proper fractions. The value of r is recursively estimated by approximating $\chi_{i,k}$ and m_i at fraction $\chi_{i,k}/m_i$ $(i = 1, 2, \ldots, k)$. To avoid a division in the fraction by modulus m_i, denominator m_i is replaced by 2^n, while numerator $\chi_{i,k}$ is approximated by its most significant v bits, where n is common to all moduli, $m_i < 2^n$, $n = \lceil \log_2 m \rceil$, $m = \max_{1 \leqslant i \leqslant k} \{m_i\}$, $v < n$ $(i = 1, 2, \ldots, k)$. Since the division by powers of 2 is equivalent to simple shifts, then the calculation of reconstruction coefficient r can be realized by addition only. The main drawbacks of this method consist of the following reasons. First of all, full precision fractional computations are required. On the other hand, the iterations are on the order of the bit-length needed. For example, the method described in [33] uses a very high precision of $\lceil \log_2(kM_k) \rceil$ bits. As for the method proposed in [16, 26], it uses a sequential bit-by-bit manner for the evaluations of reconstruction coefficient r. Therefore, the fractional domain methods are prolonged in the case of large word-length numbers (for example, of the 1024 bit-length commonly used in cryptographic applications). At the same time, these methods, of course, are crucial for high-performance applications in the field of digital signal processing, where the used dynamic range is covered by the use of small moduli of 4–6 bit-lengths.

The main advantage of the proposed minimally redundant RNS over non-redundant analogues consists of a significant simplification of the calculation of rank $\rho_k(X)$. First of all, the transition from a non-redundant to a minimally redundant residue representation of numbers allows us to replace correction coefficient $\Delta_k(X)$ by two-valued characteristic $\delta_k(X)$, which has a trivial computational structure and can be calculated in one modular clock cycle by means of a lookup tables technique or by using the simplest combination logic circuit. As for an inexact rank $\widehat{\rho}_k(X)$ (see Theorem 1), this is calculated by counting the occurred overflows when summing k specified residues modulo m_k, both in the case of a non-redundant and minimally redundant RNS.

In accordance with (18) and (19), the computational complexity of the rank calculation is significantly reduced as compared with non-redundant analogues due to the use of minimum code redundancy. At the same time, efficiency factors C_{MO} and C_{LUT} show significant gains with respect to the number of required modular addition operations and lookup tables for storing the required constants depending on the number of used modules and, accordingly, the dynamic range of the RNS.

6. Conclusion

This paper presents a computationally efficient approach that allows us to improve the implementation of the CRT algorithm based on the use of the rank of a number, which is a positional characteristic of primary importance in an RNS. The use of the minimum-redundancy residue code enables the optimization of the rank computation and, consequently, the performance of the non-modular operations based on the CRT.

In contrast to a straightforward implementation of the CRT algorithm, the new method (which is based on the modified form of the CRT represented as exact integer equality) does not use a large word-length modular addition operations modulo M_k representing a product of k relatively prime moduli $m_1, m_2, \ldots, m_k$. Moreover, the proposed method also does not require arithmetic operations in the fractional domain for the rank calculation.

We investigated the structure of rank $\rho_k(X)$ and proposed a novel method for the fast calculation of rank correction $\Delta_k(X)$ to inexact rank $\widehat{\rho}_k(X)$. This method used the fact that rank correction $\Delta_k(X)$ is a two-value number: $\Delta_k(X) \in \{0,1\}$. This allows for a significant reduction in the computational complexity of the rank calculation to $O(k)$ instead of $O\left(k^2\right)$, owing to the extension of primary non-redundant moduli-set $\{m_1, m_2, \ldots, m_k\}$ by redundant extra modulus $m_0 = 2$. In essence, two-valued excess residue χ_0 determines the parity of an RNS number.

The reduction factors of computational complexity of the rank $\rho_k(X)$ calculation concerning the numbers of required modular addition operations and lookup tables in a minimally redundant RNS are presented. As shown, these factors increase with the number k of non-redundant RNS moduli. For example, the use of a minimally redundant RNS with length k of primary residue code from 5 to 30 digits enables us to reduce computational costs by 3.2-16.37 times in terms of the required modular addition operations and by 2.8-15.47 times in terms of the lookup table memory.

This circumstance allows us to simplify the implementation of the CRT and, consequently, to construct variants of RNS arithmetic that are faster and optimal in cost by improving the performance of the non-modular procedures.

Therefore, the minimally redundant RNS proposed in this paper takes priority over a conventional non-redundant RNS in the field of high-speed computing, especially in the case of a large dynamic range computation, for example, for implementing various complex cryptographic algorithms. Thus, the considered approach coincides with the development vector of modern methods and algorithms for high-performance computing.

References

[1] Abdul-Mumin S., Gbolagade K.A.: Rivest Shamir Adleman Encryption Scheme Based on the Chinese Remainder Theorem, *Advances in Networks*, vol. 6(1), pp. 40–47, 2018.

[2] Akushskii I.Y., Juditskii D.I.: *Machine Arithmetic in Residue Classes* [in Russian], Sovetskoe Radio, Moscow, 1968.

[3] Amerbayev V.M.: *Theoretical Foundations of Machine Arithmetic*, Nauka, Alma--Ata, Kazakh. SSR, 1976.

[4] Aremu I.A., Gbolagade K.A.: Redundant Residue Number System Based Multiple Error Detection and Correction Using Chinese Remainder Theorem, *Software Engineering*, vol. 5(5), pp. 72–80, 2017.

[5] Bajard J.C., Eynard J., Merkiche N.: Montgomery reduction within the context of residue number system arithmetic, *Journal of Cryptographic Engineering*, vol. 8(3), pp. 189–200, 2018.

[6] Burton D.M.: *Elementary Number Theory.* Allyn and Bacon, Boston, 1980.

[7] Chernyavsky A.F., Danilevich V.V., Kolyada A.A., Selyaninov M.Y.: *High-Speed Methods and Systems of Digital Information Processing.* Belarusian State University, Minsk, Belarus, 1996.

[8] Courtois J., Abbas-Turki L.A., Bajard J.C.: Resilience of Randomized RNS Arithmetic With Respect to Side-Channel Leaks of Cryptographic Computation, *IEEE Transactions on Computers*, vol. 68(12), pp. 1720–1730, 2019.

[9] Ding C., Pei D., Salomaa A.: *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography.* World Scientific Publishing Co., River Edge, NJ, United States, 1996.

[10] Duseja T., Deshmukh M.: Image compression and encryption using chinese remainder theorem, *Multimedia Tools and Applications*, vol. 78(3), pp. 16727–16753, 2019.

[11] Hardy G.H., Wright E.M., Wiles A.: *An Introduction to the Theory of Numbers.* Oxford University Press, Ely House, London, 6 ed., 2008.

[12] Hsiao T.C., Huang Y.T., Huang Y.M., Chen T.L., Chen T.S., Wang S.D.: Efficient and Scalable Access Management Scheme Based on Chinese Remainder Theorem, *Sensors and Materials*, vol. 30(3), pp. 413–422, 2018.

[13] Ibrahim M.B., Gbolagade K.A.: A Chinese Remainder Theorem Based Enhancements of Lempel-ziv-welch and Huffman Coding Image Compression, *Asian Journal of Research in Computer Science*, vol. 3(3), pp. 1–9, 2019.

[14] Jakubski A.: Selected application of the Chinese Remainder Theorem in multiparty computation, *Journal of Applied Mathematics and Computational Mechanics*, vol. 15(1), pp. 39–47, 2016.

[15] Jia X., Song Y., Wang D., Niel D., Wu J.: A collaborative secret sharing scheme based on the Chinese Remainder Theorem, *Mathematical Biosciences and Engineering*, vol. 16(3), pp. 1280–1299, 2019.

[16] Kawamura S., Koike M., Sano F., Shimbo A.: Cox-Rower Architecture for Fast Parallel Montgomery Multiplication. In: *EUROCRYPT'00: Proceedings 19th International Conference on Theory and Application of Cryptographic Techniques (Bruges, Belgium, May 14–18, 2000)*, pp. 523–538. Springer, Berlin, 2000.

[17] Kirankumar V., Ramasubbareddy S., Kannayaram G., Vishalo G., Nikhil K.: Enhanced Security Using Rivest, Shamir, and Adelman With Chinese Remainder Theorem, *Journal of Computational and Theoretical Nanoscience*, vol. 16(5-6), pp. 2018–2021, 2019.

[18] Knuth D.E.: *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, Addison-Wesley Longman Publishing Co., Boston, 3 ed., 1997.

[19] Kolyada A.A., Pak I.T.: *Modular Structures for Pipelining Digital Information Processing*, Belarusian State University, Minsk, Belarus, 1992.

[20] Kolyada A.A., Selyaninov M.Y.: Generation of integral characteristics of symmetric-range residue codes, *Cybernetics and Systems Analysis*, vol. 22(4), pp. 431–437, 1986.

[21] Madandola T.N., Gbolagade K.A., Yusuf-Asaju A.W.: Effect of Chinese Remainder Theorem on Principal Component Analysis, *International Journal of Scientific & Engineering Research*, vol. 10(7), pp. 2142–2148, 2019.

[22] Meng K., Miao F., Huang W., Xiong Y.: Tightly coupled multi-group threshold secret sharing based on Chinese Remainder Theorem, *Discrete Applied Mathematics*, vol. 268, pp. 152–163, 2019.

[23] Mohan P.V.A.: *Residue Number Systems. Theory and Applications*, Springer, Cham, Switzerland, 2016.

[24] Molahosseini A., Sousa de L.S., Chang C.H.: *Embedded Systems Design with Special Arithmetic and Number Systems*, Springer, Cham, Switzerland, 2017.

[25] Niyi M.T., Gbolagade K.A.: Reducing computational Time of Principal Component Analysis with Chinese Remainder Theorem, *International Journal of Discrete Mathematics*, vol. 4(1), pp. 1–7, 2019.

[26] Nozaki H., Motoyama M., Shimbo A., Kawamura S.: Implementation of RSA Algorithm Based on RNS Montgomery Multiplication. In: *CHES 2001: Cryptographic Hardware and Embedded Systems, Third International Workshop (Paris, France, May 14–16, 2001)*, pp. 364–376, Springer, Berlin, 2001.

[27] Omondi A.R., Premkumar B.: *Residue Number Systems: Theory and Implementation*, Imperial College Press, London, 2007.

[28] Sheikhi-Garjan M., Bahramian M., Doche C.: Threshold verifiable multi-secret sharing based on elliptic curves and Chinese remainder theorem, *IET Information Security*, vol. 13(3), pp. 278–284, 2019.

[29] Shenoy M.A., Kumaresan R.: A fast and accurate RNS scaling technique for high speed signal processing, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, pp. 929–937, 1989.

[30] Shoup V.: *A Computational Introduction to Number Theory and Algebra*, Cambridge University Press, UK, 2 ed., 2005.

[31] 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, NJ, 1986.

[32] Szabo N.S., Tanaka R.I.: *Residue Arithmetic and Its Application to Computer Technology*, McGraw-Hill, New York, 1967.

[33] Vu T.V.: Efficient Implementations of the Chinese Remainder Theorem for Sign Detection and Residue Decoding, *IEEE Transactions on Computers*, vol. C-34(7), pp. 646–651, 1985.

[34] Yan X., Lu Y., Liu L., Liu J., Yang G.: Chinese remainder theorem-based two-in--one image secret sharing with three decoding options, *Digital Signal Processing*, vol. 82, pp. 80–90, 2018.

Affiliations

Mikhail Selianinau
Jan Dlugosz University in Czestochowa, Armii Krajowej 13/15, 42-200 Częstochowa, Poland, m.selianinau@ujd.edu.pl, ORCID ID: https://orcid.org/0000-0001-5669-701X

Received: 10.01.2020
Revised: 27.03.2020
Accepted: 28.03.2020