computation

MDPI

Article

Fast Operation of Determining the Sign of a Number in RNS Using the Akushsky Core Function

Egor Shiriaev *, Nikolay Kucherov, Mikhail Babenko and Anton Nazarov

Faculty of Mathematics and Computer Science, North-Caucasus Federal University, 355017 Stavropol, Russia; nkucherov@ncfu.ru (N.K.); mgbabenko@ncfu.ru (M.B.); anazarov@ncfu.ru (A.N.)
* Correspondence: eshiriaev@ncfu.ru

Abstract: This article presents a study related to increasing the performance of distributed computing systems. The essence of fog computing lies in the use of so-called edge devices. These devices are low-power, so they are extremely sensitive to the computational complexity of the methods used. This article is aimed at improving the efficiency of calculations while maintaining an appropriate level of reliability by applying the methods of the Residue Number System (RNS). We are investigating methods for determining the sign of a number in the RNS based on the core function in order to develop a new, fast method. As a result, a fast method for determining the sign of a number based on the Akushsky core function, using approximate calculations, is obtained. Thus, in the course of this article, a study of methods for ensuring reliability in distributed computing is conducted. A fast method for determining the sign of a number in the RNS based on the core function using approximate calculations is also proposed. This result is interesting from the point of view of nebulous calculations, since it allows maintaining high reliability of a distributed system of edge devices with a slight increase in the computational complexity of non-modular operations.

Keywords: Residue Number System; core function; modular calculations; reliability; approximate calculations; positional characteristic; number rank

Citation: Shiriaev, E.; Kucherov, N.; Babenko, M.; Nazarov, A. Fast Operation of Determining the Sign of a Number in RNS Using the Akushsky Core Function. *Computation* **2023**, *11*, 124. https://doi.org/10.3390/computation11070124

Academic Editor: Rafael Lahoz-Beltra

Received: 30 May 2023
Revised: 23 June 2023
Accepted: 26 June 2023
Published: 28 June 2023

1. Introduction

Fog calculations (FC) [1,2]—this is a concept of distributed computing, which involves the use of resources from not only centralized cloud services but also distributed devices in close proximity to end users (for example, Internet of Things (IoT) devices [3], sensors, routers, etc.). The main idea is to move computing power closer to the data usage location to reduce latency, provide faster access to data, and reduce the load on the cloud.

Such a computing system architecture can be effective, for example, for the Smart City (SM) [4]. However, when building an FC system, a few problems arise related to the reliability and security of the system being developed.

One of the main advantages of FC is the ability to collect and analyze data in real time, which allows faster decision-making and improves responsiveness. In addition, they can be used to offload computing and storage from the cloud, reducing the costs and delays associated with cloud solutions. However, this also creates new challenges such as security, uncertainty, data management, and deployment.

In existing implementations of cloud applications, most of the data that needs to be stored, analyzed, and made decisions about are sent to cloud data centers. However, as the volume of data increases, there are problems of efficiency and even the impossibility of transferring information between IoT devices and the cloud. This is due to bandwidth limitations and minimum latency requirements in the computer network. The advent of applications that require instant response (such as patient monitoring, self-driving cars, and others) makes the remote cloud unable to provide reliable, low-latency communications for these applications. Additionally, in some cases, sending data to the cloud may not be possible due to privacy issues.

In order to overcome the problems associated with the high bandwidth of the computer network, work with geographically distributed data sources, minimal delays, and data processing in place, there is a need for a universal approach to organizing computing both in the cloud and on nodes, closer to connected devices. To solve this problem, the concept of FC is used, which bridges the gap between the cloud and IoT devices by providing computing, storage, networking, and management capabilities at nodes located directly next to IoT devices. This allows more efficient processing of data and interaction in the network.

For example, [5] H. Hua et al. consider the organization and construction of boundary calculations. The problems of computing offloading, resource allocation, and privacy and security are also considered. Various researchers commonly use the Lyapunov optimization algorithm to solve the described problems [6–9]. But the authors propose the use of a combined approach that combines boundary computing and artificial intelligence. This paper considers several ideas of artificial intelligence for working both in the cloud and in the fog with data processing in them. The authors offer an important message to all researchers, which is the importance of combining artificial intelligence and edge computing. However, the use of artificial intelligence is a rather computationally complex task. In addition, when working with artificial intelligence, special attention must be paid to the reliability of the system. To ensure reliability and security, large amounts of calculations are required. To reduce the computational load, RNS can be used, with its parallelism and self-correction properties. There are works that prove the effectiveness of the use of RNS in distributed computing systems [10,11].

After analyzing the existing literature on the topic of security and reliability of distributed systems, you can notice a certain trend. To increase reliability, block [12–16] and correction codes are usually used [17–21], as well as data replication [22]. However, these methods significantly increase the redundancy of information in the system. To reduce the effect of redundancy, many researchers propose the use of RNS and its self-correcting properties. For example, there are several methods for detecting and correcting errors in RNS [23–30]. The use of these developments makes it possible to reduce the redundancy of the system with a high level of reliability and fault tolerance. Usually these methods in RNS are divided into two groups: methods based on number projection [31] and methods based on the error syndrome [32]. The methods discussed earlier are either modifications of the syndrome or projection method, or they are also used.

The security of distributed systems is traditionally proposed to be provided using asymmetric ciphers [33], such as RSA [34]. There are many works devoted to this topic [35–37]. In addition, in the scientific community, works related to security in distributed using hybrid technologies are popular. This is caused by the low speed of RSA encryption. Hybrid encryption schemes are often used in practice, in which RSA is used to encrypt a symmetric key [38–43]. To reduce the costs associated with the use of asymmetric encryption together with symmetric encryption in scientific papers, the use of secret division schemes is proposed [44] instead of asymmetric encryption. Some secret sharing schemes use RNS to speed up their arithmetic [45–47]. In addition, for distributed systems, the use of fully homomorphic encryption schemes is proposed, which also use RNS to speed up their arithmetic [48–51].

Now, RNS finds its application in various scientific fields. This proves the relevance of this number system and confirms its positive properties. As mentioned above, RNS is relevant in the field of security due to its properties—for example, to speed up the operation of a homomorphic encryption scheme, as per A. Kim et al. [52]. With the help of RNS, a reduction in the approximation error is achieved, which makes it possible to increase the efficiency of the circuit calculation. In work [53], the authors propose RNS as the basis of a new asymmetric cryptalgorithm. RNS modules are proposed to be used as secret keys. The public keys are freely chosen coefficients for multiplying the residuals. Another application of RNS is in its self-correcting properties. In their work [54], I. A. Kalmykov et al. use polynomial RNS to correct one and two errors. The authors claim that their solution allows

to increase the speed of fault-tolerant high-speed computing systems for digital signal processing operating in Gaul fields. The positive properties of RNS are also demonstrated in the work by K. Givaki et al. [55]. In their work, the authors argue that the use of integrated circuits based on RNS will reduce peak power and reduce thermal costs. In addition, as a result, the authors claim a decrease in the area of circuits compared to analogues.

However, for effective implementation of RNS methods in distributed computing systems, it is necessary to reduce the computational complexity of non-modular operations in RNS. Such operations include, for example, operations of comparison and determination of the sign of a number. When processing and storing information, these operations are extremely important since they allow you to determine the correspondence of data when searching for matches. When considering practical applications—determining the readings of various sensors, such as temperature—the transfer of numbers from RNS to a positional system is not advisable from the point of view of the organization of the system. The methods discussed above ensure the reliability and security of the system when presenting data in RNS. Thus, the task is to increase the speed of non-modular RNS operations. One of the main operations is the operation of determining the sign of a number since it allows you to implement other operations, such as comparison, division, etc. In the case of RNS, to determine the sign of a number, it is optimal to find the so-called positional characteristic of the number. To calculate the positional characteristic of a number, Akushsky proposed a kernel function that allows you to determine the positional characteristic in a simpler way. Thus, in this paper, we propose a fast operation for determining the sign of a number using the core function by Akushsky.

This work is organized as follows: in Section 2, the general terminology of RNS and its main properties and methods are given; in Section 3, the core function by Akushsky, its properties and methods, the concept of the positional character of a number, as well as methods for determining the sign of a number are given; in Section 4, a description of an experimental study and its results and discussions are given; in Section 5, the conclusions obtained in the course of the study are presented, as well as a description of future work.

2. Residue Number System

Residue Number System (RNS) is a non–positional number system [56], which is based on the principles of the Chinese Remainder Theorem (CRT) [57]. CRT represents statements related to the solution of linear comparison systems, which is interpreted as

$$\begin{cases} x_1 \equiv X \ mod \ p_1 \\ x_2 \equiv X \ mod \ p_1 \\ \quad \dots \\ x_n \equiv X \ mod \ p_n \end{cases}, \tag{1}$$

where n—the number of elements in the tuple of integers $(p_1, p_2, \dots, p_n)$; X—integer. If the tuple $(p_1, p_2, \dots, p_n)$ consists of mutually prime numbers, then the correspondence $X \Leftrightarrow \{x_1, x_2, \dots, x_n\}$ is one-to-one. It also follows from the CRT that one-to-one is provided, if $X < P$. Then, X is represented in RNS as $X \overset{RNS}{\to} \{x_1, x_2, \dots, x_n\} = \sum_{i=1}^{n} X \ mod \ p_i$, and the comparison system (1) is a representation of RNS.

The main advantage of RNS is computationally simple modular operations that can be performed in parallel. These operations include addition, multiplication, and subtraction. Let $A \overset{RNS}{\to} \{a_1, a_2, \dots, a_n\}$ and $B \overset{RNS}{\to} \{b_1, b_2, \dots, b_n\}$; then, the general view of the modular operation, the result of which is $C \overset{RNS}{\to} \{c_1, c_2, \dots, c_n\}$, will be the following:

$$A \circ B = \{(a_1 \circ b_1) \ mod \ p_1, (a_2 \circ b_2) \ mod \ p_2, \dots, (a_n \circ b_n) \ mod \ p_n\} == \{c_1, c_2, \dots, c_n\} = C.$$

Operations on each i-th remainder are independent of other residuals, which allows operations to be performed in parallel. In addition, the independence of the residuals from each other and the dependence on the RNS basis allow us to talk about the properties

of self-correction since each remainder can be checked for correctness and, if necessary, restored or deleted from the general system, provided that after its deletion, $X < P$ is performed for the new basis. There are several methods for error control in RNS.

To handle negative numbers, CRT supports the following ratio: $-P < 2X < P$. Then, for a positive number, $|X| < \frac{P}{2}$. However, in order to distinguish between positive and negative numbers, the so-called complement is introduced in the unsigned RNS. The complement is a system of residues obtained as $P \overset{RNS}{\rightarrow} \{\rho_1, \rho_2, \ldots, \rho_n\} = \sum_{i=1}^{n} P \ mod \ p_i$.

Then, a negative number can be expressed as

$$x_i' = X' \ mod \ p_i + P \ mod \ p_i.$$

Negative numbers are in the range $\frac{P}{2} < X' < P$.

Non-modular RNS operations include the following: determining the sign of a number, comparison, base expansion, division, etc.

To implement the definition of the sign of a number, the traditional method is to convert a number to a Weighted Number System (WNS) and compare the number in positional form with the condition that if the resulting number $X'' > \frac{P}{2}$, P is subtracted from it to obtain a negative value. CRT supports the conversion of a number from RNS to WNS based on the following provisions. If the comparison system (1) is a representation of the number X in RNS, then the number itself is its solution. Thus, to find a number, it is necessary to solve this system. The CRT supports the theorem on the inverse values of modules, on the basis of which it is possible to solve this system. Then, the return of the number is reduced to the following formula:

$$X = \left| \sum_{i=1}^{n} x_i \cdot P_{p_i}^{-1} \cdot P_i \right|_P, \tag{2}$$

where $P_i = P/p_i$, $P_{p_i}^{-1}$—multiplicative inversion $\frac{1}{P_i} \ mod \ p_i$ which can be obtained, for example, using the Euclid algorithm [58]. The product $P_{p_i}^{-1} \cdot P_i$ for RNS is defined as the orthogonal basis B_i. Then, expression (2) is transformed to the form

$$X = \left| \sum_{i=1}^{n} x_i \cdot B_i \right|_P. \tag{3}$$

In addition to this method, there are other methods based on various properties of RNS—for example, the Garner method [59], Mixed Radix System (MRS) method [60], etc. However, most of them have computational complexity equal from $O(n)$ to $O(n^2)$. In addition, the very operation of converting a number from RNS to WNS to determine the sign reduces the efficiency and speed of data processing in systems where RNS is used. Algorithms based on the positional characteristic of a number allow avoiding such a translation—for example, the core function method. However, this method also has disadvantages related to computational complexity. In this paper, we will consider the properties of RNS, which will allow us to modify the method of the kernel function and reduce the impact of shortcomings on computation.

3. Methods for Determining the Sign of a Number Based on the Core Function

3.1. Core Function Method

As noted in the previous section, the method of determining the sign of a number based on CRT is not effective from the point of computational complexity. In this article, we consider another, more effective method. This method was developed by Akushsky in his work [61]. Consider it.

For the core function in RNS, the so-called core $C(P)$ is selected, which is the largest module from the set $\{p_1, p_2, \ldots, p_n\}$ or the product of two or more modules.

In the case of the core function, the so-called weights are used, which are determined by the following formula:

$$w_i = \left(\left|P_i^{-1}\right|_{p_i} \cdot C(P)\right) \; mod \; p_i. \tag{4}$$

However, the weights must satisfy the condition, so that $C(P) = \sum_{i=1}^{n} P\frac{w_i}{p_i}$. This allows us to assert the possibility of the existence of negative weights as well. In addition, for the basis $B_i = \left|P_i^{-1}\right|_{p_i}$, the core is also defined:

$$C(B_i) = B_i \cdot \frac{C(P)}{P} - \frac{w_i}{p_i}. \tag{5}$$

Then, having weights as well as the core functions of the bases $C(B_i)$ and a set of modules $C(P)$, it is possible to obtain a function of the number itself X:

$$C(X) = \left(\sum_{i=1}^{n} x_i \cdot C(B_I)\right) \; mod \; C(P). \tag{6}$$

Having the core function of a number, it is also possible to restore a number. However, this is not necessary, since the core function of a number is its positional characteristic, which follows from the definition of the core function itself. Then, $C(X)$ is sufficient to determine its sign. Based on CRT, if $C(X) < \frac{P}{2}$, then the number is positive, and if $C(X) > \frac{P}{2}$, then the number is negative.

3.2. Core Function Method Based on the Rank of a Number

However, although the considered method has greater efficiency than, for example, the CRT method, the division operation modulo $C(P)$ is computationally complex and, with a large size of modules, will reduce system performance. Thus, it is necessary to develop a method for quickly determining the sign of a number based on the core function by Akushsky. To solve this problem, it is possible to use the so-called rank of the RNS number.

As noted earlier, when using RNS, there are several problems associated with non-modular operations. As a solution to this problem, it is possible to use the so-called positional characteristics. In addition to using the core function of a number, the positional characteristic of a number is its rank.

However, there is a problem associated with its definition, since in comparison

$$x_1B_1 + x_2B_2 \ldots + x_nB_n \equiv N \; mod \; P = N + rP$$

rank is a coefficient at P, not a modular quantity. The relation of the core and the rank of the number is established in the work [62]. Then, Formula (5) can be rewritten in the following form:

$$(X) = \left(\sum_{i=1}^{n} x_i \cdot B_i\right) - r_X P, \tag{7}$$

where r_X—rank of number X. Rank calculation formula is as follows:

$$r_X = \frac{\left(\sum_{i=1}^{n} x_i \cdot B_i\right)}{P}. \tag{8}$$

However, as mentioned earlier, the number P is quite large. Therefore, it is necessary to perform several transformations to simplify the expression. Taking Formula (3) of the return of the number from RNS to WNS from Section 2 and Formula (7), we can obtain a formula such as

$$C(X) = \left(\sum_{i=1}^{n} w_i \cdot \left(\sum_{j=1}^{n} \frac{B_j x_j}{p_i} - r_x P\right)\right).$$

Given Formula (2) and the fact that p_i is a divisor of $C(B_i)$, and often $p_n = C(P)$, we obtain

$$C(X) = \left(\sum_{i=1}^{n} x_i \cdot C(B_i) \right) - r_X C(P). \quad (9)$$

Thus, it is possible to apply the rank of a number to the core function. In addition, we got rid of the computationally complex modulo division operation. The resulting expression is already more productive than the one obtained in Section 3.2, but this result was obtained in the work [62], and it is effective in the hardware implementation of the RNS system based on programmable logic integrated circuits, which is not effective in the implementation of FC.

3.3. The Core Function Method Based on the Rank of Number by Chervyakov

There are several more efficient methods for calculating the rank of the RNS number. Let us consider them and modify the method of determining the sign of a number using the core function.

Chervyakov in his work [63] suggested a different formula for calculating the rank of a number. In fact, Formula (8) undergoes changes related to the properties of the RNS basis, which allows you to omit unnecessary operations and obtain a rank based on the expression

$$r_X = \sum_{i=1}^{n} \left\lfloor x_i \cdot \frac{\left| P_i^{-1} \right|_{p_i}}{p_i} \right\rfloor. \quad (10)$$

Such transformations follow from the following Lemma.

Lemma 1.

$$r_X = \sum_{i=1}^{n} \left\lfloor x_i \cdot \frac{\left| P_i^{-1} \right|_{p_i}}{p_i} \right\rfloor.$$

This transformation of the formula makes it possible to increase productivity by several percents (see Section 4). However, it is possible to calculate the rank of a number more efficiently by calculating the rank using approximate methods. Consider this method.

3.4. The Core Function Method Based on the Approximate Rank of a Number

We propose a fast function for determining the sign of a number based on the core function with approximate rank. Approximate rank calculation allows to increase efficiency due to less strict iterative formulas with loss of calculation accuracy. However, RNS is an integer number system. Due to the correct lower and upper estimates, it is possible to keep the accuracy within the necessary limit to preserve the correct integer value, which was proved in the work [64].

To obtain an approximate rank, the coefficient is used: $N = \log_2(P \cdot (\sum_{i=1}^{n} p_i - n))$. This coefficient has to establish a connection between the RNS and the binary number system due to 2^N. Then, we can establish the relationship between the rank of the number and 2^N due to the approximation factor

$$k_i = \left\lfloor \frac{\left| P_i^{-1} \right|_{p_i} 2^N}{p_i} \right\rfloor. \quad (11)$$

Substitute Formula (11) into (10) and obtain a modified formula to calculate the rank of a number. Calculations using this formula will be carried out faster due to a decrease in the accuracy and synergy of the approximate Chervyakov method and the method for

calculating the core function by Akushsky. The approximate value of the rank of a number can be found by

$$r_x^* = \sum_{i=1}^{n} \left\lfloor \frac{k_i x_i}{2^N} \right\rfloor. \tag{12}$$

because the values of 2^N and the value of the approximate coefficients $x_1, x_2, \ldots, x_n$ can be found at the stage of precomputation. This allows us to reduce the computational complexity of operations due to a slight (compared to other methods based on the core function) increase in memory consumption.

Then, the full-fledged sign definition function, based on the core function with approximate rank, has the following form:

$$sign(X) = \begin{cases} 1 \quad if \quad C(X) = \left(\sum\limits_{i=1}^{n} x_i \cdot C(B_i) \right) - r_x^* C(P) < \frac{P}{2} \\ 0 \quad if \quad C(X) = \left(\sum\limits_{i=1}^{n} x_i \cdot C(B_i) \right) - r_x^* C(P) \geq \frac{P}{2} \end{cases} \tag{13}$$

In the next section, a performance study of the considered methods will be conducted in order to confirm the relevance of the obtained result in the study.

4. Results

Based on the methods described above, we will conduct a study of their performance in terms of execution time. For the study, mutually simple modules ranging in size from 8 to 1024 bits and in the amount of 21 elements in one set were taken; there are only 9 sets. Research is conducted on the basis of programs written in C++ on equipment with the following characteristics:

- CPU: Frequency: 2.90 GHz, Core—6, Process technology 14 nm
- GPU: Video memory 6144 MB, Memory frequency 14,000 MHz, GPU frequency 1680 MHz, TDP 500 W
- RAM: 16 GB, Frequency 3200 MHz
- OS: Windows 10

The NTL library is used to implement long arithmetic [65]. The experiment is as follows, and the study is carried out in two stages:

- Stage A—performance study of 9 sets, 21 modules, dimension from 8 to 1024 bits;
- Stage B—performance study of 20 sets, from 3 to 21 modules, dimension of 32 bits.

When conducting a two-stage simulation, the time characteristics of each method were obtained. The results obtained are reflected in the tables (Tables 1 and 2). The time is given in seconds.

Table 1. The result of the study of stage A (Bit—the size of one module in a set in bits).

Bit	Core Method	Rank Core Method	Rank Chervyakov Core Method	Approximate Rank Core Method
8	0.00003753	0.00004114	0.00004959	0.00003219
16	0.00003776	0.00003826	0.00004781	0.00003226
32	0.00005424	0.00004512	0.00005613	0.00003582
64	0.00009028	0.00006913	0.00007189	0.00004645
128	0.00017094	0.00013323	0.00010866	0.00007626
256	0.00035464	0.00015862	0.00012334	0.00008889
512	0.00097338	0.00018338	0.00013638	0.00010082
1024	0.00332057	0.00022026	0.00015774	0.00012020

Table 2. The result of the study of stage B (p[n] is the length of the set of modules, where n is the number of modules in the set).

p[n]	Core Method	Rank Core Method	Rank Chervyakov Core Method	Approximate Rank Core Method
3	0.00001990	0.00001951	0.00002319	0.00001634
4	0.00002614	0.00002465	0.00002959	0.00002045
5	0.00003163	0.00002893	0.00003493	0.00002365
6	0.00003847	0.00003473	0.00004164	0.00002776
7	0.00004784	0.00004030	0.00004936	0.00003214
8	0.00005414	0.00004488	0.00005407	0.00003505
9	0.00006395	0.00005184	0.00006285	0.00004180
10	0.00007199	0.00005894	0.00007015	0.00004647
11	0.00008225	0.00006477	0.00007784	0.00005093
12	0.00009216	0.00007083	0.00008187	0.00005461
13	0.00010498	0.00007804	0.00009002	0.00005895
14	0.00011476	0.00008619	0.00009779	0.00006389
15	0.00012540	0.00009271	0.00010423	0.00006829
16	0.00014282	0.00010458	0.00011384	0.00007590
17	0.00015564	0.00011428	0.00012296	0.00008210
18	0.00017346	0.00012126	0.00013020	0.00008711
19	0.00018540	0.00013041	0.00013854	0.00009289
20	0.00020574	0.00014039	0.00014750	0.00009914
21	0.00021779	0.00014707	0.00015429	0.00010362

Let us take a closer look at the resulting tables.

Table 1 shows the time characteristics of stage A. Stage A is of high importance. It allows you to track the behavior of the method when the size of the processed information increases. Larger modules increase the range of processed values. Analyzing the obtained values, we can notice that the growth is linear. This allows us to talk about the stability of the obtained method and the nominal method of approximate rank. Next, for greater clarity, graphs will be presented.

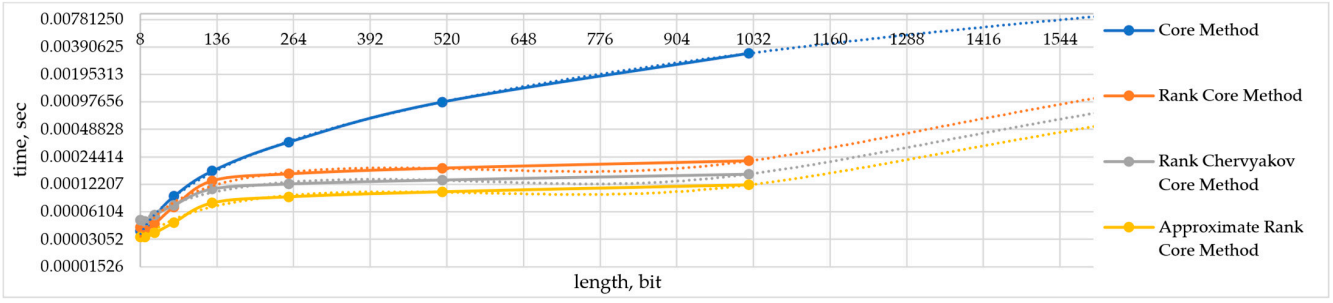
Table 2 shows the time characteristics of stage B. Stage B is just as important as stage A. Some examples of the application of RNS have been presented previously. For example, redundant RNS (RRNS) is used for error correction techniques. The main point of RRNS is that two base systems are used: working and control systems of bases. Thus, if we have a working system consisting of four modules, then the control system can consist of two, four, six, or more modules. Therefore, the study of the behavior of methods depending on the number of modules in the system is also necessary. Based on the obtained data, we can also state the stability of the method.

Based on the data presented in the table, illustrations were prepared. Illustrations allow you to analyze the result more clearly. The graphs are presented on a logarithmic scale for a more convenient perception of the result.

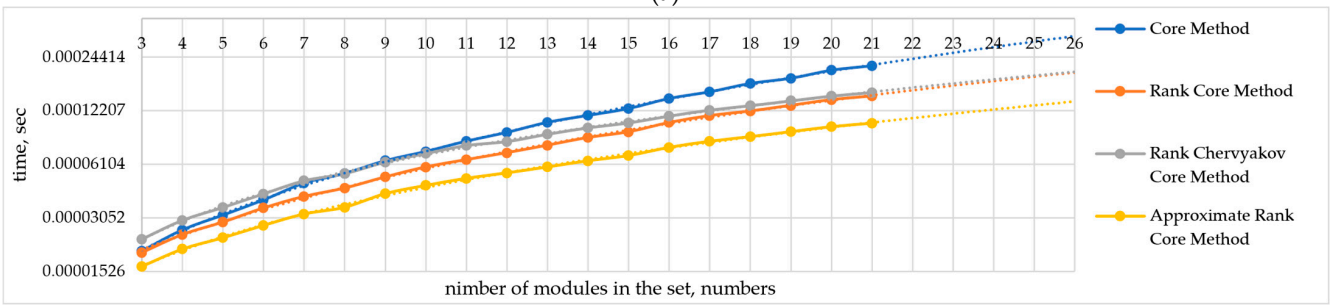
For a more detailed analysis, extrapolation of the obtained values was also carried out using polynomials. The following polynomials were used to carry out the polynomial extrapolation.

Analyzing the results, we can draw the following conclusions. Analyzing the graph Figure 1a, one can notice a tendency that the classical core function method is effective only on small modules (8–32 bits). However, starting from the dimension of 64 bits, the method begins to lose to methods based on the rank of the number. A similar situation is

observed in the graph Figure 1b, where the classical core function method is more efficient than the number rank method on small sets of modules (3–6 modules in a set). Further, its effectiveness decreases. The constructed graphs allow you to analyze in more detail the advantages that the rank of a number gives for RNS methods.



(**a**)



(**b**)

Figure 1. The results of the study: (**a**) stage A; (**b**) stage B.

At stage A, we can observe the following. The performance of the classical method starts to drop sharply starting with numbers 128 bits long, falling, compared with other methods, from 2 to 100–200 times. At the same time, the approximate rank method is about 1.5 times more productive than the others, which proves its effectiveness.

At stage B, we can observe the following. With an increase in the number of modules in the set, a slightly different situation is observed. Here, the method of approximate rank of a number is also the most effective. With a RNS set size of 3 modules, this method is 1.2 times more efficient than other considered methods. However, this gap increases with the size of the set. With a set size of 21, the proposed method is twice as efficient. It is worth noting separately the time characteristic of the method based on the Chervyakov rank. On small sets of modules, this method is the least efficient. However, with the number of modules in the set from 20, its performance becomes almost equal to the performance of the method based on the rank of the number. At the same time, at stage A, the Chervyakov rank method is the second in terms of performance. This is due to the fact that the classical number rank method uses bases. Chervyakov's method uses the usual RNS bases. The use of the usual basis P_i is effective in terms of the speed of its calculation and, in comparison with the method of orthogonal bases, its size depending on the size of the number.

In the course of the study, the method with the best efficiency was determined, namely the method of approximate rank of the core function. To prove the statement put forward, an extrapolation of the values was carried out from Figure 1, and the results obtained were stored at the nearest values.

Stage A:

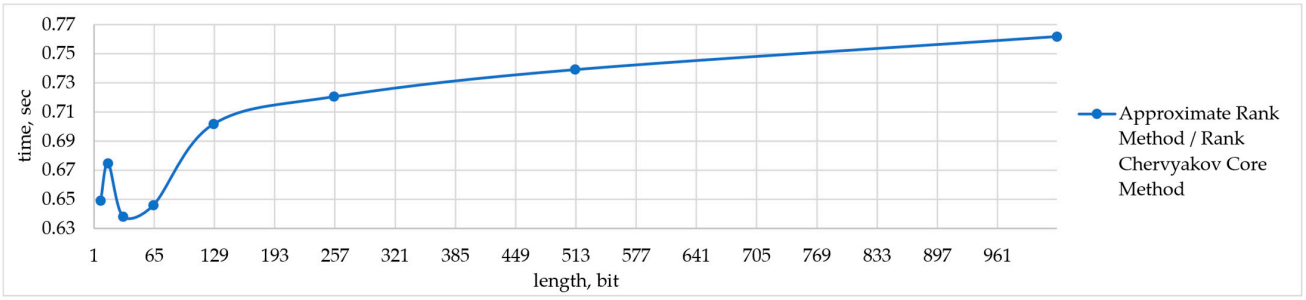
- Core Method $7 \cdot 10^{-13} \cdot x^3 + 2 \cdot 10^{-9} \cdot x^2 + 8 \cdot 10^{-7} \cdot x + 3 \cdot 10^{-5}$ with the accuracy of the approximation $R^2 = 1$;
- Rank Core Method $9 \cdot 10^{-13} \cdot x^3 + 2 \cdot 10^{-9} \cdot x^2 + 9 \cdot 10^{-7} \cdot x + 3 \cdot 10^{-5}$ with the accuracy of the approximation $R^2 = 0.9857$

- Rank Chervyakov Core Method $6 \cdot 10^{-13} \cdot x^3 - 1 \cdot 10^{-9} \cdot x^2 + 6 \cdot 10^{-7} \cdot x + 4 \cdot 10^{-5}$ with the accuracy of the approximation $R^2 = 0.9881$
- Approximate Rank Core Method $4 \cdot 10^{-13} \cdot x^3 - 1 \cdot 10^{-9} \cdot x^2 + 6 \cdot 10^{-7} \cdot x + 4 \cdot 10^{-5}$ with the accuracy of the approximation $R^2 = 0.9885$

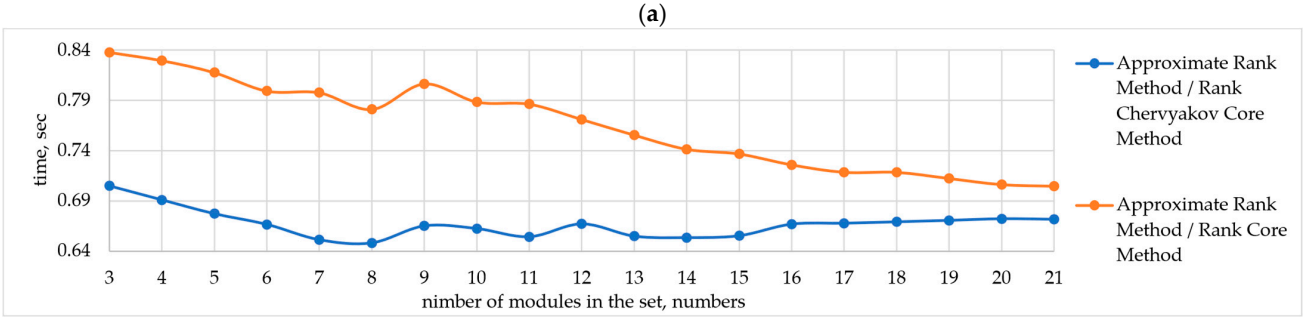
Stage B:

- Core Method $3 \cdot 10^{-9} \cdot x^3 + 2 \cdot 10^{-7} \cdot x^2 + 4 \cdot 10^{-6} \cdot x + 6 \cdot 10^{-6}$ with the accuracy of the approximation $R^2 = 0.9995$
- Rank Core Method $6 \cdot 10^{-10} \cdot x^3 + 5 \cdot 10^{-8} \cdot x^2 + 6 \cdot 10^{-6} \cdot x + 5 \cdot 10^{-6}$ with the accuracy of the approximation $R^2 = 0.9995$
- Rank Chervyakov Core Method $2 \cdot 10^{-9} \cdot x^3 + 2 \cdot 10^{-7} \cdot x^2 + 3 \cdot 10^{-6} \cdot x + 1 \cdot 10^{-5}$ with the accuracy of the approximation $R^2 = 0.9992$
- Approximate Rank Core Method $1 \cdot 10^{-10} \cdot x^3 + 7 \cdot 10^{-8} \cdot x^2 + 3 \cdot 10^{-6} \cdot x + 6 \cdot 10^{-6}$ with the accuracy of the approximation $R^2 = 0.999$

Let us conduct an additional study, namely the relation of the method of approximate rank of the core function, as well as the rank of the core function, and for stage B, additionally, the Chervyakov rank method of the core function. This is necessary because the graphs of the rank and Chervyakov rank methods converge at stage B (Figure 2).



(**a**)



(**b**)

Figure 2. Time difference of methods: (**a**) stage A; (**b**) stage B.

Analyzing the charts, one can notice the following trend. In the case of stage A, the ratio of time characteristics (Figure 2a) tends to unity. Considering the extrapolation also carried out, it can be argued that the approximation method will be the most productive at least up to numbers with a length of 2000 or more bits. For stage B, the ratio of the time characteristics of the methods tends to zero, which means that the approximation method is less prone to performance degradation with an increase in the number of modules in the set. This allows us to talk about its effectiveness from the point of view of practical application.

Thus, the conducted studies prove the effectiveness of the approximate calculation of the rank of a number for the method of determining the sign of a number using the core function. The implementation of a fast operation for determining the sign of a number will allow RNS methods to be applied in distributed computing systems, such as cloud and fog computing, with greater efficiency. This, in turn, improves the reliability and fault tolerance of computing systems.

5. Conclusions and Future Work

The work carried out a study aimed at improving the performance of distributed computing systems. This result is achieved by using RNS, which, due to the properties of a non-positional number system, the independence of calculations on bases with natural parallelism, and corrective abilities, makes it possible to increase reliability and fault tolerance. However, RNS has several disadvantages associated with non-modular operations. In this paper, we propose a fast method for performing the operation of determining the sign of a number based on the RNS Core Function. In addition, an experimental analysis was carried out, during which the effectiveness of the proposed method was established.

In addition, an extrapolation of the time characteristics of the methods was carried out, and a study was made of the relationship of the proposed method with the methods closest in performance. Such a study allows for a performance analysis in a wider range. The data obtained during extrapolation are achieved based on the calculation of the confidence values of the approximation. The results obtained allow us to state that the proposed method is applicable for building data transmission models.

Thus, this paper presents a method for quickly determining the sign of a number from the core function based on the approximate rank of a number. For this method, experimental confirmation of the effectiveness was obtained based on a computational experiment. In the future, it is planned to study this method in practical application.

The result obtained allows us to extend the use of RNS for FC and other distributed computing systems. In the future, we plan to develop a full-fledged computing system based on RNS for FC. To do this, it is necessary to implement such a non-modular operation as division. The implementation of an efficient division algorithm will allow you to perform full-fledged arithmetic on RNS numbers. The development of a complete computing system will expand the use of FC for SC and IoT.

Author Contributions: E.S.: conceptualization, methodology, software, validation, research, writing; N.K.: methodology, research, writing, supervision; M.B.: conceptualization, methodology, research, writing, supervision; A.N.: conceptualization, methodology, research, writing. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by the Russian Science Foundation Grant No. 22-71-10046, https://rscf.ru/en/project/22-71-10046/.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yi, S.; Li, C.; Li, Q. A Survey of Fog Computing: Concepts, Applications and Issues. In Proceedings of the 2015 Workshop on Mobile Big Data, Hangzhou, China, 21 June 2015; pp. 37–42.
2. Priyadarshini, R.; Barik, R.K.; Dubey, H. Deepfog: Fog Computing-Based Deep Neural Architecture for Prediction of Stress Types, Diabetes and Hypertension Attacks. *Computation* **2018**, *6*, 62. [CrossRef]
3. Al-Turjman, F.; Abujubbeh, M. IoT-Enabled Smart Grid via SM: An Overview. *Future Gener. Comput. Syst.* **2019**, *96*, 579–590. [CrossRef]
4. Su, K.; Li, J.; Fu, H. Smart City and the Applications. In Proceedings of the 2011 International Conference on Electronics, Communications and Control (ICECC), Ningbo, China, 9–11 September 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1028–1031.
5. Hua, H.; Li, Y.; Wang, T.; Dong, N.; Li, W.; Cao, J. Edge Computing with Artificial Intelligence: A Machine Learning Perspective. *ACM Comput. Surv.* **2023**, *55*, 1–35. [CrossRef]
6. Ren, C.; Lyu, X.; Ni, W.; Tian, H.; Song, W.; Liu, R.P. Distributed Online Optimization of Fog Computing for Internet of Things under Finite Device Buffers. *IEEE Internet Things J.* **2020**, *7*, 5434–5448. [CrossRef]
7. Chang, Z.; Liu, L.; Guo, X.; Sheng, Q. Dynamic Resource Allocation and Computation Offloading for Iot Fog Computing System. *IEEE Trans. Ind. Inform.* **2020**, *17*, 3348–3357. [CrossRef]

8. Abouaomar, A.; Cherkaoui, S.; Kobbane, A.; Dambri, O.A. A Resources Representation for Resource Allocation in Fog Computing Networks. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa Village, HI, USA, 9–13 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
9. Huang, C.; Wang, H.; Zeng, L.; Li, T. Resource Scheduling and Energy Consumption Optimization Based on Lyapunov Optimization in Fog Computing. *Sensors* **2022**, *22*, 3527. [CrossRef] [PubMed]
10. Isupov, K. High-Performance Computation in Residue Number System Using Floating-Point Arithmetic. *Computation* **2021**, *9*, 9. [CrossRef]
11. Chang, C.-H.; Molahosseini, A.S.; Zarandi, A.A.E.; Tay, T.F. Residue Number Systems: A New Paradigm to Datapath Optimization for Low-Power and High-Performance Digital Signal Processing Applications. *IEEE Circuits Syst. Mag.* **2015**, *15*, 26–44. [CrossRef]
12. Cluzeau, M. Block Code Reconstruction Using Iterative Decoding Techniques. In Proceedings of the 2006 IEEE International Symposium on Information Theory, Washington, DC, USA, 9–14 July 2006; IEEE: Piscataway, NJ, USA, 2006; pp. 2269–2273.
13. Jafarkhani, H. A Quasi-Orthogonal Space-Time Block Code. *IEEE Trans. Commun.* **2001**, *49*, 1–4. [CrossRef]
14. Muder, D.J. Minimal Trellises for Block Codes. *IEEE Trans. Inf. Theory* **1988**, *34*, 1049–1053. [CrossRef]
15. Solomon, G.; Tilborg, H.C.A. A Connection between Block and Convolutional Codes. *SIAM J. Appl. Math.* **1979**, *37*, 358–369. [CrossRef]
16. Tarokh, V.; Jafarkhani, H.; Calderbank, A.R. Space-Time Block Codes from Orthogonal Designs. *IEEE Trans. Inf. Theory* **1999**, *45*, 1456–1467. [CrossRef]
17. Chen, C.-L.; Hsiao, M.Y. Error-Correcting Codes for Semiconductor Memory Applications: A State-of-the-Art Review. *IBM J. Res. Dev.* **1984**, *28*, 124–134. [CrossRef]
18. Choukroun, Y.; Wolf, L. Error Correction Code Transformer. *arXiv* **2022**, arXiv:2203.14966.
19. Knill, E.; Laflamme, R. Theory of Quantum Error-Correcting Codes. *Phys. Rev. A* **1997**, *55*, 900. [CrossRef]
20. Peterson, W.W.; Peterson, W.; Weldon, E.J.; Weldon, E.J. *Error-Correcting Codes*; MIT Press: Cambridge, MA, USA, 1972.
21. Sellers, F. Bit Loss and Gain Correction Code. *IRE Trans. Inf. Theory* **1962**, *8*, 35–38. [CrossRef]
22. Lamehamedi, H.; Szymanski, B.; Shentu, Z.; Deelman, E. Data Replication Strategies in Grid Environments. In Proceedings of the Fifth International Conference on Algorithms and Architectures for Parallel Processing, Beijing, China, 23–25 October 2002; IEEE: Piscataway, NJ, USA, 2002; pp. 378–383.
23. Babenko, M.; Tchernykh, A.; Pulido-Gaytan, B.; Cortés-Mendoza, J.M.; Shiryaev, E.; Golimblevskaia, E.; Avetisyan, A.; Nesmachnow, S. RRNS Base Extension Error-Correcting Code for Performance Optimization of Scalable Reliable Distributed Cloud Data Storage. In Proceedings of the 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Portland, OR, USA, 17–21 June 2021; pp. 548–553.
24. Babenko, M.; Nazarov, A.; Tchernykh, A.; Pulido-Gaytan, B.; Cortés-Mendoza, J.M.; Vashchenko, I. Algorithm for Constructing Modular Projections for Correcting Multiple Errors Based on a Redundant Residue Number System Using Maximum Likelihood Decoding. *Program Comput. Soft* **2021**, *47*, 839–848. [CrossRef]
25. Di Claudio, E.D.; Orlandi, G.; Piazza, F. Parallel Error Correction Algorithm in RNS VLSI Digital Circuits. In Proceedings of the ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing, New York, NY, USA, 11–14 April 1988; IEEE: Piscataway, NJ, USA, 1988; pp. 1738–1741.
26. Gladkov, A.; Gladkova, N.; Kucherov, N. Analytical Review of Methods for Detection, Localization and Error Correction in the Residue Number System. In Proceedings of the International Conference on Mathematics and its Applications in New Computer Systems, Munich, Germany, 25–28 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 507–514.
27. Shiryaev, E.; Bezuglova, E.; Babenko, M.; Tchernykh, A.; Pulido-Gaytan, B.; Cortés-Mendoza, J.M. Performance Impact of Error Correction Codes in RNS with Returning Methods and Base Extension. In Proceedings of the 2021 International Conference Engineering and Telecommunication (En&T), Online, 24–25 November 2021; pp. 1–5.
28. Tay, T.F.; Chang, C.-H. A New Algorithm for Single Residue Digit Error Correction in Redundant Residue Number System. In Proceedings of the 2014 IEEE International Symposium on Circuits and Systems (ISCAS), Melbourne, Australia, 1–5 June 2014; pp. 1748–1751.
29. Pontarelli, S.; Cardarilli, G.C.; Re, M.; Salsano, A. A Novel Error Detection and Correction Technique for RNS Based FIR Filters. In Proceedings of the 2008 IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems, Boston, MA, USA, 1–3 October 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 436–444.
30. Mohan, P.A.; Ananda Mohan, P.V. Error Detection, Correction and Fault Tolerance in RNS-Based Designs. In *Residue Number Systems: Theory and Applications*; Birkhäuser: Cham, Switzerland, 2016; pp. 163–175.
31. Jenkins, W.K. Residue Number System Error Checking Using Expanded Projection. *Electron. Lett.* **1982**, *18*, 927–928. [CrossRef]
32. Tay, T.F.; Chang, C.-H. Fault-Tolerant Computing in Redundant Residue Number System. In *Embedded Systems Design with Special Arithmetic and Number Systems*; Springer: Cham, Switzerland, 2017; pp. 65–88.
33. Garg, N.; Yadav, P. Comparison of Asymmetric Algorithms in Cryptography. *J. Comput. Sci. Mob. Comput.* **2014**, *3*, 1190–1196.
34. Milanov, E. The RSA Algorithm. RSA Lab. 2009, 1–11. Available online: https://sites.math.washington.edu/~morrow/336_09/papers/Yevgeny.pdf (accessed on 27 June 2023).
35. Boneh, D.; Franklin, M. Efficient Generation of Shared RSA Keys. In Proceedings of the Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 1997; Proceedings 17. Springer: Berlin/Heidelberg, Germany, 1997; pp. 425–439.

36. da Silva, J.C.L. Factoring Semiprimes and Possible Implications for RSA. In Proceedings of the 2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel, Eilat, Israel, 17–20 November 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 182–183.
37. Park, J.M.; Chong, E.K.; Siegel, H.J. Constructing Fair-Exchange Protocols for E-Commerce via Distributed Computation of RSA Signatures. In Proceedings of the Twenty-Second Annual Symposium on Principles of Distributed Computing, Boston, MA, USA, 13–16 July 2003; pp. 172–181.
38. Jamaludin, J.; Romindo, R. Implementation of Combination Vigenere Cipher and RSA in Hybrid Cryptosystem for Text Security. *Int. J. Inf. Syst. Technol.* **2020**, *4*, 471–481.
39. Ren, W.; Miao, Z. A Hybrid Encryption Algorithm Based on DES and RSA in Bluetooth Communication. In Proceedings of the 2010 Second International Conference on Modeling, Simulation and Visualization Methods, Las Vegas, NV, USA, 12–15 July 2010; pp. 221–225.
40. Kuppuswamy, P.; Al-Khalidi, S.Q.Y. Hybrid Encryption/Decryption Technique Using New Public Key and Symmetric Key Algorithm. *Int. J. Inf. Comput. Secur.* **2014**, *6*, 372–382. [CrossRef]
41. Ramaraj, E.; Karthikeyan, S.; Hemalatha, M. A Design of Security Protocol Using Hybrid Encryption Technique (AES-Rijndael and RSA). *Int. J. Comput. Internet Manag.* **2009**, *17*, 34–43.
42. Jintcharadze, E.; Iavich, M. Hybrid Implementation of Twofish, AES, ElGamal and RSA Cryptosystems. In Proceedings of the 2020 IEEE East-West Design & Test Symposium (EWDTS), Varna, Bulgaria, 4–7 September 2020; pp. 1–5.
43. Alkady, Y.; Habib, M.I.; Rizk, R.Y. A New Security Protocol Using Hybrid Cryptography Algorithms. In Proceedings of the 2013 9th International Computer Engineering Conference (ICENCO), Giza, Egypt, 28–29 December 2013; pp. 109–115.
44. Beimel, A. Secret-Sharing Schemes: A Survey. In Proceedings of the Coding and Cryptology, Qingdao, China, 30 May–3 June 2011; Chee, Y.M., Guo, Z., Ling, S., Shao, F., Tang, Y., Wang, H., Xing, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 11–46.
45. Dong, X. A Multi-Secret Sharing Scheme Based on the CRT and RSA. *Int. J. Electron. Inf. Eng.* **2015**, *2*, 47–51.
46. Mignotte, M. How to Share a Secret. In Proceedings of the Workshop on Cryptography, Burg Feuerstein, Germany, 29 March–2 April 1982; Springer: Berlin/Heidelberg, Germany, 1982; pp. 371–375.
47. Asmuth, C.; Bloom, J. A Modular Approach to Key Safeguarding. *IEEE Trans. Inf. Theory* **1983**, *29*, 208–210. [CrossRef]
48. Gomathisankaran, M.; Tyagi, A.; Namuduri, K. HORNS: A Homomorphic Encryption Scheme for Cloud Computing Using Residue Number System. In Proceedings of the 2011 45th Annual Conference on Information Sciences and Systems, Baltimore, MD, USA, 23–25 March 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–5.
49. Muhammed, K.J.; Isiaka, R.M.; Asaju-Gbolagade, A.W.; Adewole, K.S.; Gbolagade, K.A. Improved Cloud-Based N-Primes Model for Symmetric-Based Fully Homomorphic Encryption Using Residue Number System. In *Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics: Theories and Applications*; Chiroma, H., Abdulhamid, S.M., Fournier-Viger, P., Garcia, N.M., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 197–216, ISBN 978-3-030-66288-2.
50. Al Badawi, A.; Polyakov, Y.; Aung, K.M.M.; Veeravalli, B.; Rohloff, K. Implementation and Performance Evaluation of RNS Variants of the BFV Homomorphic Encryption Scheme. *IEEE Trans. Emerg. Top. Comput.* **2021**, *9*, 941–956. [CrossRef]
51. Lee, J.-W.; Lee, E.; Lee, Y.; Kim, Y.-S.; No, J.-S. High-Precision Bootstrapping of RNS-CKKS Homomorphic Encryption Using Optimal Minimax Polynomial Approximation and Inverse Sine Function. In Proceedings of the Advances in Cryptology—EUROCRYPT 2021, Zagreb, Croatia, 17–21 October 2021; Canteaut, A., Standaert, F.-X., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 618–647.
52. Kim, A.; Papadimitriou, A.; Polyakov, Y. Approximate Homomorphic Encryption with Reduced Approximation Error. In Proceedings of the Topics in Cryptology–CT-RSA 2022: Cryptographers' Track at the RSA Conference 2022, Virtual Event, 1–2 March 2022; Proceedings. Springer: Berlin/Heidelberg, Germany, 2022; pp. 120–144.
53. Nykolaychuk, Y.M.; Yakymenko, I.Z.; Vozna, N.Y.; Kasianchuk, M.M. Residue Number System Asymmetric Cryptoalgorithms. *Cybern. Syst. Anal.* **2022**, *58*, 611–618. [CrossRef]
54. Kalmykov, I.A.; Pashintsev, V.P.; Tyncherov, K.T.; Olenev, A.A.; Chistousov, N.K. Error-Correction Coding Using Polynomial Residue Number System. *Appl. Sci.* **2022**, *12*, 3365. [CrossRef]
55. Givaki, K.; Khonsari, A.; Gholamrezaei, M.H.; Gorgin, S.; Najafi, M.H. A Generalized Residue Number System Design Approach for Ultra-Low Power Arithmetic Circuits Based on Deterministic Bit-Streams. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2023**, *1*, 14. [CrossRef]
56. Garner, H.L. The Residue Number System. In Proceedings of the Western Joint Computer Conference, San Francisco, CA, USA, 3–5 March 1959; pp. 146–153.
57. Pei, D.; Salomaa, A.; Ding, C. *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography*; World Scientific: Singapore, 1996.
58. Brown, W.S.; Traub, J.F. On Euclid's Algorithm and the Theory of Subresultants. *J. ACM* **1971**, *18*, 505–514. [CrossRef]
59. Kocherov, Y.N.; Samoylenko, D.V.; Koldaev, A.I. Development of an Antinoise Method of Data Sharing Based on the Application of a Two-Step-Up System of Residual Classes. In Proceedings of the 2018 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), Vladivostok, Russian, 2–4 October 2018; pp. 1–5.
60. Gbolagade, K.A.; Cotofana, S.D. An O(n) Residue Number System to Mixed Radix Conversion Technique. In Proceedings of the 2009 IEEE International Symposium on Circuits and Systems, Taipei, Taiwan, 24–27 May 2009; pp. 521–524.
61. Akushsky, I.Y.; Akushsky, V.M.; Pak, I.T. About the New Positional Characteristic of the Non-Positional Code and Its Application. In *Theory of Coding and Optimization of Complex Systems*; Nauka: Alma-Ata, Kazakhstan, 1977; pp. 8–16.

62. Akushsky, I.Y.; Burtsev, V.M.; Pak, N.T. Calculation of the Positional Characteristic (Core) of the Non-Positional Code. In *Theory of Coding and Optimization of Complex Systems*; Nauka: Alma-Ata, Kazakhstan, 1977; pp. 17–25.
63. Chervyakov, N.I.; Averbukh, V.M.; Babenko, M.G.; Lyakhov, P.A.; Gladkov, A.V.; Gapochkin, A.V. An Approximate Method for Performing Non-Modular Operations in a System of Residual Classes. *Fundam. Res.* **2012**, *6*, 189–193.
64. Babenko, M.; Golimblevskaia, E. About One Property of Number Rank in RNS. In Proceedings of the 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), St. Petersburg, FL, USA, 26–29 January 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 212–216.
65. NTL: A Library for Doing Number Theory. Available online: https://libntl.org/ (accessed on 23 May 2023).