Euclidean Division Method for the Homomorphic Scheme CKKS

Mikhail Babenko
North-Caucasus Federal University
Stavropol, Russia
Ivannikov Institute for System Programming
Moscow, Russia
mgbabenko@ncfu.ru

Elena Golimblevskaia
North-Caucasus Federal University
Stavropol, Russia
elena.golimblevskaya@gmail.com

2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus) | 978-1-6654-0476-1/20/$31.00 ©2021 IEEE | DOI: 10.1109/ElConRus51938.2021.9396347

***Abstract*—The use of cloud computing can reduce the economic costs of maintaining IT infrastructure, but at the same time, the likelihood of confidential data theft increases. To reduce the likelihood of it, cloud computing uses homomorphic encryption. However, a homomorphic cipher only allows adding and multiplying encrypted numbers; in some cases, a division operation is required to implement algorithms. To implement the division operation with encrypted numbers it is necessary to implement the encrypted number comparison operation. Considering that the operation of comparing encrypted numbers is carried out using numerical methods, it is necessary to adapt the existing algorithms for Euclidean division. In this article, we propose a two-stage algorithm for Euclidean division of numbers encrypted using the CKKS scheme and investigate its properties.**

Keywords—Homomorphic encryption; Euclidean division; cloud computing; comparison operation; modular arithmetic

I. INTRODUCTION

Cloud storage and computing resources have evolved rapidly in recent years. More and more organizations and users are choosing to use them for a variety of tasks [1]. However, secure cloud computing is one of the most important challenges at the moment, as it allows sensitive data to be processed in distributed storage without disclosing the data itself. This will allow organizations to save infrastructure costs and use their resources more efficiently. Full data security is especially important if the data is confidential, for example, medical data, financial data, etc. One approach that has made such computations possible is homomorphic encryption.

Homomorphic encryption is a type of encryption that allows performing computations on encrypted data without decrypting them and obtaining a result corresponding to the result of the same operations on unencrypted data [2]. The big breakthrough in homomorphic encryption came in 2009 when Gentry introduced the first fully homomorphic encryption (FHE) scheme [3, 4]. Since then, many modifications have been presented to his scheme, and several new FHE schemes have been created [5, 6, 7, 8, 9, 10, 11]. Fully homomorphic encryption differs from other types of homomorphic encryption in that it allows performing both addition and multiplication operations on data, while other types allow performing only one of these operations.

Since 2009, many researchers have been conducting research on the modification of the scheme proposed by Gentry. The authors suggested various ways of modifying such schemes in order to increase the speed of performing operations with ciphertexts. The most popular were the use of cryptographic lattices. However, the authors of the CKKS scheme came to an interesting solution, they use the plaintext over a complex field, and also introduce mathematics into the Residue Number System (RNS) to speed up modular operations. However, RNS has its drawbacks, such as the high computational complexity of non-modular operations.

Considering that the effective use of FHE is still limited by the fact that performing non-modular operations such as division, comparison of numbers, etc., requires great computational complexity in FHE, we propose the Euclidean division method for modifying the FHE CKKS scheme. Considering a variant of the CKKS scheme based on the Residue Number System, we can improve performance by introducing an approximate method for determining the sign of a number for a division operation in RNS, which, due to the peculiarity of constructing CKKS ciphertexts, also applies to FHE, without introducing additional parameters.

The work is structured as follows. In Section II, we conducted research on other works on the division of FHE encrypted values. In Section III, we provide the theoretical foundations of RNS and how it performs modular operations. In Section IV we describe the CKKS scheme, its properties and parameters. In Section V, we provide a theoretical basis for our method based on the function of determining the sign of a number. In Section VI, we describe the algorithm for the operation of our method. In section VII, we present the conclusions obtained during this work.

II. RELATED WORK

Many researchers have wondered how to create efficient division methods in homomorphic encryption schemes. In FHE schemes, the possibility of division is due to the fact that the arithmetic operations of addition and multiplication are homomorphic, therefore, other arithmetic operations also have a homomorphism.

T. Veugen in his work [12] proposes a new solution in the field of integer division. In his work, the author is based on the assumption that to perform the division operation in a distributed system, an additional protocol is required for transferring data to the server. To solve this problem, the author proposes a new technique based on data separation. This operation allows processing data with less computational complexity, which in turn allows adding additional protocols in

978-1-6654-0476-1/21/$31.00 ©2021 IEEE

a distributed environment with low network load, namely, fine and approximate dividing protocols.

However, the effectiveness of this method is justified only on the basis of data transfer protocols. This is not the case within the system. Calculating quotient has the same performance, only with smaller data.

N.I. Chervyakov in his work [13] proposes a method for dividing numbers in RNS by system moduli. Despite the rather elegant solution, this method is difficult to apply, since in this case it is necessary that the numbers be relatively prime. Also, if the number is absent in the system of moduli, a computationally complex operation of expanding the basis of RNS is necessary, and if the number is not prime, dividing it into prime factors, which also affects the performance.

N. I. Chervyakov in [14] proposed a method devoted to integer division in RNS. In this paper, algorithms based on the methods of Fermat and Newton [15, 16], carried over to RNS, are presented. This allows dividing RNS by almost any number, since Fermat's descent is used, on the basis of which an approximate divisor is selected. However, this method requires a large number of iterations as it relies on recursion.

Therefore, in our work, we propose a modification of the division operation in the homomorphic encryption scheme CKKS, based on its RNS properties, using an approximate function for determining the sign of a number.

III. Residue Number System

Residue Number System (RNS) is a non-weighted number system [17]. The number X in RNS is presented as $X = \{x_1, x_2, \dots, x_n\}$, where x_i is the remainder of dividing the number X by the corresponding modulus of the RNS basis $\{p_1, p_2, \dots, p_n\}$. A number in RNS can be unambiguously represented if $X \in [0, P-1]$, where $P = p_1 \cdot p_2 \cdot \dots \cdot p_n$.

To convert a number from a weighted number system to RNS, the following operations are performed:

$$\begin{aligned} x_1 &\equiv X (mod\, p_1) \\ x_2 &\equiv X (mod\, p_2) \\ &\dots\dots\dots\dots\dots\dots \\ x_n &\equiv X (mod\, p_n) \end{aligned} \quad (1)$$

Due to the fact that the remainders that make up a number in RNS are independent of each other, the calculation of modular operations can be performed in parallel using the formula:

$$\begin{aligned} A * B = (\alpha_1, \alpha_2, \dots, \alpha_n) * (\beta_1, \beta_2, \dots, \beta_n) = \\ = ((\alpha_1 * \beta_1)\, mod\, p_1), ((\alpha_2 * \beta_2)\, mod\, \rho_2), \dots, ((x_n \\ * \beta_n)\, mod\, p_n) \end{aligned} \quad (2)$$

where $*$ is the operation of addition, subtraction or multiplication.

Operations such as division and comparison in RNS are computationally complex. However, it is worth noting that modular division is possible under several restrictions, but strictly for integers, and when dividing without remainder. Such division is possible according to the following formula:

$$\begin{aligned} \frac{A}{B} = ((a_1 * b_1^{-1}) mod\, p_1, (a_2 * b_2^{-1}) mod\, p_2, \dots, \\ (a_n * b_n^{-1}) mod\, p_n) \end{aligned} \quad (3)$$

As already mentioned, the quotient must be uniquely whole, in addition, for division, it is necessary that there are no remainders equal to zero in the RNS system. In the real world of RNS operation in FHE, such a situation is unlikely. Therefore, division in RNS cannot be called a modular operation.

Comparison in RNS is to determine the sign of the number after subtracting the number b from a. If the difference of numbers is positive, then $a > b$, if 0, then $a = b$, if the difference is negative, then $a < b$.

Also, the use of redundancy in RNS allows creating robust data structures and performing error correction. Thus, the use of RNS in homomorphic encryption has a number of advantages, the main of which is the acceleration of computations, which is essential for homomorphic encryption.

IV. CKKS Scheme and It's Properties

We consider an RNS version of the CKKS scheme proposed by Cheon, Kim, Kim and Song [18]. This scheme is designed to encrypt complex numbers, and it provides arithmetic over $C^{N/2}$.

In the CKKS scheme, complex numbers are encrypted on integer polynomials, so the plaintext and ciphertext space is as follows:

$$Z_Q[X]/(X^N + 1) \quad (4)$$

where N is a power of 2.

The batch encoding is performed as follows:

$$C^{\frac{N}{2}} \leftrightarrow Z_Q[X]/(X^N + 1) \quad (5)$$

so that

$$decode(encode(m_1) \otimes encode(m_2)) \approx m_1 \odot m_2 \quad (6)$$

where $\otimes$ is component multiplication and $\odot$ is non-cyclic convolution.

The CKKS scheme uses the following security parameters:

$N = 2^{\log N}$ is a ring dimension. It determines the degree of the cyclotomic polynomial, the number of coefficients in the plaintext and in the ciphertext polynomials.

Q is a ciphertext modulus. Q is a product of small coprime modulus $q_i \equiv 1 \bmod 2N$. It affects security and performance, and closely related to N.

σ is a variance used for error polynomials.

To encode complex numbers into integer polynomials, the $scale$ parameter is used.

$scale$ is a plaintext scale. It is required for encoding complex numbers on polynomials with integer coefficients. This parameter is the power of two. $scale$ influences accuracy of computations and the number of operation that can be

performed before overflow. This parameter does not have security impact.

Main functions of CKKS:

- CKKS.Setup(): Set a ring dimension N, a ciphertext modulus Q, a key and error distributions χ and Ω over R correspondingly.
- SymEnc(m, sk): $m \in R$ is an input plaintext, and $\text{sk} = s \in R_{qp}$ is a secret key. $a \leftarrow U(R_{qp})$ and $e \leftarrow \Omega$. $b = -a \cdot s + e \in R_{qp}$. Return the ciphertext $\text{ct} = (c_0, c_1) = (b, a)$.
- CKKS.KeyGen(): Secret key $\text{sk} = s$, where $s \leftarrow \chi$, and public key $\text{pk} = \text{SymEnc}(0, \text{sk})$;
- CKKS.Dec(ct, sk): $\text{ct} = (c_0, c_1) \in R_{q_\ell}^2$ is a ciphertext at the ℓ-th level, return $c_0 + c_1 \cdot s (\text{mod} q_\ell)$.
- CKKS.Add(ct_0, ct_1): Return $\text{ct} = \text{ct}_0 + ct_1$.
- CKKS.Mul(ct_0, ct_1): Return $\text{ct} = \text{ct}_0 \cdot ct_1$.

V. Calculation of the Number's Sign

The proposed division method for numbers presented in RNS and encrypted in CKKS contains a key function – determining the sign of a number.

To determine the sign of a number in homomorphic encryption, we will use the method proposed in [19].

This method consists in approximating the sign of the number by a composition of polynomials $f \circ ... \circ f \circ g \circ ... \circ g$.

The polynomials of the family $f(x)$ д must meet the following conditions:

1. It is an odd function;
2. $f(1) = 1, f(-1) = -1$;
3. $f'(x) = c(1-x)^n(1+x)^n$ for some constant $c > 0$

Thus, they can be calculated by the formula

$$f_n(x) = \sum_{i=0}^{n} \frac{1}{4^i} \cdot \binom{2i}{i} \cdot x(1-x^2)^i. \qquad (7)$$

In the original paper, the following polynomials were calculated:

- $f_1(x) = -\frac{1}{2}x^3 + \frac{3}{2}x$
- $f_2(x) = \frac{3}{8}x^5 - \frac{10}{8}x^3 + \frac{15}{8}x$
- $f_3(x) = -\frac{5}{16}x^7 + \frac{21}{16}x^5 - \frac{35}{16}x^3 + \frac{35}{16}x$
- $f_4(x) = \frac{35}{128}x^9 - \frac{180}{128}x^7 + \frac{378}{128}x^5 - \frac{420}{128}x^3 + \frac{315}{128}x$

The polynomials $g(x)$ are minimax polynomials and serve to accelerate the approximation of the final function to the sign function. They must meet the following conditions:

1. It is an odd function
2. $\exists 0 < \delta < 1 \text{ s.t. } x < g(x) \leq 1 \text{ for all } x \in (0, \delta], \text{and } g([\delta, 1]) \subseteq [1 - \tau, 1]$

Thus, in [19], the following polynomials $g(x)$ were given:

- $g_1(x) = -\frac{1359}{2^{10}} \cdot x^3 + \frac{2126}{2^{10}} \cdot x$
- $g_2(x) = \frac{3796}{2^{10}} \cdot x^5 - \frac{6108}{2^{10}} \cdot x^3 + \frac{3334}{2^{10}} \cdot x$
- $g_3(x) = -\frac{12860}{2^{10}} \cdot x^7 + \frac{25614}{2^{10}} \cdot x^5 - \frac{16577}{2^{10}} \cdot x^3 + \frac{4589}{2^{10}} \cdot x$
- $g_4(x) = \frac{46623}{2^{10}} \cdot x^9 - \frac{113492}{2^{10}} \cdot x^7 + \frac{97015}{2^{10}} \cdot x^5 - \frac{34974}{2^{10}} \cdot x^3 + \frac{5850}{2^{10}} \cdot x$

We found out that the most effective option is the strict sequence of composition $f_n(g_n(x))$. Thus, we define a function for determining the sign of a number:

$$sign(x) = f_4(g_4(x)) \qquad (8)$$

VI. Euclidian Division Method

The proposed improved algorithm for dividing modular numbers encrypted in CKKS consists of the following steps:

1. Let a be the dividend and b the divisor. Then we calculate $sign(a-b)$. If $sign(a-b) = -1$, then the division process ends, and $\left\lfloor \frac{a}{b} \right\rfloor = 0$. If $sign(a-b) = 0$, then $\left\lfloor \frac{a}{b} \right\rfloor = 1$, the division process also ends. Otherwise, if $sign(a-b) = 1$, then the search for the highest power 2^k is performed when the quotient is approximated by a binary code.
2. Shift $sign(a-b)$ to the left until the most significant bit is transferred to the sign bit. Then, the number of shifts will determine the highest degree.
3. Next, the constant 2^k is taken, where k is the highest degree of the digit. We multiply the constant by the divisor b, getting $sign_1(b) = b2^k$ and compare it with $2^j \ mod \ p_i$, where $i = \overline{1,n}, 1 \leq j \leq \log_2 P$.
4. Calculate $\Delta_i = sign(a-b) - sign_1(a-b)$. If "1" is written in the sign bit Δ_i, then the degree is discarded, if "0" is written, then 2^k is added to the quotient.
5. Next, the digits $sign_1$ are shifted to the right, and the next 2^{k-1} is checked.
6. Then $\Delta_2 = \Delta_1 - sing_1(a-b)$ is calculated and steps 4-5 are repeated.
7. Similarly, we check all the remaining members of the series up to degree zero. The resulting remainder $\Delta_i = \Delta_{i-1} - sing_{i-1}(a-b) \approx 0$. In the case when the divisor takes the minimum value and the dividend takes the maximum, then the threshold Δ_i can be taken greater than zero, which will reduce the number of iterations when dividing a large dividend and a small divisor. In the process of these transformations, we summarize all the allowed members of the series.

VII. Conclusion

In the course of the research carried out in the work, the RNS was analyzed for the possibility of performing non-

modular operations such as division and comparison. This was necessary to modify the FHE scheme. It was found that to perform non-modular operations in RNS, it is necessary to introduce a function for determining the sign of a number. During the analysis of the FHE CKKS scheme, it was found that, despite the fact that it is in the field of complex numbers, it is also full-RNS, which allows applying the same methods as in RNS.

The division operation has attracted the most attention. Researchers have established a number of polynomials [19], applicable to calculate the sign of the RNS number. Thus, using this calculation method for division, a fairly accurate approximate division can be performed. Its peculiarity lies in the fact that even if the quotient is obtained with a remainder, division is performed, but without it. In most situations, as, for example, in ciphertexts, this does not do much harm, since this error is added to the general "noise" in the encoded information.

Based on these conclusions, it can be concluded that the use of such a division in the RNS is possible, and it will speed up the execution of operations with ciphertext, since, based on the number sign function, it is possible to compare values in ciphertexts with high speed. This allows the CKKS scheme to be used in distributed computing systems with higher performance. For example, in cloud structures, improving the performance of the comparison operation will increase the speed of user access to their data. Faster division will speed up the system as a whole. Thus, the modification of the FHE CKKS scheme will increase the popularity of homomorphic encryption for use in various structures.

ACKNOWLEDGMENT

The work was supported by the Russian Science Foundation Grant No. 19-71-10033.

REFERENCES

[1] S. Kamara, K. Lauter. “Cryptographic cloud storage”, International Conference on Financial Cryptography and Data Security, Berlin, Heidelberg, 2010, pp. 136-149.

[2] M. Ogburn, C. Turner, P. Dahal, “Homomorphic encryption”, Procedia Computer Science, 2013, vol. 20, pp. 502-509.

[3] F. Armknecht et al., “A Guide to Fully Homomorphic Encryption,” IACR Cryptol. ePrint Arch., 2015, vol. 2015, p. 1192,.

[4] C. Gentry and D. Boneh, A fully homomorphic encryption scheme, vol. 20, no. 9. Stanford university Stanford, 2009

[5] K. Han, S. Hong, J. H. Cheon, and D. Park, “Logistic Regression on Homomorphic Encrypted Data at Scale,” Proc. AAAI Conf. Artif. Intell., vol. 33, pp. 9466–9471, Jul. 2019.

[6] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, “Fully Homomorphic Encryption over the Integers,” Lecture Notes in Computer Science, vol. 6110, pp. 24–43, 2010.

[7] A. Kim, Y. Song, M. Kim, K. Lee, and J. H. Cheon, “Logistic regression model training based on the approximate homomorphic encryption,” BMC Med. Genomics, vol. 11, no. S4, p. 83, Oct. 2018.

[8] D. Kim, Y. Son, D. Kim, A. Kim, S. Hong, and J. H. Cheon, “Privacy-preserving approximate GWAS computation based on homomorphic encryption,” BMC Med. Genomics, vol. 13, no. S7, p. 77, Jul. 2020.

[9] A. Qaisar Ahmad Al Badawi, Y. Polyakov, K. M. M. Aung, B. Veeravalli, and K. Rohloff, “Implementation and Performance Evaluation of RNS Variants of the BFV Homomorphic Encryption Scheme,” IEEE Trans. Emerg. Top. Comput., pp. 1–1, 2019.

[10] J. H. Cheon, D. Kim, and D. Kim, “Efficient Homomorphic Comparison Methods with Optimal Complexity,” IACR Cryptol. ePrint Arch., vol. 2019, p. 1234, 2019.

[11] H. Chen, I. Chillotti, and Y. Song, “Improved Bootstrapping for Approximate Homomorphic Encryption,” Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2019, pp. 34–54.

[12] T. Veugen, “Encrypted integer division,” 2010 IEEE International Workshop on Information Forensics and Security, Seattle, WA, 2010, pp. 1-6, doi: 10.1109/WIFS.2010.5711448.

[13] L. B. Kopytkova, N. I. Chervyakov, “Realization of division of numbers in the system of residual classes into modules of the system,” Science. Innovation. Technology, 2003, vol. 34.

[14] N. I. Chervyakov, M. V. Lobes, “Integer division in the system of residual classes,” Proceedings of the III MNTK “Infocommunication technologies in science, production and education.” 2008, vol. 3, p. 198.

[15] N. I. Chervyakov, I. N. Lavrinenko, S. V. Lavrinenko, O. S. Mezentseva, “Methods and algorithms for rounding, scaling and dividing numbers in modular arithmetic,” Collection of scientific paper, Moscow, Zelinograd: Angstrem publishing house, 2006, p. 291-311.

[16] N. I. Chervyakov, “Methods for scaling modular numbers used in digital signal processing,” Infocommunication technologies, 2006, vol. 4, pp. 15-23.

[17] H. L. Garner, “The residue number system,” Western joint computer conference, 1959, pp. 146-153.

[18] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yong Soo Song, “Homomorphic encryption for arithmetic of approximate numbers,” In Tsuyoshi Takagi and Thomas Peyrin, editors, Advances in Cryptology – ASIACRYPT 2017, Part I, vol. 10624 of Lecture Notes in Computer Science, pages 409–437, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany.

[19] J. H. Cheon, D. Kim, and D. Kim, “Efficient Homomorphic Comparison Methods with Optimal Complexity.” IACR Cryptol. ePrint Arch., vol. 2019, p. 1234, 2019.