Hindawi
Security and Communication Networks
Volume 2018, Article ID 9325082, 8 pages
https://doi.org/10.1155/2018/9325082

Research Article

Generalized Bootstrapping Technique Based on Block Equality Test Algorithm

Xiufeng Zhao and Ailan Wang

Department of Information Research and Security, Zhengzhou Information Science Technology Institute, Zhengzhou, 450001, China

Correspondence should be addressed to Xiufeng Zhao; zhao_xiu_feng@163.com

Received 29 September 2018; Revised 19 November 2018; Accepted 9 December 2018; Published 24 December 2018

Guest Editor: Pelin Angin

With the rapid development of cloud computation and big data, the data storage and outsource computation are delegated to the untrusted cloud, which has led to a series of challenging security and privacy threats. Fully homomorphic encryption can be used to protect the privacy of cloud data and solve the trust problem of third party. The key problem of achieving fully homomorphic encryption is how to reduce the increasing noise during the ciphertext evaluation. Bootstrapping procedure can refresh ciphertext with large error, such that the resulting ciphertext has potentially smaller error and allows being continuous homomorphic evaluation. In this paper, we investigated the bootstrapping procedure used to construct fully homomorphic encryption scheme. We proposed a new concept of block homomorphic equality test algorithm and gave an instance based on the FH-SIMD scheme. Furthermore, based on the block homomorphic equality test algorithm, we proposed a faster bootstrapping procedure with smaller bootstrapping keys. Both theory analysis and experiment simulation validate high performance of our bootstrapping algorithm.

1. Introduction

Rapidly developing cloud storage and computation platform allow user delegate data outsource to the cloud server. Cloud computing has the characteristics of data concentration, resource sharing, highly interconnecting, fully opening, etc. It breaks the information island of traditional IT field; meanwhile, it brings even more serious security problems. To protect the privacy of data and the confidential of business secret, it is necessary to encrypting the upload data. However, it is difficult to process ciphertext for traditional encryption algorithm, and this promoted the improvement and development of fully homomorphic encryption (FHE). The prominent advantage of the fully homomorphic encryption is that it can solve ciphertext evaluation problem.

In 2009, Gentry [1, 2] constructed the first fully homomorphic encryption scheme using ideal lattice, which supports arbitrary depth circuit evaluation. Since then many fully homomorphic encryption schemes have appeared involving new mathematical concepts and NP hard problems and improving efficiency, such as FHE from LWE [3], Ring LWE [4], Integer [5], and LWR [6].

In PKC 2010, Smart and Vercauteren [7] proposed a variant of Gentry's scheme with relatively small key and ciphertext sizes. Packing messages allows us to apply single-instruction-multiple data (SIMD) homomorphic operations to many encrypted messages. Smart and Vercautren [8] showed that applying the Chinese reminder theorem (CRT) to number fields partitions the message space of Gentry's FHE scheme into a vector of plaintext slots, resulting in a substantial speed-up, the scheme denoted as FH-SIMD. In the work, they explained that the SIMD operations could be utilized to perform many higher level operations, such as performing AES encryption homomorphically and searching an encrypted database on a remote untrusted server.

Gentry, Sahai and Waters [9] constructed a simple homomorphic encryption scheme from learning with errors in Crypto 2013, called GSW scheme. In this work, they proposed a new technique for building FHE scheme via the approximate eigenvector method. The homomorphic addition and multiplication In GSW scheme are just matrix addition and multiplication, which makes GSW scheme both asymptotically faster and easier to understand. Otherwise, GSW scheme operates single bit once encryption and it is

required to take heavy cost for evaluating a large number of ciphertexts.

Bootstrapping technique is a central technique on fully homomorphic encryption (FHE), which converts "somewhat homomorphic" encryption (SHE) scheme into a fully homomorphic one. That is, bootstrapping procedure homomorphically evaluating the SHE scheme's decryption function on a ciphertext that cannot support any further homomorphic operations, and produces a new one that encrypts the same message and can handle more homomorphic operations.

Bootstrapping procedure is computationally very expensive, and it is becomes the main bottleneck of fully homomorphic encryption practicability. Therefore, there are lots of works try to improve its efficiency. Gentry, Halevi, and Smart [9] proposed a simpler approach that bypasses the homomorphic modular-reduction bottleneck by working with a modulus very close to a power of two. In Crypto 2013, Alperin-Sheriff and Peikert [10] gave entirely algebraic algorithm for bootstrapping in quasilinear time. They gave a method for homomorphically evaluating a class of structured linear transformation using "ring-switching" procedure, resulting in evaluating the decryption function efficiently.

Recently, Alperin-Sheriff and Peikert [11] proposed generalized bootstrapping technique using GSW scheme. The homomorphic decryption of FHE scheme from LWE concludes inner production and rounding operation, and homomorphic equation text algorithm is the key subprocedure of the rounding operation. Embedding the additive group $\mathbb{Z}_q$ into the symmetric group of $q \times q$ permutation matrices is another technique used in the work [11].

In Eurocrypt 2015, Ducas and Micciancio [12] gave an efficient bootstrapping technique by encoding the cyclic group $\mathbb{Z}_q$ into the group of roots unity: $i \longmapsto X^i$, where i is primitive $q-$ th root of unity. This allows implementing a bootstrapping procedure similar to the work of Alperin-Sheriff and Peikert [11], but where each cyclic group element is encoded by a single ciphertext, rather than a vector of ciphertext, this efficiently reduces the size of bootstrapping key.

In AsiaCrypt2016, Chillotti et al. constructed an efficient bootstrapping fully homomorphic encryption scheme, called TFHE [13]. Its time of running bootstrapping is less than 0.1 second. In AsiaCrypt2017, Chillotti et al. [14] optimized the multiple addends of work [13], and made the bootstrapping time reduced 13 milliseconds. 2018, Zhou et al. [15] optimized the serial addends to parallel addends, and the speed of single bootstrapping gate is faster that of work [14]. TFHE scheme and the optimized version both are single bit bootstrapping procedure [13–15]. Although a lot of effort is being spent on improving bootstrapping, the efficient and effective method has yet to be developed. And how to construct efficient multibit bootstrapping procedure is worth further study.

Our Results. In this paper we investigate the homomorphic equality test algorithm in bootstrapping procedure and proposed the concept of block homomorphic equality test algorithm B_Eq? and give an instance based on the FH-SIMD scheme. Furthermore, we proposed a faster bootstrapping procedure based on the block homomorphic equality test algorithm. Both theory analysis and experiment simulation validate the higher performance of our bootstrapping algorithm than that of Alperin-Sheriff and Peikert's work [11].

Organization. In Section 2, we describe some preliminaries on the field and homomorphism, and the concept of generalized bootstrapping technique. In Section 3, we proposed block homomorphic equality test algorithm B_Eq? and give a faster bootstrapping procedure based on B_Eq? algorithm. In Section 4, we give theory analysis and experiment simulation. We give conclusions in Section 5.

2. Preliminaries

2.1. Field and Homomorphism. Let $F(\mathrm{x}) \in \mathbb{F}_2[x]$ be a monic polynomial of degree N, which decomposed to exactly l distinct irreducible factors as follows:

$$F(\mathrm{x}) = \sum_{i=1}^{l} F_i(x), \tag{1}$$

where every polynomial $F_i(x)$ has degree $D = N/l$.

Letting A denote the algebra $\mathrm{A} \coloneqq \mathbb{F}_2[x]/(F)$, we can get the natural homomorphism via Chinese Remainder Theorem (CRT):

$$\mathrm{A} \cong \frac{\mathbb{F}_2[x]}{(F_1)} \otimes \cdots \otimes \frac{\mathbb{F}_2[x]}{(F_l)} \cong \mathbb{F}_{2^D} \otimes \cdots \otimes \mathbb{F}_{2^D}. \tag{2}$$

For $n \mid D$, the finite field $\mathbb{K}_n \coloneqq \mathbb{F}_{2^n}$ is a subfield of $\mathbb{F}_{2^D}$. Let $\mathbb{F}_2[x]/(K_n(x))$ denote a fixed canonical representation of $\mathbb{K}_n$, where $K_n(x) \in \mathbb{F}_2[x]$ is some irreducible polynomial of degree n. Let ψ be a fixed root of $K_n(x)$ in the algebraic closure of $\mathbb{F}_2$. Since $\mathbb{K}_n$ is contained in each of $\mathbb{L}_i \coloneqq \mathbb{F}_2[x]/(F_i)$, there is a homomorphic embedding as follows:

$$\Psi_{n,i}: \begin{cases} \mathbb{K}_n \longrightarrow \mathbb{L}_i \\ \alpha(\psi) \longmapsto \alpha(\sigma_{n,i}(\theta_i)), \end{cases} \tag{3}$$

where $\sigma_{n,i}(\theta_i)$ is a root of $K_n(x)$ in algebra $\mathbb{L}_i$, that is,

$$K_n(\sigma_{n,i}(x)) \equiv 0 \pmod{F_i(x)}. \tag{4}$$

According to CRT and the above homomorphic embedding, we can obtain a homomorphic embedding of $\mathbb{K}_n^l$ into the algebra A which defined as follows:

$$\Gamma_{n,l}: \begin{cases} \mathbb{K}_n^l \longrightarrow A \\ (\mathscr{k}_1(\psi), \ldots, \mathscr{k}_l(\psi)) \longmapsto \sum_{i=1}^{l} \mathscr{k}_i(\sigma_{n,i}(x)) \cdot H_i(x) \cdot G_i(x). \end{cases} \tag{5}$$

where the polynomials $H_i(x)$ and $G_i(x)$ is obtained by CRT and computed as follows:

$$\begin{aligned} H_i(x) &\longleftarrow \frac{F(x)}{F_i(x)}, \\ G_i(x) &\longleftarrow (H_i(x))^{-1} \pmod{F_i(x)}. \end{aligned} \tag{6}$$

From the above definition of $\Gamma_{n,l}$, we can see that $\Gamma_{n,l}$ maps a vector of l binary polynomials $(\mathscr{k}_1(\psi), \ldots, \mathscr{k}_l(\psi))$ each of degree less than n, into a single polynomial $a(x)$ of degree less than N. The map $\Gamma_{n,l}$ defines an isomorphism between $\mathbb{K}_n^l$ and $\Gamma_{n,l}(\mathbb{K}_n^l)$, so the inverse map $\Gamma_{n,l}^{-1}$ is well defined from $\Gamma_{n,l}(\mathbb{K}_n^l)$ to $\mathbb{K}_n^l$. We can represent $\Gamma_{n,l}^{-1}$ as follows:

$\Gamma_{n,l}^{-1}$:

$$\begin{cases} \Gamma_{n,l}\left(\mathbb{K}_n^l\right) \subseteq A \longrightarrow \mathbb{K}_n^l \\ a(x) \longmapsto \left(a(x) \bmod F_1(x), \ldots, a(x) \bmod F_l(x)\right). \end{cases} \tag{7}$$

There are two methods to compute elements in $\mathbb{K}_n^l$: one method is computes component wise on vectors of l elements in $\mathbb{K}_n$; the other concludes three process, firstly, mapping all the inputs to the algebra A by $\Gamma_{n,l}$; secondly, performing computations in algebra A; finally, mapping the results back to $\mathbb{K}_n^l$ by $\Gamma_{n,l}^{-1}$. Furthermore, the fully homomorphic encryption scheme FH-SIMD performs one evaluation for l elements in $\mathbb{K}_n$ using the algebra A.

2.2. Generalized Bootstrapping Technique. Gentry firstly proposed bootstrapping technique, which may transform a somewhat homomorphic encryption scheme to a fully homomorphic encryption scheme. Subsequently, Jacob Alperin-Sheriff and Chris Peikert [11] proposed generalized bootstrapping technique. The generalized bootstrapping technique involves two encryption schemes, outer encryption scheme and inner encryption scheme. It performs decryption procedure of inner encryption scheme using outer encryption scheme, resulting in reducing error in ciphertext. The generalized bootstrapping technique allows that the outer encryption is different from the inner one, realizing that we can design corresponding outer encryption scheme for the concretely inner encryption scheme, such that it effectively performs the decryption circuit of inner encryption scheme. Therefore, the generalized bootstrapping is more efficient than the ordinary one.

2.3. The Decryption of FHE from LWE. The decryption of all fully homomorphic schemes based on LWE involved computing inner production and rounding, that is, input secret key $s \in \mathbb{Z}_q^d$ and binary ciphertext $c \in \{0,1\}^d$; the decryption algorithm is written as

$$\mathrm{Dec}(s, c) = \lfloor \langle s, c \rangle \rceil_2 \in \{0,1\}, \tag{8}$$

where the modular rounding function $\lfloor \cdot \rceil_2 : \mathbb{Z}_q \longrightarrow \{0,1\}$ indicates whether its arguments is "far from" or "close to" 0 (modulo q), and the modulus q and the dimension d can both be made as small as quasi-linear $\widetilde{O}(\lambda)$ in the security parameter via dimension-modulus reduction [3], while still providing provable 2^λ security under conventional lattice assumption. The inner product $\langle s, c \rangle$ is just summing the elements of vector s selectively, that is,

$$\langle s, c \rangle = \sum_{c_j=1} s_j. \tag{9}$$

Supposing that $\langle s, c \rangle = v$, the algorithm rounding can be interpreted by iteration as

$$\lfloor v \rceil_2 = \sum_{x \in \mathbb{Z}_q \ s.t. \lfloor x \rceil_2 = 1} [x = v], \tag{10}$$

where $[x = v]$ denotes the equality test algorithm, when x is equality to v, $[x = v]$ outputs 1; otherwise, $[x = v]$ outputs 0.

Now, we give the decryption algorithm of FHE based LWE in the ciphertext state. During the bootstrapping procedure, the ciphertext of secret $s \in \mathbb{Z}_q^d$ is written by $\overline{s} = (\overline{s_1}, \ldots, \overline{s_d})$ as bootstrapping public key. The inner product in the ciphertext state is denoted as $\overline{v} = \langle \overline{s}, c \rangle = \sum_{c_j=1} \overline{s}_j$. And the rounding algorithm in the ciphertext state is denoted as

$$\lfloor \overline{v} \rceil_2 = \boxplus_{x \in \mathbb{Z}_q \ s.t. \lfloor x \rceil_2 = 1} \left(\overline{[x = v]}\right), \tag{11}$$

where "$\boxplus$" denotes the homomorphic addition on the ciphertext space and $\overline{[x = v]}$ indicates the homomorphic equality test algorithm; it outputs the ciphertext of 1 if and only if $x = v$; otherwise it outputs the ciphertext of 0. We let $\overline{1}$ denote the ciphertext of 1 and $\overline{0}$ denote the ciphertext of 0.

2.4. Generalized Bootstrapping Procedure of FHE from LWE. Assume that the binary ciphertext to be bootstrapped is $c \in \{0,1\}^d$, the secret key is $s \in \mathbb{Z}_q^d$, and the dimension d and the module q are enough small $(q, d = \widetilde{Q}(\lambda))$. The decryption function of FHE scheme from LWE is $Dec_s(c) = \lfloor \langle s, c \rangle \rceil_2 \in \{0,1\}$. We also supposed that the outer encryption scheme is FH-SIMD, that is, FHE scheme which supports SIMD operation. The generalized bootstrapping technique concludes two algorithms: **BootGen** algorithm and **Bootstrap** algorithm [11].

(i) **BootGen**($s \in \mathbb{Z}_q^d$, pk): input secret key vector $s \in \mathbb{Z}_q^d$, and the public key of FH-SIMD encryption; output the bootstrapping public key bk, that is, encrypt the secret key vector s via FH-SIMD scheme and resulting the ciphertext as the bootstrapping public key bk.

(ii) **Bootstrap**($bk, c = \{0,1\}^d$): input the bootstrapping pubic key bk and the ciphertext vector $c = \{0,1\}^d$, output a new ciphertext c' of original encryption scheme based LWE, and the result of decrypting c' using secret key $s \in \mathbb{Z}_q^d$ is same as the one decrypting c using secret key $s \in \mathbb{Z}_q^d$, but c' with less error.

3. Faster Bootstrapping Based on FH-SIMD

3.1. Main Ideas. Jacob Alperin-Sheriff and Chris Peikert proposed the generalized bootstrapping method based on the GSW scheme. Homomorphic equality test is a key component of the generalized bootstrapping algorithm, that is, for the fixed $v \in \mathbb{Z}_q$, under the ciphertext state, travels every $x \in \mathbb{Z}_q$ which satisfies $\lfloor x \rceil_2 = 1$, and decide that whether $v = x$ or not, see Figure 1.

We intend to proposed block homomorphic equality test algorithm, that is, it travels a block $(x_1, x_2, \ldots, x_l)$, $x_i \in \mathbb{Z}_q$

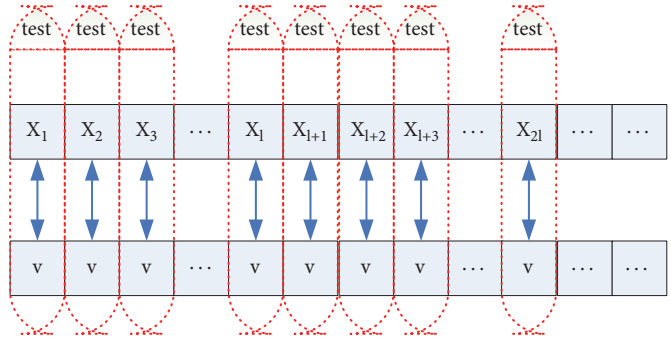

Figure 1: Homomorphic equality test.

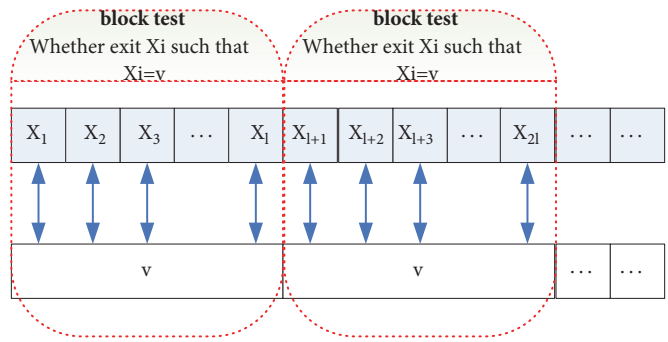

Figure 2: Block homomorphic equality test.

which satisfies $\lfloor x \rceil_2 = 1$, and decide that whether this batch of $(x_1, x_2, \ldots, x_l)$ exits a x_i such that $v = x_i$ holds, see Figure 2.

Resort to the FH-SIMD homomorphic encryption scheme [7], we will give a block homomorphic equality test algorithm B_Eq?, and then propose an efficient generalized bootstrapping algorithm. The bootstrapping key is an encryption of each coordinate of the secret key $s \in \mathbb{Z}_q^d$, and consists of d FH-SIMD ciphertexts. To bootstrapping $c \in \{0,1\}^d$, the inner product $\langle s, c \rangle \in \mathbb{Z}_q$ is computed homomorphically as a subset-sum using addition method, and the rounding function is computed using block homomorphic equality test algorithm and addition method.

3.2. Block Homomorphic Equality Test Algorithm. In this section, we describe our block homomorphic equality test algorithm, called B_Eq?.

Input: ciphertex $\overline{v} \in \mathbb{Z}_q$ and plaintext block $(x_1, \ldots, x_l) \in \mathbb{Z}_q^l$.

Output: if there exits a $x_i \in \{x_1, x_2, \ldots, x_l\}$ such that $v = x_i$ holds, then the block homomophic equality test algorithm outputs $\overline{1}$; otherwise it outputs $\overline{0}$.

We assume that $\overline{v}$ and all x_i are n bits in length, and thus can be encoded as an element of the finite filed $\mathbb{K}_n = \mathbb{F}_{2^n}$, where $n = \lceil \log q \rceil$. The concrete procedure is described as follows:

(1) For $(x_1, \ldots, x_l) \in \mathbb{Z}_q^l$, where every x_i satisfies $\lfloor x_i \rceil_2 = 1$, embed each coordinate $x_i \in \mathbb{Z}_q$ as an element of $\mathbb{F}_{2^n}$. Our aim is to pack $(x_1, \ldots, x_l)$ into single element $X \in \mathbb{K}_n$, so we compute $X = \Gamma_{n,l}(x_1, \ldots, x_l)$. Then compute the trivial encryption of X in the algebra $\mathbf{A}$ using FH-SIMD scheme. It is worth noting that we encrypt X without random, such that saving computational cost. That is,

$$\overline{X} = \text{FH} - \text{SIMD.Encrypt}\left(\Gamma_{n,l}\left(x_1, \ldots, x_l\right), pk\right). \tag{12}$$

Then compute

$$\overline{V} = \Gamma_{n,l}\left(\overline{v}, \ldots, \overline{v}\right). \tag{13}$$

(2) Sum the ciphertext $\overline{V}$ and $\overline{X}$, and denote the sum as $c^{(1)}$, that is,

$$c^{(1)} = \overline{V} \boxplus \overline{X}. \tag{14}$$

(3) Homomorphically raised $c^{(1)}$ to the power $(2^n - 1)$, that is,

$$c^{(2)} = \left(c^{(1)}\right)^{2^n - 1}. \tag{15}$$

And compute

$$\Gamma_{n,l}^{-1}\left(c^{(2)}\right) = \left(\left(\overline{v} \boxplus \overline{x_1}\right)^{2^n-1}, \ldots, \left(\overline{v} \boxplus \overline{x_l}\right)^{2^n-1}\right). \tag{16}$$

According to the homomorphism of encryption scheme FH-SIMD, the $(2^n - 1)$ power of ciphertext is corresponding to the $(2^n - 1)$ power of plaintext. Of course, the ciphertext is homomorphically raised to the power $(2^n - 1)$, via performing $2n$ applications of multiplication. Since the plaintext corresponding to $(\overline{v} \boxplus \overline{x_i})$ is an element of finite field $\mathbb{F}_{2^n}$ and

its max multiplicative order is $(2^n - 1)$, then the plaintext corresponding to $(\overline{v} \boxplus \overline{x}_i)^{2^n-1}$ is either 0 or 1. Therefore, $(\overline{v} \boxplus \overline{x_i})^{2^n-1}$ is a ciphertext of encrypting 1 (nonzero element) or a ciphertext of encrypting 0 (zero element). When $(\overline{v} \boxplus \overline{x}_i)$ is an encryption of 0, this means that $v = x_i$ holds; when $(\overline{v} \boxplus \overline{x}_i)$ is an encryption of 1, this means that $v = x_i$ does nit hold.

(4) Compute

$$c^{(3)} = c^{(2)} \boxplus \mathrm{FH} - \mathrm{SIMD.Encrypt}\left(\Gamma_{n,l}(1, \ldots, 1), pk\right), \quad (17)$$
$$\Gamma_{n,l}^{-1}\left(c^{(3)}\right) = \left(\left(\overline{v} \boxplus \overline{x_1}\right)^{2^n-1} \boxplus \overline{1}, \ldots, \left(\overline{v} \boxplus \overline{x_l}\right)^{2^n-1} \boxplus \overline{1}\right).$$

We can see that as long as the equation $x_i = v$ holds, the i-th component of $c^{(3)}$ is $\overline{0} \boxplus \overline{1} = \overline{1}$; otherwise, the i-th component of $c^{(3)}$ is $\overline{1} \boxplus \overline{1} = \overline{0}$. Therefore, if there exits a $x_i \in \{x_1, x_2, \ldots, x_l\}$ such that $v = x_i$ holds, then $(\overline{v} \boxplus \overline{x}_i)^{2^n-1} \boxplus \overline{1} = \overline{1}$, and all other $(\overline{v} \boxplus \overline{x}_j)^{2^n-1} \boxplus \overline{1} = \overline{0}$, for all $j \neq i$. It follows that $c^{(3)} = \overline{1}$. That is, if there exits a $x_i \in \{x_1, x_2, \ldots, x_l\}$ such that $v = x_i$ holds, the block homomophic equality test algorithm outputs $\overline{1}$; otherwise it outputs $\overline{0}$.

From the above steps, we finish the block homomorphic equality test; then we can homomorphic compute $\lfloor v \rceil_2$, that is, $\lfloor \overline{v} \rceil_2 \coloneqq c^{(3)}$.

3.3. Faster Bootstrapping Technique. In this section, we construct faster bootstrapping procedure from the block homomorphic equality test algorithm B_Eq?. The bootstrapping procedure consists of two algorithms: BootKeyGen and Bootstrap. The procedure is used to refresh ciphertexts of all known standard LWE-based FHE. We get the input ciphertext $c \in \{0,1\}^d$ for Bootstrap, and it is from the dimension-modulus reduction and bit-decomposition of the ciphertext to be bootstrapped. Let $\mathrm{s} \in \mathbb{Z}_q^d$ be the secret key that corresponding to the ciphertext c.

BootGen($\mathrm{s} \in \mathbb{Z}_{\mathrm{q}}^{\mathrm{d}}$, pk): input the secret key $\mathrm{s} \in \mathbb{Z}_{\mathrm{q}}^{\mathrm{d}}$ for the ciphertext to be refreshed and the public key pk of FH-SIMD scheme. Without loss of generality, for every $j \in [d]$, encode each coordinate $\mathrm{s}_{\mathrm{j}} \in \mathbb{Z}_{\mathrm{q}}$ to the element of finite field $\mathbb{F}_{2^{\mathrm{n}}}$. Then encrypt its ciphertext under FH-SIMD scheme, and generate the bootstrapping key:

$$\overline{\mathrm{s}}_j = \mathrm{FH} - \mathrm{SIMD.Encrypt}\left(\mathrm{s}_{\mathrm{j}}, pk\right). \quad (18)$$

Output the bootstrapping public key $bk = \{\overline{s_1}, \ldots, \overline{s_d}\}$. The bootstrapping key consists of d FH-SIMD ciphertexts.

Bootstrap($bk, c \in \{0,1\}^d$): input the binary ciphertext $c \in \{0,1\}^d$, and perform the following two phases:

(i) **Inner Product** Homomorphically compute inner product $\overline{v}$ using the bootstrapping public key bk. It is known that

$$v \triangleq \langle s, c \rangle = \sum_{j:c_j=1} s_j \in \mathbb{Z}_q. \quad (19)$$

It follows that

$$\overline{v} \triangleq \langle \overline{s}, c \rangle = \boxplus_{j:c_j=1} \overline{\mathrm{s}}_j. \quad (20)$$

(ii) **Round** For every $x_i \in \mathbb{Z}_q$ which satisfies $\lfloor x_i \rceil_2 = 1$, arrange in order of size, and divide them into blocks of l items, and let us suppose they are distinct from one another, and there are altogether k blocks:

$$\left(x_1, \ldots, x_l\right), \left(x_{l+1}, \ldots, x_{2l}\right), \ldots, \left(x_{(k-1)l+1}, \ldots, x_{kl}\right). \quad (21)$$

For every block $(x_{jl+1}, \ldots, x_{(j+1)l})$, $j = 0, 1, \ldots, k-1$, run block homomorphic equality test algorithm in parallel,

$$c_j = \mathrm{B\_Eq?}\left(\overline{v}, \left(x_{jl+1}, \ldots, x_{(j+1)l}\right)\right). \quad (22)$$

Then compute

$$C = \boxplus_{j:1 \leq j \leq k} c_j. \quad (23)$$

We can see that c_j is either $\overline{1}$ or $\overline{0}$, and then C is the encryption of $\lfloor v \rceil_2$, C is also either $\overline{1}$ or $\overline{0}$, and it refreshed ciphertext with smaller error. Note that the output C is a FH-SIMD ciphertext encrypted under pk. If desired, we can convert this ciphertext back to one for the original LWE FHE cryptosystem. We can also perform key-switch from sk back to the original secret keys.

4. Analysis

4.1. Correctness Analysis

Lemma 1 (correctness). *For $bk \longleftarrow$ BootGen(s, pk), the FH-SIMD ciphertext $C \longleftarrow Bootstrap(bk, c)$ is designed to encrypt $Dec_s(c) = \lfloor \langle s, c \rangle \rceil_2 \in \{0,1\}$.*

Proof. Firstly, the FH-SIMD ciphertext $\overline{s}_j$ is designed to encrypt s_j from (18). Therefore, since $\varphi : \mathbb{Z}_q \longrightarrow \mathbb{F}_{2^n}$ is homomorphic embedding, the ciphertext $\overline{v}$ as defined as in (20) designed to encrypt

$$\sum_{j:c_j=1} s_j = \langle s, c \rangle = v. \quad (24)$$

By correctness of block homomorphic equality test algorithm B_Eq?, the homomorphic sum $C = \boxplus_{j:1 \leq j \leq k} c_j$ is designed to encrypt 1 if and only if $v = x$. Finally, since the homomorphic sum is taken over every $x \in \mathbb{Z}_q$ such that $\lfloor x \rceil_2 = 1$, it is designed to encrypt 1 if and only if $\lfloor v \rceil_2 = 1$. □

4.2. Security Analysis

Lemma 2 (semantic security). *Suppose that the FH-SIMD scheme secret key sk is generated independently of the secret key $\mathrm{s} = (s_1, \ldots, s_d) \in \mathbb{Z}_q^d$ of FHE scheme from LWE; then Ind-CPA security of the bootstrapping key follows immediately from the Ind-CPA security of FH-SIMD, hence from SIVP of ideal lattice.*

Table 1: Compare of the performance of homomorphic equality test algorithm.

algorithm	Enc	C_A	C_M	$\Gamma_{n,l}$	$\Gamma_{n,l}^{-1}$
Eq? [11]	0	0	$O(q)$	0	0
B_Eq?	2	2	$O(\log q)$	2	1

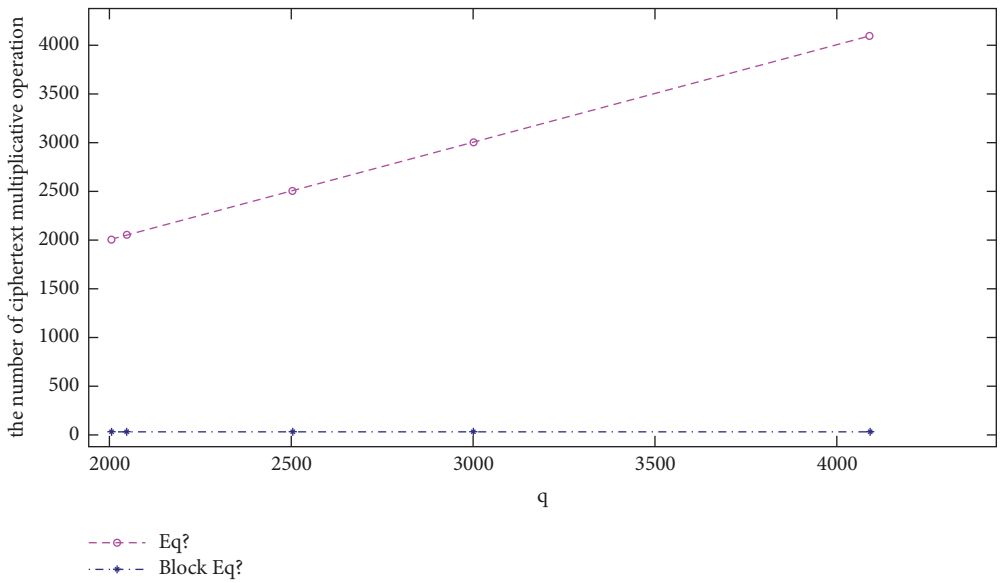

Figure 3: The relation between the ciphertext multiplicative quantity and the modulus q.

Proof. For $bk \longleftarrow$ BootGen(s, pk), if there is not an adversary can distinguish the bootstrapping key bk from a random element in the same space, then the bootstrapping procedure called satisfing semantic security.

In our bootstrapping procedure, for $s = (s_1, \ldots, s_d) \in \mathbb{Z}_q^d$, the bootstrapping key consists of d FH-SIMD ciphertexts; that is, $bk = \{\overline{s_1}, \ldots, \overline{s_d}\}$ is generated via FH-SIMD scheme, where

$$\overline{s}_j = \text{FH} - \text{SIMD.Encrypt}\left(s_j, pk\right). \tag{25}$$

Suppose that there is an adversary $\mathscr{A}$ can distinguish the bootstrapping bk from random element. Then we can construct an algorithm $\mathscr{B}$ by calling the adversary $\mathscr{A}$ and break the Ind-CPA security of FH-SIMD scheme and, furthermore, solve the SIVP of ideal lattice. □

4.3. Performance Analysis. Our block homomorphic equality test algorithm B_Eq? has a cost of $(2\cdot\text{Enc}+2\cdot C_A+2\log q\cdot C_M+2\cdot\Gamma_{n,l}+\Gamma_{n,l}^{-1})$ per data block, where C_A denotes add operation of the ciphertext and C_M denotes multiplicative operation of the ciphertext, whereas the homomorphic equality test algorithm Eq? involves $q \cdot C_A$, which is exponential times of B_Eq? algorithm, meaning that the computation of Eq? algorithm is more costly, reference to Table 1.

In the work of Alperin-Sheriff and Peikert [11], one inner product evaluation of bootstrapping needs to compute d ciphertexts compose evaluation, and one rounding evaluation of bootstrapping needs to call $O(q)$ Eq? algorithm, and $O(q)$ ciphertext multiplicative operation. Whereas one inner production of our faster bootstrapping needs to compute d ciphertext additions, and one rounding evaluation needs to call k B_Eq? algorithm, where l is the size of block, and k is the number of block, $k = \lceil q/l \rceil$.

Suppose that LWE problem has 80 bits security when q is set to be 2003. Parameters setting as above, and when q is set to be 2003, 2047, 2501, 3001, 4093, and 12899, we give the relation between multiplicative operation quantity and modulus q, as shown in Figure 3. As the modulus q increases, the number of ciphertext multiplicative operation grows swiftly in the AP's bootstrapping procedure, whereas the number of ciphertext multiplicative operation grows slowly.

On the other hand, for the fixed modulus q and m, we give the relation between multiplicative operation quantity once running our faster bootstrapping procedure based on the block size of block homomorphic equality test algorithm B_Eq?. When $m = 11441$, $N = \varphi(m) = 10752$, and $2^{48} \bmod 11441 = 1$, so $d = 48$, we set $n = 12$, which satisfies $n \mid d$. Then we set the size of block as $16, 32, 64, 128, 224$, that is, the size of block $l = 16, 32, 64, 128, 224$. Then the number of blocks $k = 3024, 1512, 768, 384, 216$. We can see from Figure 4, as the block size increases, the ciphertext multiplicative operation drops dramatically.

5. Conclusions

Fully homomorphic encryption scheme allows evaluating encrypted data, without decrypting the corresponding ciphertext. In fully homomorphic encryption scheme, the

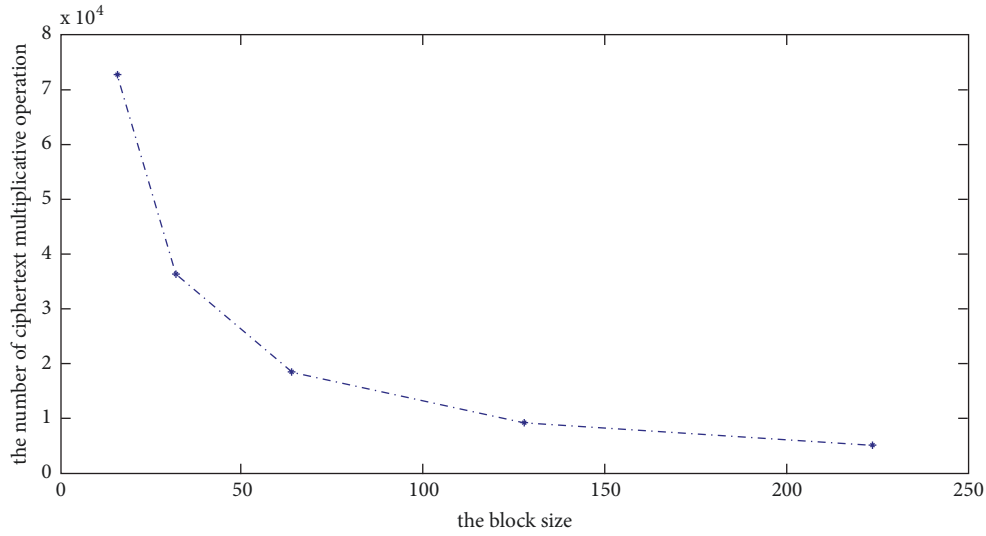

Figure 4: The relation between the ciphertext multiplicative quantity and the block size.

ciphertext has a noise that grows at each homomorphic evaluation. When the noise reaches a threshold, then the ciphertext cannot be decrypted correctly. The number of homomorphic operations can be made asymptotically large using bootstrapping technique.

In this paper, we further investigated the bootstrapping procedure. We proposed the concept of block homomorphic equality test algorithm and give an instance based on the FH-SIMD scheme. Furthermore, we give a faster bootstrapping procedure based on the block homomorphic equality test algorithm. Both theory analysis and experiment simulation validate the higher performance of our bootstrapping than that of the work [11].

Data Availability

Our underlying data related to the article is the paper as cited as in [11].

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Nature Science Foundation of China under Grant no. 61601515 and Nature Science Foundation of Henan Province under Grant no. 162300410332.

References

[1] C. Gentry, "Fully homomorphic encryption using ideal lattices," In Proc of the 41th Annual ACM Symp on Theory of Computing (STOC)," pp. 169–178, ACM, New York, NY, USA, 2009.

[2] C. Gentry, *A fully homomophic encryption scheme [Ph.D. thesis]*, Stanford University, 2009, http://crypto.stanford.edu/craig.

[3] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS '11)*, pp. 97–106, Palm Springs, Calif, USA, October 2011.

[4] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Advances in Cryptology—CRYPTO 2011*, R. Phillip, Ed., vol. 6841, pp. 505–524, Springer, Berlin, Germany, 2011.

[5] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Advances in cryptology (EUROCRYPT)*, vol. 6110, pp. 24–43, Springer, Berlin, Germany, 2010.

[6] Fucai Luo, Fuqun Wang, Kunpeng Wang, Jie Li, and Kefei Chen, "LWR-Based Fully Homomorphic Encryption, Revisited," *Security and Communication Networks*, vol. 2018, Article ID 5967635, 12 pages, 2018.

[7] N. P. Smart and F. Vercauteren, "Fully homomorphic SIMD operations," *Designs, Codes and Cryptography*, vol. 71, no. 1, pp. 57–81, 2014.

[8] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based," in *CRYPTO, LNCS*, R. Canetti and J. A. Garay, Eds., vol. 8042, pp. 75–92, Springer, Heidelberg, Germany, 2013.

[9] C. Gentry, S. Halevi, and N. P. Smart, "Better bootstrapping in fully homomorphic encryption," in *Public key cryptography (PKC)*, vol. 7293 of *Lecture Notes in Comput. Sci.*, pp. 1–16, Springer, Heidelberg, 2012.

[10] J. Alperin-Sheriff and C. Peikert, "Practical bootstrapping in quasilinear time," in *Advances in Cryptology – CRYPTO*, vol. 8042, pp. 1–20, 2013.

[11] J. Alperin-Sheriff and C. Peikert, "Faster bootstrapping with polynomial error," in *Proceedings of the International Cryptology Conference*, J. A. Garay and R. Gennaro, Eds., pp. 297–314, Springer, Berlin, Germany, 2014.

[12] L. Ducas and D. Micciancio, "FHEW: Bootstrapping homomorphic encryption in less than a second," in *EUROCRYPT, Part I, LNCS*, E. Oswald and M. Fischlin, Eds., vol. 9056, pp. 617–640, Springer, Heidelberg, Germany, 2015.

[13] I. Chillotti, N. Gama, M. Georgieva et al., "Faster fully homomorphic encryption: bootstrapping in less than 0.1 seconds," in *Advances in Cryptology - ASIACRYPT*, pp. 3–33, Springer, Heidelberg, Germany, 2016.

[14] I. Chillotti, N. Gama, M. Georgieva et al., "Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE," in *ASIACRYPT 2017. LNCS*, T. Takagi and T. Peyrin, Eds., vol. 10624, pp. 377–408, Springer, Cham, UK, 2017.

[15] T. Zhou, X. Yang, L. Liu, W. Zhang, and N. Li, "Faster Bootstrapping With Multiple Addends," *IEEE Access*, vol. 6, pp. 49868–49876, 2018.