Methods for Efficient Homomorphic Integer Polynomial Evaluation based on GSW FHE

Husen Wang, Qiang Tang

University of Luxembourg
husen.wang@uni.lu, qiang.tang@uni.lu

May 20, 2016

Abstract We introduce new methods to evaluate integer polynomials with GSW FHE. Our methods cause much slower noise growth and result in much better efficiency in the evaluation of low-degree large plaintext space polynomials. One method is about a new encryption procedure and its application homomorphic integer multiplication. The other is about an extended version to SIMD by packing more integers diagonally into a matrix. We show the possibility of combining ciphertext compression with these techniques, in order to achieve better efficiency for ciphertext transmission and homomorphic polynomial evaluation.

Keywords: GSW, integer multiplication, packing, ciphertext compression

1 Introduction

Fully Homomorphic encryption (FHE) has received considerable attention ever since the breakthrough by Gentry in [8], which put forward a paradigm for converting a "somewhat homomorphic" encryption scheme with a limited evaluation depth to "fully homomorphic" encryption scheme with unlimited depth, by bootstrapping a noisy ciphertext to less noisy one. Since then, a lot of efforts have been dedicated to make the scheme more practical by improving the evaluation performance and reducing the number of bootstrapping times. The second generation of schemes such as BGV [3], LTV [14](NTRU based), the scale-invariant version such as FV [7], YASHE [2] utilize techniques such as bit decomposition, modulus switching, key switching to reduce the noise growth when performing evaluations, especially the multiplications. However, the unnatural key switching leads to serious computation overhead. To resolve this problem, the third generation FHE was proposed by Gentry in [10], which uses a matrix as the ciphertext. Later, [4] achieved quasi-additive noise growth property by utilizing the asymmetrical property of the matrix multiplication. [1] improved the bootstrapping procedure by arithmetically evaluating the decryption circuit and embedding elements in $\mathbb{Z}_q$ into smaller symmetric groups using Chinese Reminder Theorem.

Recently, [11] enabled packing multiple bits by adopting a new structure for the ciphertexts, and further improved the bootstrapping procedure in [1]. However, all the above improvements target at binary plaintexts, since the multiplication noise growth is related to the plaintext size and the noise in a multiplication chain is exponential with the size.

In real world application, there are two scenarios for the circuits that are to be evaluated by FHE. One is large plaintext space, low-degree polynomials [6] [23], the other scenario is polynomials with binary plaintext space but large circuit depth [9] [13]. For circuits with both large plaintext space and large depth, the parameters will be too large to be practical [17]. The existing GSW FHE and its variants can handle the latter, and can achieve good performances as shown in [12]. But it's still an open problem to efficiently deal with circuits in the first category.

In this paper, we solve this problem with a new evaluation method, by utilizing the asymmetrical matrix multiplication property and some circuit sequentialization technique. We also extend the method to SIMD.

1.1 Our contribution

Firstly, to reduce the noise growth in integer multiplication, we decompose one integer to binary vectors with known weights, then perform the multiplication through several multiplications of one integer with a binary number and then a weight, finally a sum. The first multiplication only increase additive noise, since one of them is kept always as a fresh ciphertext with binary plaintext. The second multiplication can be done without increasing the noise by having some constraints on the modulus and the encoding of weights, because the weights are all the power of two and encoded with zero noise inside. Finally we achieve the noise growth of ℓ times (where ℓ is the bit width of the plaintext space), in contrast to 2^ℓ if directly using the methods in [4,10]. We do admit that out method has the disadvantage of ciphertext expansion, ℓ times more memory compared to a direct application of the GSW FHE scheme. The homomorphic multiplication computation cost is also about ℓ times (the multiplication with the encoded weights can be ignored since they are all sparse matrixes). Using this method, we can evaluate an integer polynomial sequentially with the noise growth equivalent to a multiplication chain, so that we can keep one ciphertext in every multiplication step be a fresh one. This leads to a circuit depth of d (which is the degree of the polynomial), in contrast to $\log d$ in [10]. Since we target at large plaintext space, low degree scenario, where d is small, this sacrifice is acceptable.

Secondly, based on a simple observation that that if the integers are assigned diagonally in a matrix then the multiplication of two matrixes will be equivalent to the multiplication of that of the corresponding diagonal integers, we pack multiple integers as a diagonal matrix. We don't encrypt the integers directly, but encrypt their binary decomposed vectors, which will be beneficial to the evaluation.

Thirdly, we consider the possibility of combining the concepts of ciphertext compression with integer polynomial evaluation. In our scenario, the integers to be used in the evaluation can be encrypted with stream ciphers and an untrusted server can homomorphically decrypt and evaluate. As long as that we can constrain the plaintext space to $\mathbb{Z}_2$ within the homomorphic decryption, the noise in the decrypted ciphertext will be small. In this way, we can seamlessly combine ciphertext compression with the evaluation, which is more efficient than existing designs that can only evaluate boolean circuit after homomorphic decryption.

1.2 Organization

The rest of the paper is organized as follows. In Section 2 we recall some mathematical preliminaries such as subgaussian variables and DLWE. In Section 3, we introduce the GSW FHE variant from [1] and our modification. In Section 4 we discuss the homomorphic operations for integers and polynomials, then analyze their noise growth property. In Section 5, we extend the method to packing GSW FHE and introduce the operations on packed integers. In Section 6, we show combination of our evaluation method with ciphertext compression. Finally, we make the conclusion.

2 Preliminaries

We denote the set of natural numbers by $\mathbb{N}$, the set of integers by $\mathbb{Z}$, the set of real numbers by $\mathbb{R}$. Let $\mathbb{G}$ be a set, χ be some probability distribution, then we use $a \xleftarrow{\mathbf{U}} \mathbb{G}$ to denote that a is chosen from $\mathbb{G}$ uniformly at random, and use $b \xleftarrow{\mathbf{R}} \chi$ to denote that b is chosen along χ. We take all logarithms log to base 2, unless otherwise noted.

We assume that column vectors are represented with bold lower case letters, e.g., $\mathbf{x}$, and the transpose as $\mathbf{x}^t$. We also use $()_{i\in[0,\ell)}$ to represent a vector with the length ℓ. Let $\otimes$ be the tensor product, $\cdot$ be the multiplications between matrices or scalars. The inner product between two vectors is denoted by $< \mathbf{x}, \mathbf{y} >$. We denote $||x||_2$ as the Euclidean norm, $||x||_\infty$ as the Maximum norm.

Matrices are represented with bold capital letters, e.g., $\mathbf{X}$, and the i-th column vector of a matrix is denoted by $\mathbf{x}_i$. For matrices $\mathbf{A} \in \mathbb{Z}^{m\times n_1}$ and $B \in \mathbb{Z}^{m\times n_2}$, $[\mathbf{A}||\mathbf{B}] \in \mathbb{Z}^{m\times(n1+n2)}$ denotes the column concatenation of $\mathbf{A}$ and $\mathbf{B}$. We denote the $n \times n$ identity matrix with $\mathbf{I}_n$. $\ell = \lceil \log q \rceil$. Let $\mathbf{g} = (2^0, 2^1, 2^2, \dots 2^{\ell-1}) \in \mathbb{Z}_q^\ell$. For an integer $x \in \mathbb{Z}_q$, we use $x[i]$ to denote the i-bit of x and $(x[i])_{i\in[0,\ell)} \in \mathbb{Z}_2^\ell$ to denote the binary representation of x.

2.1 Subgaussian

Alperin-Sheriff and Peikert [1] used a randomized function $\mathbf{G}^{-1} : \mathbb{Z}_q^{n\times m} \to \mathbb{Z}_q^{n\ell\times m}$ instead of the decomposition procedure to randomize the ciphertexts and more accurately analyze the resulted noise growth property. Here, we take the necessary Claims from related papers.

Definition 1 ([22]). *A real random variable X (or its distribution) is subgaussian with parameter $r > 0$, if for all $t \in \mathbb{R}$, its (scaled) moment-generating function satisfies $\mathbb{E}[exp(2\pi tX)] \leq exp(\pi r^2 t^2)$.*

The subgaussain random variables has two properties:

- Homogeneity: If the subgaussian variable X has parameter s, cX is subgaussian with parameter cs.
- Pythagorean additivity: If two independent subgaussian random variables X_1 and X_2 with parameters s_1 and s_2 respectively, $X_1 + X_2$ is subgaussian with parameter $\sqrt{s_1^2 + s_2^2}$.

Claim 1 *For $a \in \mathbb{Z}_q$, there is a randomized, efficiently computable function $g^{-1} : \mathbb{Z}_q \to \mathbb{Z}^\ell$ such that $\mathbf{x} \leftarrow \mathbf{g}^{-1}(a)$ is subgaussian with parameter $O(1)$ and always satisfies $< \mathbf{g}, \mathbf{x} >= a$.*

Claim 2 *For $\mathbf{X} \in \mathbb{Z}_q^{n \times m}$, there is a randomized, efficiently computable function $\mathbf{G}^{-1} : \mathbb{Z}_q^{n \times m} \to \mathbb{Z}^{n\ell \times m}$ such that $\mathbf{X} \leftarrow \mathbf{G}^{-1}(\mathbf{A})$ is subgaussian with parameter $O(1)$ and always satisfies $\mathbf{G} \cdot \mathbf{X} = \mathbf{A}$.*

2.2 DLWE

The LWE problem by Regev [20] and its decisional version $\mathbf{DLWE}_{n,m,q,\chi}$ are recapped in Definition 2. The reductions from $\mathbf{DLWE}_{n,m,q,\chi}$ to $\mathbf{GapSVP}_{\tilde{O}(n/\alpha)}$ based on quantum algorithm [20] and classical algorithm [19] are illustrated in **Corollary 1**, as we rewrite the **Corollary** in [4]. The $\mathbf{GapSVP}_\gamma$ is assumed to be hard, since the best algorithm requires at least $2^{\tilde{\Omega}(n/\log\gamma)}$ time [21].

Definition 2 (DLWE). *For $q = q(n) \in \mathbb{N}$ and an error distribution $\chi = \chi(n)$ over $\mathbb{Z}$,the (average-case) decision learning with errors problem, denoted $\mathbf{DLWE}_{n,m,q,\chi}$, is to distinguish (with non-negligible advantage) m samples chosen according to $\mathbf{A}_{s,\chi}$ (for $s \xleftarrow{\mathbf{U}} \mathbb{Z}_q^n$), from m samples chosen according to the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. We denote $\mathbf{DLWE}_{n,m,q,\chi}$ the variant where the adversary gets oracle access to $\mathbf{A}_{s,\chi}$, and is not a-priori bounded in the number of samples.*

Corollary 1 (DLWE to GapSVP). *Let $q = q(n) \in \mathbb{N}$ be either a prime power $q = p^r$ or a product of co-prime numbers $q = \prod q_i$ such that for all i, $q_i = poly(n)$, Let $\alpha \geq \sqrt{n}/q$. If there is an efficient algorithm that solves the (average-case) $\mathbf{DLWE}_{n,m,q,\chi}$ problem, then:*

- *There is an efficient quantum algorithm that solves $\mathbf{GapSVP}_{\tilde{O}(n/\alpha)}$ and $\mathbf{SIVP}_{\tilde{O}(n/\alpha)}$ on any n-dimensional lattice.*
- *If in addition $q \geq \tilde{O}(2^{n/2})$, there is an efficient classical algorithm for $\mathbf{GapSVP}_{\tilde{O}(n/\alpha)}$ on any n-dimensional lattice.*

3 New Evaluation Method for GSW

3.1 GSW Variant and Method

We first describe the GSW FHE variant in [1], which is identical to original GSW FHE scheme [10] except for some syntactical difference.

$\mathsf{Setup}(\lambda, L)$: Choose a modulus $q = q(\lambda, L)$, the lattice dimension $n = n(\lambda, L)$. $\ell = \lceil \log q \rceil$. The distribution χ as subgaussian over $\mathbb{Z}$. Let $\mathbf{G} = \mathbf{g}^t \otimes \mathbf{I}_n$. Output $params = (n, q, \chi, \ell, \mathbf{G})$.

$\mathsf{KeyGen}(params)$: sample $\bar{\mathbf{s}} \xleftarrow{\mathbf{R}} \chi^{n-1}$, output secret key $\mathbf{s} = [\bar{\mathbf{s}}||1] \in \mathbb{Z}^n$.

$\mathsf{Enc}(params, \bar{\mathbf{s}}, \mu \in \mathbb{Z}_q)$: $\bar{\mathbf{C}} \xleftarrow{\mathbf{U}} \mathbb{Z}_q^{(n-1)\times n\ell}$, $\mathbf{e} \xleftarrow{\mathbf{R}} \chi^{n\ell}$. Let $\mathbf{b}^t = [\mathbf{e}^t - \bar{\mathbf{s}}^t\bar{\mathbf{C}}]_q$. Output the ciphertext

$$\mathbf{C} = \begin{pmatrix} \bar{\mathbf{C}} \\ \mathbf{b}^t \end{pmatrix} + \mu\mathbf{G}$$

$\mathsf{Dec}(params, \mathbf{s}, \mathbf{C})$: For $q = 2^{\ell-1}$, select the last $\ell - 1$ columns of $\mathbf{C}$ as $\mathbf{C}_{(\ell-1)}$. Then $\mathbf{s}^t\mathbf{C}_{(\ell-1)} = \mu \cdot \mathbf{g}^t + \mathbf{e}'$. Recover $LSB(\mu)$ from $\mu \cdot 2^{\ell-2} + \mathbf{e}'_{\ell-2}$, then the next-LSB from $(\mu - LSB(\mu)) \cdot 2^{l-3} + \mathbf{e}'_{\ell-3}$, etc. (See [16] for the general q case).

With respect to the decryption algorithm Dec, we introduce an different procedure BEnc which basically decomposes the input into binary vector and encrypts every bit.

$\mathsf{BEnc}(params, \bar{\mathbf{s}}, \mu \in \mathbb{Z}_q)$: For an integer $\mu \in \mathbb{Z}_q$, decompose it into binary representation $(\mu[i])_{i\in[0,l)} \in Z_2^\ell$, then encrypt every bit respectively to $\mathbf{C}[i] = \mathsf{Enc}(params, \bar{\mathbf{s}}, \mu[i] \in \mathbb{Z}_2)$. The final ciphertext is $(\mathbf{C}[i])_{i\in[0,l)}$, together with a weight 2^i for every $\mathbf{C}[i]$.

For a known constant $\alpha \in \mathbb{Z}_q$, we define $\mathbf{M}_\alpha = [\alpha\mathbf{G}]_q$.

3.2 Correctness and Security

Definition 3. *For a ciphertext $\mathbf{C} \in \mathbb{Z}_q^{n\times n\ell}$, it's designed to encrypt $\mu \in \mathbb{Z}$ under the secret key $\mathbf{s}$, if there is an error vector $\mathbf{e}^t \in \mathbb{Z}^{n\ell}$, and*

$$\mathbf{s}^t\mathbf{C} - \mu \cdot \mathbf{s}^t\mathbf{G} = \mathbf{e}^t \bmod q$$

From [16] Sec 4.1, if $||\mathbf{e}^t||_\infty < q/8$, $\mathsf{Dec}(params, \mathbf{s}, \mathbf{C})$ can correctly output μ. It also works the general case q ([16] Sec 4.2). This bound will set a limit to the final ciphertext noise, which can be estimated according to the evaluated function and the noise growth by performing basic homomorphic operations such as addition and multiplication. These basic homomorphic operation noise bound will be given in the section 4.

Since the GSW FHE variant in [1] is IND-CPA secure based on the $\mathbf{DLWE}_{n,m,q,\chi}$ assumption, it is clear that adopting the new encryption procedure BEnc will keep the same security property.

4 Homomorphic Operations

In this section, we introduce our new homomorphic polynomial evaluation method based on the encryption procedure BEnc, which has a noise growth rate $O(\ell^d)$ for a degree d polynomial, compared with the original GSW method [10] with $(N+T')^{\log d}B \approx (N+2^\ell)^{\log d}B$ and the BV method [4] with $T^{d-1}B+T^{d-2}\tilde{O}_\kappa(\alpha q \cdot \sqrt{n \log q})$, where T' is the max intermediate value, B is the noise in the fresh ciphertext and T is the max input value. But our method also has disadvantages, such as ciphertext expansion and some limits to the modulus q. We also explain the noise growth analysis in detail about homomorphic multiplication. As to homomorphic addition, the noise growth is small and the same as in other schemes.

4.1 Homomorphic Addition

The addition is performed in the same way as in [10] [4],we take the subgaussian analysis in [1].

Lemma 1. *For two ciphertexts $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{Z}_q^{n \times n\ell}$ which encrypt $\mu_1, \mu_2 \in \mathbb{Z}$ respectively with error vectors $\mathbf{e}_1, \mathbf{e}_2$, the homomorphic addition is:*

$$\mathsf{Add}(\mathbf{C}_1, \mathbf{C}_2) = [\mathbf{C}_1 + \mathbf{C}_2]_q \tag{1}$$

The result has a error vector $\mathbf{e}_1 + \mathbf{e}_2$.

4.2 Homomorphic Multiplication

About the homomorphic multiplication, we firstly take the subgaussian analysis method in [1].

Lemma 2. *For two ciphertexts $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{Z}_q^{n \times n\ell}$ which encrypt $\mu_1, \mu_2 \in \mathbb{Z}$ respectively with error vectors $\mathbf{e}_1, \mathbf{e}_2$, the homomorphic multiplication is:*

$$\mathsf{Mult}(\mathbf{C}_1, \mathbf{C}_2) = [\mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)]_q. \tag{2}$$

*This is a randomized procedure as shown in the **Claim** 2, because $\mathbf{G}^{-1}$ is randomized. The result has a error vector $\mathbf{e}^t + \mu_1 \mathbf{e}_2^t$, where the entries of $\mathbf{e}$ are independent and subgaussian with the parameter $O(||\mathbf{e}_1||)$.*

Now we consider computing the multiplication of two integers $\mu_1, \mu_2 \in \mathbb{Z}_q$.

$$\mu_1 \cdot \mu_2 = \mu_1 \cdot \mu_2[0] \cdot 2^0 + \mu_1 \cdot \mu_2[1] \cdot 2^1 + \ldots + \mu_1 \cdot \mu_2[\ell-1] \cdot 2^{\ell-1} \tag{3}$$

$$= \sum_{i=0}^{\ell-1} \mu_1 \cdot \mu_2[i] \cdot 2^i \tag{4}$$

$$= \sum_{i=0}^{\ell-1} \mu_2[i] \cdot \mu_1 \cdot 2^i \tag{5}$$

Where $\mu_2 = (\mu_2[0], \ldots, \mu_2[\ell-1]) \in \mathbb{Z}_2^\ell$ is the binary representation of μ_2. Correspondingly, we can homomorphically evaluating the above process:

$$\mathsf{Enc}(\mu_1 \cdot \mu_2) = \mathsf{Enc}(\sum_{i=0}^{\ell-1} \mu_2[i] \cdot \mu_1 \cdot 2^i) \tag{6}$$

$$= \sum_{i=0}^{\ell-1} \mathsf{Enc}(\mu_2[i]) \cdot \mathsf{Enc}(\mu_1) \cdot \mathsf{Enc}(2^i) \tag{7}$$

Clearly we need the BEnc to encrypt μ_2, we derive two Corollaries 2 and 3 for special q to show the noise growth property. Firstly, we show the limit on the selection of q.

Proposition 1. *There exists such q and a procedure $\mathbf{G}^{*-1} : \mathbb{Z}_q^{n \times n\ell} \to \mathbb{Z}^{n\ell \times n\ell}$ that for any $i \in [0, \ell)$, $\mathbf{M}_{2^i} = 2^i \cdot \mathbf{G} \in \mathbb{Z}_q^{n \times n\ell}$, if $\mathbf{X} = \mathbf{G}^{*-1}(\mathbf{M}_{2^i})$, then $\mathbf{G} \cdot \mathbf{X} = \mathbf{M}_{2^i}$. For any $\mathbf{e} \in \mathbb{Z}_q^{n\ell}$, $\mathbf{e}_a = \mathbf{e} \cdot \mathbf{X}$, $||\mathbf{e}_a|| \approx ||\mathbf{e}||$.*

Proof. Here we don't need the procedure to be randomize. It's easy to find such pair of q and $\mathbf{G}^{*-1}$. Take $q = 2^{\ell-1}$ as an example, $\mathbf{G}^{*-1}$ is the bit decomposition for every element in the matrix to a column vector $\mathbb{Z}^{\ell \times 1}$, ie, $2^3 \to \underbrace{(0,0,0,1,0,\ldots,0)}_{\ell}^t \in \mathbb{Z}^{\ell \times 1}$. It can be verified that $\mathbf{G} \cdot \mathbf{X} = \mathbf{M}$. Then $\mathbf{X} = \mathbf{G}^{*-1}(2^i \cdot \mathbf{G}) \in \mathbb{Z}^{n\ell \times n\ell}$ has no more than one non-zero element in every column, so for any $\mathbf{e} \in \mathbb{Z}_q^{n\ell}$, $||\mathbf{e}_a|| \approx ||\mathbf{e}||$. Here we use "$\approx$" if the difference is much smaller than the value $||\mathbf{e}||$. □

The constraint on q will not influence the efficiency or security. The reason is that given a security level, for any bit length ℓ, we can always find such q.

Corollary 2. *For such q and $\mathbf{G}^{*-1}$ that satisfies the **Proposition 1**, for two ciphertexts $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{Z}_q^{n \times n\ell}$ which encrypt $\mu_1, \mu_2 \in \mathbb{Z}$ respectively with error vectors $\mathbf{e}_1, \mathbf{e}_2$, and a constant $\mathbf{M}_{2^i} = 2^i \cdot \mathbf{G}$, the homomorphic multiplication is:*

$$\mathsf{Mult}(\mathbf{C}_1, \mathbf{C}_2, \mathbf{M}_{2^i}) = [([\mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)]_q) \cdot \mathbf{G}^{*-1}(\mathbf{M}_{2^i})]_q. \tag{8}$$

The result has a error vector $(\mathbf{e}^t + \mu_1 \mathbf{e}_2^t)$, where the entries of $\mathbf{e}$ are independent and subgaussian with the parameter $O(||\mathbf{e}_1||)$.

Proof. The correctness is straightforward, since this is the homomorphic evaluation process of the Equation (3). The multiplication result $[\mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)]_q$ has the noise vector $(\mathbf{e}^t + \mu_1 \mathbf{e}_2^t)$, where the entries of $\mathbf{e}$ are independent and subgaussian with the parameter $O(||\mathbf{e}_1||)$. According to the **Proposition 1**, the multiplication with $\mathbf{G}^{*-1}(\mathbf{M}_{2^i})]$ will not add extra noise, so the final noise is still $(\mathbf{e}^t + \mu_1 \mathbf{e}_2^t)$. □

Now we use the **Corollary 2** to get the noise growth of the homomorphic evaluation of the Equation (3).

Corollary 3. *For such q and $\mathbf{G}^{*-1}$ that satisfies the **Proposition 1**, for a ciphertexts $\mathbf{C}_1 \in \mathbb{Z}_q^{n \times n\ell}$ which encrypt $\mu_1 \in \mathbb{Z}$ with the error vector $\mathbf{e}_1$, and a vector of ciphertexts $(\mathbf{C}_2[i])_{i\in[\ell]}$ encrypting $(\mu_2[i])_{i\in[\ell]} \in \mathbb{Z}_2^\ell$ respectively with the noise vector $(\mathbf{e}_2^t[i])_{i\in[\ell]}$, and a vector of $(\mathbf{M}_{2^i})_{i\in[\ell]} = (2^i \cdot \mathbf{G})_{i\in[\ell]}$. The homomorphic evaluation of the Equation (3) is:*

$$\mathbf{C}_1 \boxtimes (\mathbf{C}_2[i])_{i\in[\ell]} = \left[\sum_{i=0}^{\ell-1} (\mathbf{C}_2[i] \cdot \mathbf{G}^{-1}(\mathbf{C}_1)) \cdot \mathbf{G}^{*-1}(\mathbf{M}_{2^i})\right]_q . \tag{9}$$

Here, we define the symbol $\boxtimes$ as the operation of one ciphertext with a vector of ciphertexts. The result has an error vector whose entries are independent and subgaussian with the parameter $O(||\mathbf{e}||)$, where

$$\mathbf{e}^t = [\underbrace{\mathbf{e}_1^t || \ldots \mathbf{e}_1^t}_{\ell} || \mathbf{e}_2^t[0] || \mathbf{e}_2^t[1] || \ldots || \mathbf{e}_2^t[\ell-1]] \in \mathbb{Z}^{2n\ell^2}.$$

Proof. Considering every $(\mathbf{C}_2[i] \cdot \mathbf{G}^{-1}(\mathbf{C}_1)) \cdot \mathbf{G}^{*-1}(\mathbf{M}_{2^i})$, the result has a error vector $(\mathbf{e}^t[i] + \mu_2[i]\mathbf{e}_1^t)$, where the entries of $\mathbf{e}^t[i]$ are fresh independent and subgaussian with the parameter $O(||\mathbf{e}_2[i]||)$. Since all the noise $\mathbf{e}_2[i]$ is independent, the final result has the error vector of the sum of $(\mathbf{e}^t[i] + \mu_2[i]\mathbf{e}_1^t)$ for all i. □

4.3 Homomorphic Polynomial Evaluation

For any integer monic polynomial

$$\mathsf{F}(x) = p_0 + p_1 \cdot x + \ldots + x^d \tag{10}$$

$$= p_0 + (p_1 + (p_2 + (\ldots (p_{d-1} + \underbrace{x) \cdot x \ldots) \cdot x) \cdot x}_{d} \tag{11}$$

To perform the calculation sequentially rather than as a tree will allow the right operand to be fresh ciphertext all the time , so that all the integer multiplication can be performed in the way as mentioned in Section 4.2.

Corollary 4. *For a polynomial $F(x) = p_0 + p_1 \cdot x + \ldots + x^d$ with all its coefficients $p_i \in \mathbb{Z}_q$, for such q and $\mathbf{G}^{*-1}$ that satisfies the **Proposition 1**, a vector of ciphertexts $(\mathbf{C}[i])_{i\in[0,\ell)}$ encrypting $(\mu[i])_{i\in[0,\ell)} \in \mathbb{Z}_2^\ell$ respectively with the noise vector $(\mathbf{e}[i])_{i\in[0,\ell)} \in \mathbb{Z}_2^\ell$, and a vector $(\mathbf{M}_{p_j})_{j\in[0,d)}$ encoding the constant coefficient $(p_j)_{j\in[0,d)}$. The homomorphic evaluation of $\mathsf{F}(\mu)$ is:*

$$\begin{aligned}\mathsf{Eval}((\mathbf{C}[i])_{i\in[0,\ell)}, \mathsf{F}) = &\mathbf{M}_{p_0} + (\mathbf{M}_{p_1} + (\ldots (\mathbf{M}_{p_{d-1}} + \\ &\underbrace{\mathbf{G} \boxtimes (\mathbf{C}[i])_{i\in[0,\ell)}) \boxtimes (\mathbf{C}[i])_{i\in[0,\ell)}) \ldots) \boxtimes (\mathbf{C}[i])_{i\in[0,\ell)})}_{d}\end{aligned}$$

*Here the multiplication $\boxtimes(\mathbf{C}[i])_{i\in[0,\ell)}$ is performed as said in the **Corollary 3**. The result has an error vector whose entries are independent and subgaussian with the parameter $O(||\mathbf{e}'||)$, where*

$$(\mathbf{e}')^t = [\ \underbrace{\mathbf{e}_p^t || \ldots || \mathbf{e}_p^t}_{(\ell^{d-1}-1)/(\ell-1)}\ ],\ \ \mathbf{e}_p^t = [\mathbf{e}^t[0] || \mathbf{e}^t[1] || \ldots || \mathbf{e}^t[\ell-1]] \in \mathbb{Z}^{n\ell^2}.$$

Proof. Considering the addition with $\mathbf{M}_{p_i}$, which has no noise, we can see the polynomial as an integer multiplication chain, which is $x^d = \underbrace{x \cdot \ldots \cdot x}_{d}$. For every integer multiplication of one intermediate ciphertext(with the noise vector $\mathbf{e}_1$) with encrypted $(\mu[i])_{i\in[0,\ell)}$(with the noise vector $(\mathbf{e}[i])_{i\in[0,\ell)}$), the noise growth is $\sum_{i=0}^{\ell-1}(\mathbf{e}^t[i] + \mu[i]\mathbf{e}^t)$, the noise $\mathbf{e}_1$ and $(\mathbf{e}[i])$ are independent of each other because of the randomized procedure $\mathbf{G}^{-1}$, so iteratively we can calculate that the overall noise growth is $\sum_{j=0}^{j<(\ell^{d-1}-1)/(\ell-1)} \sum_{i=0}^{\ell-1}(\mathbf{e}^t[i])$. □

4.4 Comparison to Other Methods

Here, we briefly compare our method with existing methods for the homomorphically evaluating integer polynomials of degree d:

- In [10], when all the intermediate values are bounded by T', the depth of the evaluated circuit is $L = \log d$, and the fresh ciphertext noise is B, the final ciphertext's noise will be bounded by $(N+T')^{\log d} \cdot B$, which is proportional to $T'^{\log d} \approx 2^{\ell \cdot \log d} = d^\ell$. Our scheme has the noise growth of $O(\ell^d)$, so compared with the original GSW, ours has an advantage over low degree, large plaintext space application. But we also have a disadvantage of ciphertext expansion, ℓ times memory compared with the GSW scheme. The homomorphic multiplication computation cost is also about ℓ times.
- In [4], it's suggested to evaluate the circuit sequentially rather than as a binary tree, so the circuit depth $L = d$ rather than $L = \log d$. The benefit is that the multiplication chain have smaller additive noise, suppose the right side is always a fresh binary ciphertext. But if we directly use this method to evaluate an integer polynomial, the size of plaintext in a fresh ciphertext is not binary any more, suppose T, which cause the noise growth to be

  $$T^{d-1}B + T^{d-2}\tilde{O}_\kappa(\alpha q \cdot \sqrt{n \log q})$$

  which is proportional to $T^{d-1} \approx (2^{\ell \cdot d}) \gg \ell^d$.

5 New Method to Pack GSW FHE

5.1 Packed GSW FHE and the Method

We explore the possibility of packing more integers into a ciphertext by modifying the scheme from [11] which only deal with binary messages, and try to allow SIMD operations. First, we recap the scheme from [11].

Let λ be the security parameter, r be the number of integers to be encrypted, L be the depth of evaluated circuit.

$\mathsf{Setup}(\lambda, L, r)$: Choose a modulus $q = q(\lambda, L)$, lattice dimension $n = n(\lambda, L)$. $\ell = \lceil \log q \rceil$. $N = (n+r)\ell$. $\mathbf{G} = \mathbf{g}^t \otimes \mathbf{I}_{n+r} \in \mathbb{Z}_q^{(n+r)\times(n+r)\ell}$. The distribution χ as subgaussian over $\mathbb{Z}$. $m = O((n+r)\log q)$. Output $params = (r, m, n, q, N, \chi, \mathbf{G})$.

$\mathsf{KeyGen}(params)$: $\mathbf{A} \xleftarrow{\mathbf{U}} \mathbb{Z}_q^{n\times m}$, $\mathbf{S}' \xleftarrow{\mathbf{R}} \chi^{r\times n}$, $\mathbf{E} \xleftarrow{\mathbf{R}} \chi^{r\times m}$. Let $\mathbf{S} = [\mathbf{I}|| - \mathbf{S}'] \in \mathbb{Z}_q^{r\times(n+r)}$. Set

$$\mathbf{B} = \begin{pmatrix} \mathbf{S}'\mathbf{A} + \mathbf{E} \\ \mathbf{A} \end{pmatrix} \in \mathbb{Z}_q^{(n+r)\times m}$$

Let $\mathbf{M}_{(i,j)} \in \mathbf{Z}_2^{r\times r}(i,j = 1,\ldots,r)$ be the matrix with 1 in the (i,j)-th position and 0 in the other. For all $i,j = 1,\ldots,r$, first sample $\mathbf{R}_{(i,j)} \in \mathbb{Z}_2^{m\times N}$, and set

$$\mathbf{P}_{(i,j)} = \mathbf{B}\mathbf{R}_{(i,j)} + \begin{pmatrix} \mathbf{M}_{(i,j)}\mathbf{S} \\ \mathbf{0} \end{pmatrix} \mathbf{G} \in \mathbb{Z}_q^{(n+r)\times N}$$

Output $pk = (\{\mathbf{P}_{(i,j)}\}_{i,j\in[r]}, \mathbf{B})$ and $sk = \mathbf{S}$.

$\mathsf{Enc}(params, pk, \mathbf{M} \in \mathbb{Z}^{r\times r})$: $\mathbf{R}_{(i,j)} \xleftarrow{\mathbf{U}} \mathbb{Z}_2^{m\times N}$, output the ciphertext

$$\mathbf{C} = \mathbf{B}\mathbf{R} + \sum_{i,j\in[r]:\mathbf{M}[i,j]=1} \mathbf{P}_{(i,j)} \in \mathbb{Z}_q^{(n+r)\times N}$$

Where $\mathbf{M}[i,j]$ is the (i,j)-th element of $\mathbf{M}$.

$\mathsf{Dec}(params, sk, \mathbf{C})$: For $q = 2^{\ell-1}$, to decode $\mathbf{M} \in \mathbb{Z}_q^{r\times r}$, we calculate $\mathbf{SC} = \mathbf{MSG} + \mathbf{E}$. For the integer μ in the position (i,i) of $\mathbf{M}$, select the i row, $i\cdot\ell+1$ to $(i+1)\cdot\ell-1$ columns of $\mathbf{SC}$ as $\mathbf{C}_i$. Then $\mathbf{C}_i = \mu\cdot\mathbf{g} + \mathbf{e}$, where $\mathbf{g} = (1,2,\ldots,2^{l-2})$. Recover $LSB(\mu)$ from $\mu\cdot 2^{\ell-2} + \mathbf{e}_{\ell-2}$, then the next-LSB from $(\mu - LSB(\mu))\cdot 2^{l-3} + \mathbf{e}_{\ell-3}$, etc.

The scheme from [11] uses matrix plaintexts, ciphertexts and keys in order to realize the packing, so we put the integers here diagonally to achieve parallel homomorphic multiplications. We introduce a new encryption procedure PBEnc.

$\mathsf{PBEnc}(params, pk, (\mu_j)_{j\in[1,r]} \in \mathbb{Z}_q^r)$: For r input integers $(\mu_j)_{j\in[1,r]} \in \mathbb{Z}_q^r$, decompose them into binary representations, which include ℓ vectors, such that for $i \in [0,\ell)$, the weight is 2^i and the corresponding plaintext matrix is $\mathbf{M}[i] = \mathrm{diag}(\mu_1[i],\ldots,\mu_r[i]) \in \mathbb{Z}_2^{r\times r}$, then encrypt the matrix with $\mathbf{C}[i] = \mathsf{Enc}(params, pk, \mathbf{M}[i])$. The final ciphertext is $(\mathbf{C}[i])_{i\in[1,\ell)}$, together with a weight 2^i for every $(\mathbf{C}[i])_{i\in[1,\ell)}$.

For a known constant scalar $\alpha \in \mathbb{Z}_q$, we define $\mathbf{M}_\alpha = [\alpha\mathbf{G}]_q$.

5.2 Correctness and Security

The correctness analysis is about the same with the unpacked scheme,

Definition 4. *For a ciphertext $\mathbf{C} \in \mathbb{Z}_q^{(n+r)\times N}$, it's designed to encrypt $\mathbf{M}$ with $(\mu_i)_{i\in[1,r]} \in \mathbb{Z}_q^r$ in the diagnal, under the secret key $\mathbf{S}$, if there is an error matrix $\mathbf{E}^t \in \mathbb{Z}^{n\ell}$, and*

$$\mathbf{S}^t\mathbf{C} - \mathbf{MSG} = \mathbf{E} \bmod q$$

This naturally comes from the proof in [11] and from [16], if $||\mathbf{E}||_\infty < q/8$, $\mathsf{Dec}(params, \mathbf{s}, \mathbf{C})$ can correctly output μ.

Since the packed GSW FHE variant in [11] is IND-CPA secure based on the $\mathbf{DLWE}_{n,m,q,\chi}$ assumption, it is clear that adopting the new encryption procedure BEnc will keep the same security property.

5.3 Homomorphic operations

For homomorphic addition and multiplication, we can easily extend the proof in [11] to the $\mathbf{Z}_q$ field.

Corollary 5. *For two ciphertexts $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{Z}_q^{(n+r)\times N}$ which encrypt $\mathbf{M}_1, \mathbf{M}_2 \in \mathbb{Z}_q^{r\times r}$ respectively with error matrixes $\mathbf{E}_1, \mathbf{E}_2$, the homomorphic addition is:*

$$\mathsf{Add}(\mathbf{C}_1, \mathbf{C}_2) = [\mathbf{C}_1 + \mathbf{C}_2]_q. \tag{12}$$

The homomorphic multiplication is:

$$\mathsf{Mult}(\mathbf{C}_1, \mathbf{C}_2) = [\mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)]_q \tag{13}$$

This is a randomized procedure, because $\mathbf{G}^{-1}$ is randomized. The result has a error matrix $\mathbf{E}+\mathbf{M}_1\mathbf{E}_2$, where the entries of $\mathbf{E}$ has in the i-th row the independent and subgaussian entries with the parameter $O(||\mathbf{e}_{1,i}||)$, $\mathbf{e}_{1,i}$ is the i-th row of $\mathbf{E}_1$.

We then consider the packed integer matrix multiplication

$$\begin{aligned}\mathbf{M}_1 \cdot \mathbf{M}_2 &= \mathrm{diag}(\mu_{1,1}, \ldots, \mu_{1,r}) \cdot \mathrm{diag}(\mu_{2,1}, \ldots, \mu_{2,r}) \\ &= \sum_{i=0}^{\ell-1} \mathrm{diag}(\mu_{2,1}[i], \ldots, \mu_{2,r}[i]) \cdot \mathrm{diag}(\mu_{1,1}, \ldots, \mu_{1,r}) \cdot \mathrm{diag}(2^i, \ldots, 2^i)\end{aligned} \tag{14}$$

where $\mu_{2,j} = (\mu_{2,j}[0], \mu_{2,j}[1], \ldots, \mu_{2,j}[\ell-1]) \in \mathbb{Z}_2^\ell$ is the binary representation of the element $\mu_{2,j}$ in the matrix $\mathbf{M}_2$, for $j \in [1, r]$.

In the evaluation, if we always keep the $\mathbf{M}_2$ as a fresh ciphertext vector encrypted with our encryption procedure PBEnc, then the noise growth rate can be calculated as follows. It's straightforward that we can get q and $\mathbf{G}^{*-1}$ that satisfies the **Proposition** 1, the randomized procedure from the **Claim** 2, because the only difference is that n is changed to $n + r$.

Corollary 6. *For such q and $\mathbf{G}^{*-1}$ that satisfies the Proposition 1 , for a ciphertexts $\mathbf{C}_1 \in \mathbb{Z}_q^{(n+r)\times N}$ which encrypt $\mathbf{M}_1 \in \mathbb{Z}_q^{r\times r}$ with the error matrix $\mathbf{E}_1$, and a vector of ciphertexts $(\mathbf{C}_2[i])_{i\in[\ell]}$ encrypting $(\mathbf{M}_2[i])_{i\in[\ell]}$ respectively with the noise matrix $(\mathbf{E}_2[i])_{i\in[\ell]}$, and a vector of ciphertexts $(\mathbf{M}_{2^i})_{i\in[0,\ell)} = (2^i \cdot \mathbf{G})_{i\in[0,\ell)}$. the homomorphic evaluation of the Equation (14) is:*

$$\mathbf{C}_1 \boxtimes (\mathbf{C}_2[i])_{i\in[\ell]} = [\sum_{i=0}^{\ell-1} (\mathbf{C}_2[i] \cdot \mathbf{G}^{-1}(\mathbf{C}_1)) \cdot \mathbf{G}^{*-1}(\mathbf{M}_{2^i})]_q. \tag{15}$$

This is a randomized procedure, because $\mathbf{G}^{-1}$ *is randomized. Let* $\mathbf{e}_{2,j}[i]$, $\mathbf{e}_{1,j}$, $\mathbf{e}_j$ *be the* j*-th row vector of* $\mathbf{E}_2[i]$, $\mathbf{E}_1$ *and the final ciphertext noise respectively, then* $\mathbf{e}_j$ *has independent and subgaussian entries with the parameter* $O(||\mathbf{e}_j||)$*, where*

$$\mathbf{e}_j = [\underbrace{\mathbf{e}_{1,j}||\ldots||\mathbf{e}_{1,j}}_{\ell}||\mathbf{e}_{2,j}[0]||\mathbf{e}_{2,j}[1]||\ldots||\mathbf{e}_{2,j}^t[\ell-1]] \in \mathbb{Z}^{2n\ell^2}.$$

The proof can be done similar to the **Corollary 3**, while the only difference is that we treat the error in every row in the ciphertext independently. We skip the details here.

For the polynomial evaluation, we can perform noise growth analysis similar to the **Corollary 4**. Considering every polynomial evaluation as a matrix multiplication chain, which is $\mathbf{M}^d = \underbrace{\mathbf{M}\cdot\ldots\cdot\mathbf{M}}_{d}$. The input is a vector of ciphertexts $(\mathbf{C}[i])_{i\in[\ell]}$ with the noise $(\mathbf{E}[i])_{i\in[\ell]}$. For every multiplication, the noise of the two ciphertexts is independent of each other because of the randomized procedure with $\mathbf{G}^{-1}$, so iteratively we can calculate that the overall noise growth is $\sum_{j=0}^{j<(\ell^{d-1}-1)/(\ell-1)}\sum_{i=0}^{\ell-1}(\mathbf{E}[i])$. For the j-th row in the final ciphertext noise, it has independent subgaussian with parameters $O(||\mathbf{e}'_j||)$, where $\mathbf{e}'_j = \sum_{j=0}^{j<(\ell^{d-1}-1)/(\ell-1)}\sum_{i=0}^{\ell-1}(\mathbf{e}_j[i])$, $\mathbf{e}_j[i]$ being the j-th row of $\mathbf{E}[i]$.

6 Integration with Ciphertext Compression

Traditionally, to securely evaluate a specific function with an untrusted server, the user needs to homomorphically encrypt the data, then upload the ciphertest to the server who can then do perform the computations. Usually, the ciphertext size is very large compared with the original plaintext, and it is referred to as the ciphertext expansion problem. To avoid this problem, [18] proposed to send the data encrypted with a block cipher such as AES (can be symmetrical ciphers) to the cloud, so that the server can homomorphically decrypt them and perform evaluation afterwards. However, existing symmetrical ciphers encrypt the data with non-linear functions and boolean circuits such as XOR, AND, which require the plaintext to be encrypted as binary vectors. This forces the underlying homomorphic scheme to have a binary plaintext space too, which is quite inefficient in dealing with the integer polynomial evaluation. For example, for two numbers with the bit length ℓ, the homomorphic integer multiplication needs $O(\ell^2)$ multiplications, with the depth $O(\ell)$.

Next, we first choose the optimal(low-depth) symmetrical cipher for the ciphertext compression , then show how to combine it with our evaluation methods.

For symmetrical ciphers, the depth of the decryption circuit need to be small enough to allow further homomorphic evaluations. It is because, for somewhat homomorphic encryption, the total depth for a specific parameters set is fixed. To minimize the overhead of the decryption circuit, different block ciphers and

stream ciphers have been investigated, e.g. in [5,15]. The block ciphers such as AES, Simon-32/64, have a lot of rounds to guarantee the security level, which results a high decryption depth such as 32 for Simon-32/64, 44 for Simon-64/128, 40 for AES-128. The stream ciphers, on the other hand, has increasing noise with the number of decrypted ciphertext blocks, since the homomorphic pseudorandom keystream generation will add noise to the encrypted secret key. To tackle this problem, [15] proposed the stream cipher **FLIP** based on filter permutator, which has such property that the non-linear filtering function always acts on the key bits, rather than the previous output of some function, so the noise level of every decrypted ciphertext is constant. Combined with the additive noise property of the multiplicative chain in GSW, the decryption noise can be restrict to a small number. As shown in [15], for 80-bit and 128-bit security level, **FLIP** has a multiplicative depth of 4.

Here we propose to concatenate the symmetrical ciphertext decryption with the integer polynomial evaluation circuit. The homomorphic symmetric key decryption needs NAND which will restrict the message space to $\mathbb{Z}_2$ and maintain small noise increase, while the polynomial evaluation needs Add and Mult. What's more, we can integrate the batching method. A high-level workflow is as follows.

- On the user side, for plaintext integers $(\mu_1, \mu_2, \ldots, \mu_r)$, decompose every integer into binary representations

  $$(\mu_1[0], \ldots, \mu_1[\ell-1], \mu_2[0], \ldots, \mu_2[\ell-1], \ldots, \mu_r[0], \ldots, \mu_r[\ell-1])$$

  then aggregate the same weight into the same block such as

  $$(\mu_1[0], \mu_2[0], \ldots, \mu_t[0]), (\mu_1[1], \mu_2[1], \ldots, \mu_r[1]), \cdots$$

  For every block, encrypt it with symmetrical encryption, send it to the server together with the corresponding weight.
- On the server side, homomorphically decrypt all blocks into ciphertexts (of the FHE), then homomorphically evaluate the integer polynomial and send the encrypted result back to the user.
- After receiving the encrypted result, the user can decrypt it to get the evaluated result.

With our method, an integer multiplication needs $O(\ell)$ homomorphic multiplications and a depth of 1, which is more efficient than the original $O(\ell^2)$ and $O(\ell)$ complexity. Besides, in GSW scheme, we observe that the parameters (n, q) are only related to the noise growth and the security level, and for homomorphic symmetrical cipher decryption which is mainly boolean operations, the noise growth is small, so the overhead can be acceptable. Even if it's impossible to get fresh ciphertexts for the integer polynomial evaluation, it can be regarded that the homomorphically symmetrical decrypted ciphertext is the "fresh" ciphertext with a larger noise.

7 Conclusion

In this paper, we have proposed some methods to improve the integer polynomial evaluation based on GSW FHE scheme. In fact, these methods can be further extended to multi-variant polynomials and Ring-GSW, or a combination with other optimizations such as Chinese Remainder Theorem. To make a fair comparison with the state-of-art HE schemes such as YASHE, detailed security/parameters/homomorphic operations analysis remain to be a future work.

References

1. Alperin-Sheriff, J., Peikert, C.: Faster bootstrapping with polynomial error. In: Advances in Cryptology–CRYPTO 2014, pp. 297–314. Springer (2014)
2. Bos, J.W., Lauter, K., Loftus, J., Naehrig, M.: Improved security for a ring-based fully homomorphic encryption scheme. In: Cryptography and Coding, pp. 45–64. Springer (2013)
3. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. pp. 309–325. ACM (2012)
4. Brakerski, Z., Vaikuntanathan, V.: Lattice-based fhe as secure as pke. In: Proceedings of the 5th conference on Innovations in theoretical computer science. pp. 1–12. ACM (2014)
5. Canteaut, A., Carpov, S., Fontaine, C., Lepoint, T., Naya-Plasencia, M., Paillier, P., Sirdey, R.: Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In: FSE 2016. p. to appear in (2016)
6. Dowlin, N., Gilad-Bachrach, R., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy (2016), `research.microsoft.com/pubs/260989/CryptonetsTechReport.pdf`
7. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive 2012, 144 (2012)
8. Gentry, C.: A fully homomorphic encryption scheme. Thesis, Stanford University (2009)
9. Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the aes circuit. In: Advances in Cryptology–CRYPTO 2012, pp. 850–867. Springer (2012)
10. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Advances in Cryptology–CRYPTO 2013, pp. 75–92. Springer (2013)
11. Hiromasa, R., Masayuki, A., Okamoto, T.: Packing messages and optimizing bootstrapping in gsw-fhe. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences 99(1), 73–82 (2016)
12. Khedr, A., Gulak, G., Vaikuntanathan, V.: Shield: Scalable homomorphic implementation of encrypted data-classifiers. IEEE Transactions on Computers PP(99), to appear (2015)
13. Lepoint, T., Naehrig, M.: A comparison of the homomorphic encryption schemes fv and yashe. In: Progress in Cryptology–AFRICACRYPT 2014, pp. 318–335. Springer (2014)

14. Lpez-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the forty-fourth annual ACM symposium on Theory of computing. pp. 1219–1234. ACM (2012)
15. Maux, P., Journault, A., Standaert, F.X., Carlet, C.: Towards stream ciphers for efficient fhe with low-noise ciphertexts. In: Advances in Cryptology-EUROCRYPT. p. to appear (2016)
16. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. Advances in Cryptology-EUROCRYPT 2012 pp. 700–718 (2012)
17. Naehrig, M., Lauter, K., Vaikuntanathan, V.: Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on Cloud computing security workshop. pp. 113–124. ACM (2011)
18. Naehrig, M., Lauter, K., Vaikuntanathan, V.: Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on Cloud computing security workshop. pp. 113–124. ACM (2011)
19. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 333–342. ACM (2009)
20. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM 56(6), 1–40 (2009)
21. Schnorr, C.P.: A hierarchy of polynomial time lattice basis reduction algorithms. Theoretical computer science 53(2), 201–224 (1987)
22. Vershynin, R.: Compressed sensing (2012), `www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf`
23. Wang, S., Zhang, Y., Dai, W., Lauter, K., Kim, M., Tang, Y., Xiong, H., Jiang, X.: Healer: homomorphic computation of exact logistic regression for secure rare disease variants analysis in gwas. Bioinformatics 32(2), 211–8 (2016), `http://www.ncbi.nlm.nih.gov/pubmed/26446135`