Fully Homomorphic Encryption with k-bit Arithmetic Operations

Benjamin M. Case*[1], Shuhong Gao*[1], Gengran Hu†[2], and Qiuxia Xu‡[3]

[1]*School of Mathematical and Statistical Sciences, Clemson University, Clemson, SC 29634, USA*
[2]*School of Cyberspace, Hangzhou Dianzi University, Hangzhou, 310018, China*
[3]*School of Mathematics and Information Science, Guangzhou University, Guangzhou, 510006, China*

May 18, 2019

Abstract

We present a fully homomorphic encryption scheme continuing the line of works of Ducas and Micciancio (2015, [29]), Chillotti et al. (2016, [23]; 2017, [24]; 2018, [25]), and Gao (2018,[32]). Ducas and Micciancio (2015) show that homomorphic computation of one bit operation on LWE ciphers can be done in less than a second, which is then reduced by Chillotti et al. (2016, 2017, 2018) to 13ms. According to Chillotti et al. (2018, [26]), the cipher expansion for TFHE is still 8000. The ciphertext expansion problem was greatly reduced by Gao (2018) to 6 with private-key encryption and 20 for public key encryption. The bootstrapping in Gao (2018) is only done one bit at a time, and the bootstrapping design matches the previous two works in efficiency.

Our contribution is to present a fully homomorphic encryption scheme based on these preceding schemes that generalizes the Gao (2018) scheme to perform operations on k-bit encrypted data and also removes the need for the Independence Heuristic of the Chillotti et al. papers. The amortized cost of computing k-bits at a time improves the efficiency. Operations supported include addition and multiplication modulo 2^k, addition and multiplication in the integers as well as exponentiation, field inversion and the machine learning activation function RELU. The ciphertext expansion factor is also further improved, for $k = 4$ our scheme achieves a ciphertext expansion factor of 2.5 under secret key and 6.5 under public key. Asymptotically as $k \to \infty$, our scheme achieves the optimal ciphertext expansion factor of 1 under private key encryption and 2 under public key encryption. We also introduces techniques for reducing the size of the bootstrapping key.

Keywords. FHE, lattices, learning with errors (LWE), ring learning with errors (RLWE), TFHE, data security, RELU, machine learning

*The work was partially supported by the National Science Foundation under grants CCF-1407623, DMS-1403062 and DMS-1547399. Email: {bmcase, sgao}@g.clemson.edu

†The work was partially supported by the National Natural Science Foundation of China under grant 61602143 and was done during his stay at Clemson University as a Visiting Scholar.

‡The work was partially supported by the National Natural Science Foundation of China under grant 61772147 and was done during her stay at Clemson University as a Visiting Scholar.

1 Introduction

Homomorphic encryption (HE) allows computations to be done directly on encrypted data offering a great solution to data privacy in cloud computing environments. It was first proposed in 1978 by Rivest, Adleman and Dertouzos [55]. Fully homomorphic computing (FHE) allows for any function of any complexity to be computed on the encrypted data; the first FHE scheme was discovered in 2009 by Gentry [33] who introduced the idea of bootstrapping to make the scheme fully homomorphic. Many schemes have followed in this blueprint for FHE. Other works have taken a more pragmatic approach investigating somewhat homomorphic schemes (SHE) which allow for some restricted family of functions to be performed on the encrypted data.

On bootstrapping for FHE, a recent breakthrough is made by Ducas and Micciancio (2015 [29]), who use the GSW scheme [34] and some novel homomorphic embedding to design a bootstrapping procedure that can compute one homomorphic bit operation in less than a second. This scheme is then improved by Chillotti et al. (2016 [23], 2017, [24]; 2018, [25, 26]), who reduce the bootstrapping time down to 13ms per homomorphic bit operation. On cipher expansion, it is stated in [23] that the ciphertext size is still 400,000 times that of the original data and noted in (2018, [26]) that the expansion factor has been be reduced to 8000. Collectively these schemes are known as TFHE. On ciphertext expansion, a recent breakthrough is made by Gao (2018, [32]) reducing the ciphertext expansion to 6 under private key encryption and 20 under public key encryption. Another recent technique of Biasse and Ruiz (2015 [11]) allows for homomorphically computing arbitrary lookup functions on encrypted data.

Our contribution. In this paper, we present a compact FHE scheme that has the following features:

1. We generalize the scheme in [32] to work for k-bit encrypted messages and thus achieve even better ciphertext expansion ratios, e.g. with $k = 4$ the expansion ratio under private key is 2.5 and under public key is 6.5, and we remove the need for the Independence Heuristic in the TFHE schemes.

2. Our scheme supports a variety of multibit operations including k-bit addition and multiplication modulo 2^k and addition and multiplication in the integers. Performing k-bit operations gives an improved amortized cost for homomorphic computing compared to computing only one bit at a time.

3. We also incorporate the low ciphertext expansion under private key and other new techniques to reduced the size of the bootstrapping key to around 1128 MB for $k = 1$ (down from around 12GB for similar parameters in [32]).

4. The probability of failure for each bootstrapping operation is less that 2^{-140} thus practically any number of computations can be done.

5. Our scheme with suggested parameters has at least 128 bits of security.

In our scheme, the LWE ciphers after bootstrapping are always in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size bounded $4\sqrt{n}$ with probability greater than $1 - 2^{-140}$, thus allowing practically any number of computations.

Organization of the paper. In Section 2 we present a high level overview. In Section 3 we include some background material including LWE, RLWE, and GSW ciphers. We also present randomized flattening and the external product, which have appeared in the literature in one form or another, but we present them in an exact form that is important for our scheme. In Section 4, we present our encryption schemes and show how RLWE ciphers can be unpacked to LWE ciphers. In Section 5, we present our bootstrapping procedure, which is modified from those of [29, 23, 11, 17] along with homomorphic operations our scheme supports. In Sections 6 and 7, we present the parameters to instantiate an instance of our scheme along with some efficiency and security analysis.

2 High Level Overview

In this section we give a high level overview of the scheme with a focus on the scheme's functionality and usage.

Key Generation. A user generates a secret key sk, which will be used for secret key encryption and decryption, a public key pk, which will be used for public key encryption, and finally a boostrapping key bk which will be used for performing fully homomorphic computations by any (untrusted) third party.

Encryption and Storage. To encrypt with our scheme, data is first encoded as vectors of length n (e.g. $n = 2^{12}$) with k-bit entries $x_i \in \{0, 2^k - 1\}$ (e.g. $k = 4$). Each vector is then converted to a message polynomial $m(x) = \sum_{i=0}^{n-1} x_i x^i$ which is encrypted as an RLWE cipher, as shown in section Section 4. This encryption can be done using either the secret key or the public key.

$$\text{SecEncrypt}(sk, m(x)) \to \mathbf{c} \equiv (a(x), b(x))$$

and

$$\text{PubEncrypt}(pk, m(x)) \to \mathbf{c} \equiv (a(x), b(x))$$

The encrypted data $\mathbf{c}$ can then be stored in an untrusted cloud. For $k = 4$, secret key encryption has a ciphertext expansion factor of about 2.5, and public key encryption has an expansion factor of 6.5.

Homomorphic Computing. Homomorphic computations cannot be done directly on these RLWE ciphers; instead, they must first be unpacked into LWE ciphers (details given at the end of Section 4). With this unpacking, one gets an LWE cipher for each coefficient of the message polynomial $m(x)$: $c_i = \mathbf{E}_s(x_i)$ which is a vector of length $n+1$ with entries from $\mathbb{Z}_r$. These LWE ciphers have a much bigger expansion factor, so are extracted from storage during computation only when needed.

Computations on encrypted data can now be performed on these unpacked LWE ciphers. For any two LWE ciphers $c_1 = \mathbf{E}_s(x_1)$ and $c_2 = \mathbf{E}_s(x_2)$ where x_1 and x_2 are any k bit integers and c_1 and c_2 may come from the original encryption $\mathbf{c}$ or new LWE ciphers during homomorphic computing, our scheme supports a variety of operations and all the new LWE ciphers computed are still LWE ciphers (over the same $\mathbb{Z}_r$ and with the same error bound).

The operations supported include the addition and subtraction modulo $p \leq 2^k$, multiplication modulo $p \leq 2^k$ for p odd or a power of 2:

1. Addition modulo $p \leq 2^k$,
$$\text{Addmodp}(\mathbf{E}_s(x_1), \mathbf{E}_s(x_2), p) = \mathbf{E}_s(x_1 + x_2 \bmod p).$$
2. Subtraction modulo $p \leq 2^k$,
$$\text{Submodp}(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) = E_{\mathbf{s}}(x_1 - x_2 \bmod p).$$
3. Multiplication modulo $p < 2^k$ odd,
$$\text{Multmodpodd}(\mathrm{E}_{\mathbf{s}}(x_1), \mathrm{E}_{\mathbf{s}}(x_2), p) = \mathrm{E}_{\mathbf{s}}(x_1 \cdot x_2 \bmod p)$$

4. Multiplication modulo 2^k,

$$\text{Multmod2tok}(\text{E}_{\mathbf{s}}(x_1), \text{E}_{\mathbf{s}}(x_2)) = \text{E}_{\mathbf{s}}(x_1 \cdot x_2 \bmod 2^k).$$

Also supported are the following two operations in the integers that store the result across two new ciphertexts:

1. Addition in $\mathbb{Z}^+$,
$$\text{AddinZ}(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) = (E_{\mathbf{s}}(y_0), E_{\mathbf{s}}(y_1)),$$
where $x_1 + x_2 = y_0 + y_1 2^k \in \mathbb{Z}$, y_0 has k bit, and y_1 has one bit.

2. Multiplication in $\mathbb{Z}^+$,
$$\text{MultinZ}(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) = (E_{\mathbf{s}}(y_0), E_{\mathbf{s}}(y_1)),$$
where $x_1 \cdot x_2 = y_0 + y_1 2^k \in \mathbb{Z}$ and both y_0 and y_1 have k bits.

The above are all binary operations which combine two ciphertexts; we can also perform the following unary operations on a single ciphertext:

1. Inverse modulo $p \leq 2^k$, with any integer x so that $\gcd(x, p) = 1$,
$$\text{Inversemodp}(\text{E}_{\mathbf{s}}(x), p) = \text{E}_{\mathbf{s}}(x^{-1} \bmod p).$$

2. Power modulo $p \leq 2^k$, with any integer $i > 1$,
$$\text{Powermodp}(\text{E}_{\mathbf{s}}(x), i, p) = \text{E}_{\mathbf{s}}(x^i \bmod p).$$

3. RELU for any integer $x \in (-2^k, 2^k)$,
$$\text{RELU}(\text{E}_{\mathbf{s}}(x)) = \text{E}_{\mathbf{s}}(x^+)$$
where $x^+ := \max\{0, x\}$.

Each of these LWE ciphers is valid with a probability greater than $1 - 2^{-140}$, so practically any arbitrary number of computations can be performed while still preserving a high probability of correctness, (e.g. if 2^{50} homomorphic computations are done, the probability that the resulted LWE cipher is valid is still $1 - 2^{-90}$). After the desired homomorphic computations have been performed by some untrusted party, the result can be returned to the user encrypted as LWE ciphers, or if the number of such LWE ciphers is close to n or larger, the LWE ciphers can be packed back into RLWE ciphers for storage in the database or to be sent to the user. This packing technique was first introduced in [32].

3 Background concepts and techniques

3.1 Rings and norms

For q a positive integer we denote the ring of integers modulo q as $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$. In order to speak of the norms of elements in $\mathbb{Z}_q$, we must fix a set of representatives, which we choose to be $[-\lfloor \frac{q}{2} \rfloor, ..., \lfloor \frac{q}{2} \rfloor]$ for q odd and $(-\frac{q}{2}, ..., \frac{q}{2}]$ for q even, and we call these the *norm representatives*. We then define the norm of $a \in \mathbb{Z}_q$ as the absolute value of its equivalent norm representative, e.g. for $7, 4 \in \mathbb{Z}_5$, $||7|| = |2|$ and $||4|| = |-1|$. Note we still have the inequality $||a + b|| \leq ||a|| + ||b||$ for all $a, b \in \mathbb{Z}_q$.

For any polynomial $f(x) = \sum_{i=0}^{d} f_i x^i \in \mathbb{R}[x]$ and for $\ell \geq 1$, we define the ∞-norm as $\|f(x)\|_\infty := \max_{0\leq i\leq d} |f_i|$. For an integers $n, q \geq 1$, let

$$R_n := \mathbb{Z}[x]/(x^n+1) \quad R_{n,q} := \mathbb{Z}[x]/(x^n+1, q)$$

For $f(x) = \sum_{i=0}^{d} f_i x^i$ in $R_{n,q}$ we define its norm as follows. First find the unique $h(x) \in \mathbb{Z}_q[x]$ so that $\deg h(x) < n$ and $f(x) \equiv h(x) \bmod (x^n+1)$ (i.e. $h(x)$ is the remainder of $f(x)$ modulo x^n+1), then fix each coefficient of $h(x)$ to be a norm representative of $\mathbb{Z}_q$ and then $||f(x)||_\infty := ||h(x)||_\infty$.

For $m \geq 1$, elements in $R_{n,q}^m$ are viewed as row vectors of length m. For $\mathbf{u} = (u_1(x), \ldots, u_m(x)) \in R_n^m$, and the norm of such a vector is defined as $\|\mathbf{u}\|_\infty = \max_{1\leq i\leq m} \|u_i(x)\|_\infty$. By $R_{n,q}^{k\times m}$ we mean $k \times m$ matrices with entries from $R_{n,q}$.

3.2 Probabilistic distributions

We shall use several probabilistic distributions. A random variable on $\mathbb{Z}_q$ is uniform random if it takes each element of $\mathbb{Z}_q$ with equal probability, namely $1/q$, and a random variable X on $\mathbb{Z}_q^n$ or $R_{n,q}$ is uniform random if each component (or each coefficient) is independent and uniform random on $\mathbb{Z}_q$. For any real number $\tau > 0$, by τ-bounded random variable X on $\mathbb{Z}$, we mean X is random according to some distribution on the integers i with $|i| \leq \tau$, and X never takes any other value. A random variable X on $\mathbb{R}$ is called Gaussian with parameter $\alpha > 0$ if its density function is

$$\rho_\alpha(x) = \frac{1}{\alpha}\exp(-\pi(x/\alpha)^2), \quad x \in \mathbb{R}.$$

A Gaussian random variable with parameter α has expected value 0 and standard deviation $\alpha/\sqrt{2\pi}$. In this work, the analysis of sub-Gaussian random variables will be critical and from [54] we recall several useful properties. A random variable X over $\mathbb{R}$ is called sub-Gaussian with parameter α and we write $X \sim \mathrm{subG}(\alpha^2)$ if $E(X) = 0$ and its moment generating function satisfies

$$E[\exp(tX)] \leq \exp(\alpha^2 t^2/2), \quad \forall\, t \in \mathbb{R}.$$

Lemma 3.1 (sub-Gaussian Properties)**.**

1. *X is sub-Gaussian with parameter α if and only if its tails are dominated by a Gaussian of parameter α, i.e.,*

$$Prob(|X| \geq t) \leq 2\exp(-(t/2\alpha)^2), \quad \textit{for all } t \geq 0.$$

2. *A sum of independent sub-Gaussian random variables on $\mathbb{R}$ is still sub-Gaussian; in particular, [54, Cor1.7] if $X_1, ..., X_n$ are n independent sub-Gaussians of parameter α, $X_i \sim subG(\alpha^2)$, then for any $\mathbf{a} \in \mathbb{R}^n$*

$$Prob\left(\left|\sum_{i=1}^{n} a_i X_i > t)\right| \leq t\right) \leq 2\exp\left(\frac{-t^2}{2\alpha^2||\mathbf{a}||_2^2}\right),$$

or equivalently

$$\sum_{i=1}^{n} a_i X_i \sim subG((\alpha||\mathbf{a}||_2)^2).$$

3. *A max bound over several sub-Gaussians is given by [54, Thm 1.14] as, if $X_1, ..., X_N$ are $X_i \sim subG(\alpha^2)$ (not necessarily independent), then for any $t > 0$*

$$Prob\left(\max_{0\leq i\leq N} X_i > t\right) \leq N\exp(-t^2/2\alpha^2).$$

4. *A τ-bounded random variable with mean 0 is always sub-Gaussian with parameter τ [37].*

In a concurrent work [56] we study sums of sub-Gaussians where their vector of coefficients is not independent of the random variables in the sum. The main results, which will be key in the proof of our error bound Lemma 5.2, are as follows.

Theorem 3.2. *If Z_1 is $subG(t_1^2)$ and for $2 \leq i \leq n$, $(Z_i|Z_1, ..., Z_{i-1})$ is $subG(t_i^2)$ and t_i^2 is free of $Z_1, ..., Z_{i-1}$ and $E[Z_i] = 0$, then $Z_1 + \cdots + Z_n$ is $subG(t_1^2 + t_2^2 + \cdots + t_n^2)$.*

Corollary 3.3. *For the sum*

$$X_1Y_1 + X_2Y_2 + \cdots + X_nY_n$$

where Y_i are iid τ-bounded variables with mean 0 and and where X_i are α-bounded variables with mean 0 but X_i depends on $X_1, Y_1, ..., X_{i-1}, Y_{i-1}$, the total sum is sub-Gaussian of parameter $\sqrt{n}\tau\alpha$.

It is these results that have allowed us to remove the Independence Heuristic that is used in [27] and other FHE schemes. We hope that presenting these results in a clear way in [56] will help them to be used to remove the need for the Independence Heuristic in other schemes as well.

We should note that a sum of independent sub-Gaussian random variables on $\mathbb{Z}_q$ will be nearly uniform random when the number of variables is large enough and the width of the sub-Gaussians are large enough relative to the size of q; see [21].

3.3 LWE and RLWE

LWE problem. Regev (2005 [52, 53]) introduced the learning with errors (LWE) problem over $\mathbb{Z}_q$. Let χ be a probabilistic distribution on $\mathbb{Z}$ that strongly favors values with small absolute value, and let $\mathbf{s} \in \mathbb{Z}_q^n$ be an arbitrary vector (corresponding to a secret key of a user). An LWE sample is of the form $(\mathbf{a}, b)$ where $\mathbf{a} \in \mathbb{Z}_q^n$ is uniform random and

$$b = \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q$$

with $e \in \mathbb{Z}$ being randomly chosen according to the error distribution χ. The LWE problem over $\mathbb{Z}_q$ is to find $\mathbf{s}$ given LWE samples in $\mathbb{Z}_q^n \times \mathbb{Z}_q$ where the number of samples can be as large as one desires, but should be bounded by a polynomial in $n \log(q)$. The decision version of the LWE problem is to distinguish LWE samples from samples with uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

LWE ciphers. Regev [52, 53] also introduced a cryptosystem based on the LWE problem. Let $\mathbf{s} \in \mathbb{Z}_q^n$ be a secret key, $D_q = \lfloor q/4 \rfloor$ (this D_q is different from the one used later) and $1 \leq \tau < D_q/2$. To encrypt a message bit $x \in \{0, 1\}$, pick $\mathbf{a} \in \mathbb{Z}_q^n$ uniform randomly and compute

$$b := \langle \mathbf{s}, \mathbf{a} \rangle + e + xD_q \bmod q,$$

where $e \in [-\tau, \tau]$ is uniform random or truncated Gaussian. Then $(\mathbf{a}, b)$ is a ciphertext for x, denoted as

$$\mathrm{E}_{\mathbf{s}}(x) = (\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q,$$

called an LWE cipher of x. To decrypt a ciphertext $\mathrm{E}_{\mathbf{s}}(x) = (\mathbf{a}, b)$, compute

$$b_1 := b - \langle \mathbf{s}, \mathbf{a} \rangle \bmod q,$$

where $-q/2 < b_1 \leq q/2$, and $x_1 := \lfloor b_1/D_q \rceil$. Then $x = x_1$.

RLWE problem. Lyubashevsky, Peikert and Regev (2010 [45]) introduced the ring learning with error (RLWE) problem in order to get more efficient encryption schemes. Let $s(x) \in R_{n,q}$ be any secret key. An RLWE sample is of the form $(a(x), b(x)) \in R_{n,q}^2$ where $a(x) \in R_{n,q}$ is uniform random and

$$b(x) := s(x)a(x) + e(x) \bmod (x^n + 1, q),$$

where $e(x) \in R_n$ with each coefficient small and random (according to certain distribution). An RLWE sample $\mathbf{v} \in R_{n,q}^2$ is said to have error size τ if

$$\mathbf{v}(-s(x), 1)^t \equiv e(x) \pmod{(x^n + 1, q)}, \tag{1}$$

where $e(x) \in R_n$ and $||e(x)||_\infty \le \tau$. The RLWE problem over $\mathbb{Z}_q$ is to find $s(x)$ given many RLWE samples where each sample is random and independent.

RLWE ciphers. Let $m(x) = \sum_{i=0}^{n-1} m_i x^i$ where $m_i \in \{0, 1\}$, which represents an n-bit message. An RLWE cipher for $m(x)$ with error size τ is of the form

$$\mathrm{RE}_{\mathbf{s}}(m(x)) = \mathbf{v} + m(x)D_q(0, 1) \in R_{n,q}^2 \tag{2}$$

where $\mathbf{v} \in R_{n,q}^2$ is an RLWE sample with error size τ. Suppose $\mathrm{RE}_{\mathbf{s}}(m(x)) = (a(x), b(x))$. Then

$$b(x) - s(x)a(x) \equiv m(x)D_q + e(x) \bmod (x^n + 1, q),$$

where $e(x) \in R_n$ is random with $||e(x)||_\infty \le \tau$. When $\tau < D_q/2$, one can recover $m(x)$ from $b(x) - s(x)a(x)$, after reduced modulo $(x^n + 1, q)$.

3.4 Gadget matrix, flattening, external product and GSW ciphers

In order to perform homomorphic multiplication, Gentry, Sahai, and Waters (2013 [34]) introduced the idea of a gadget matrix so that new ciphertexts from multiplication of ciphertexts remain the same size, while previous methods increase the size of new ciphertexts.

Gadget matrix and Random Flattening Let B and ℓ be positive integers so that $B^\ell \ge q$. Let

$$g = (1, B, \dots, B^{\ell-1}).$$

Every element $a \in \mathbb{Z}_q$ can be represented as

$$a = a_0 + a_1 B + \cdots + a_{\ell-1}B^{\ell-1} = (a_0, a_1, \dots, a_{\ell-1})g^t$$

where $a_i \in \mathbb{Z}$ has small size. For example, if we let $-B/2 < a_i \le B/2$, then $(a_0, a_1, \dots, a_{\ell-1})$ is unique. We will allow $|a_i|$ to be as big as B, which allows us to take a random representation. In our scheme we shall fix $\ell = 2$.

Lemma 3.4. *Given $B^2 \ge Q > B$ and integer $d \in (-\frac{Q}{2}, \frac{Q}{2}]$, there exist unique integers $d_0 \in (-\frac{B}{2}, \frac{B}{2}]$ and $d_1 \in [-\frac{B}{2}, \frac{B}{2}]$ s.t. $d = d_0 + d_1 \cdot B$.*

Proof. Let

$$d_1 = \lfloor d/B \rceil, \ d_0 = d - d_1 \cdot B$$

where $x - \lfloor x \rceil \in (-\frac{1}{2}, \frac{1}{2}]$ for any $x \in \mathbb{R}$. Notice that since $B^2 \ge Q > B$, and $d \in (-\frac{Q}{2}, \frac{Q}{2}]$, we have $d/B \in (-\frac{B}{2}, \frac{B}{2}]$. By the fact that $d/B - \lfloor d/B \rceil \in (-\frac{1}{2}, \frac{1}{2}]$, then we know $\lfloor d/B \rceil \in (-\frac{B+1}{2}, \frac{B+1}{2})$, implying that $d_1 = \lfloor d/B \rceil \in [-\frac{B}{2}, \frac{B}{2}]$ as B is an integer. Moreover, $d_0 = d - d_1 \cdot B = (d/B - \lfloor d/B \rceil) \cdot B \in (-\frac{1}{2}, \frac{1}{2}] \cdot B \in (-\frac{B}{2}, \frac{B}{2}]$ as what we need.

Assume $(d_0', d_1'), (d_0, d_1) \in (-\frac{B}{2}, \frac{B}{2}] \times [-\frac{B}{2}, \frac{B}{2}]$ s.t.

$$d = d_0 + d_1 \cdot B = d_0' + d_1' \cdot B,$$

then

$$d_0 - d_0' = (d_1' - d_1) \cdot B.$$

Since $d_0, d_0' \in (-\frac{B}{2}, \frac{B}{2}]$, we know $d_0 - d_0' \in (-B, B)$. Notice that $B \mid d_0 - d_0'$, actually $d_0 - d_0' = 0$, implying that $d_0 = d_0', d_1 = d_1'$ which proves the uniqueness of d_0, d_1. □ Using Lemma 3.4, we can prove the following random flattening approach.

Lemma 3.5. *(random flattening) For $B^2 \geq Q > B$, let $S_Q = (-\frac{Q}{2}, \frac{Q}{2}]$ denote a residue class mod Q. Given $a \in S_Q$ uniform random, choose $k_0, k_1 \in S_Q$ uniform randomly and compute $i_1 = \lfloor k_1/B \rceil$, $i_0 = k_0 - \lfloor k_0/B \rceil \cdot B$, where $\lfloor x \rceil$ denotes the integer closest to x and $\lfloor x \rceil := k$ if $x = k + \frac{1}{2}$ for some integer k, there exists unique $j_0 \in (-\frac{B}{2}, \frac{B}{2}], j_1 \in [-\frac{B}{2}, \frac{B}{2}]$ s.t. $a + i_0 + i_1 \cdot B = j_0 + j_1 \cdot B$ mod Q, and letting $a_0 = j_0 - i_0$, $a_1 = j_1 - i_1$, we have*

$$a = a_0 + a_1 \cdot B \ \textit{mod}\ Q,$$

where a_0, a_1 are random variables s.t. $|a_0|, |a_1| \leq B$ and $E(a_0) = E(a_1) = 0$.

Proof. Let $a' = a + i_0 + i_1 \cdot B$ mod $Q \in S_Q$, then a' is uniform in S_Q since $a \in S_Q$ is already uniform. By Lemma 3.4, we know that there exists unique $j_0 \in (-\frac{B}{2}, \frac{B}{2}], j_1 \in [-\frac{B}{2}, \frac{B}{2}]$ s.t. $a' = j_0 + j_1 \cdot B$. Moreover, by the proof of Lemma 3.5,

$$j_1 = \lfloor a'/B \rceil,\ j_0 = a' - \lfloor a'/B \rceil \cdot B.$$

Notice that a', k_1 are both uniform in S_Q, we know j_1 and i_1 actually have the same distribution, implying that $i_1 \in [-\frac{B}{2}, \frac{B}{2}]$ and $E(j_1) = E(i_1)$. Similarly, since k_0 is also uniform in S_Q, j_0 and i_0 also have the same distribution, it follows that $i_0 \in (-\frac{B}{2}, \frac{B}{2}]$ and $E(j_0) = E(i_0)$. As a result, we obtain

$$|a_1| = |j_1 - i_1| \leq |j_1| + |i_1| \leq \frac{B}{2} + \frac{B}{2} = B$$

and $E(a_1) = E(j_1) - E(i_1) = 0$. Similarly, we have

$$|a_0| = |j_0 - i_0| \leq |j_0| + |i_0| \leq \frac{B}{2} + \frac{B}{2} = B$$

and $E(a_0) = E(j_0) - E(i_0) = 0$. □

We can extend this random flattening to any list of elements in $\mathbb{Z}_q$ and thus to polynomials.

Lemma 3.6. *Let n be a power of 2, $q = p_1 p_2 \cdots p_t$ be the product of distinct primes such that $2n|(p_i - 1)$ for each i. Every polynomial $a(x) \in R_{n,q}$ can be written as*

$$a(x) = a_0(x) + a_1(x)B = (a_0(x), a_1(x))g^t$$

where $||a_i(x)||_\infty \leq B$ for $0 \leq i \leq \ell - 1$. Furthermore, we can get $a(x) = a_0(x) + a_1(x)B$ with $a_0(x)$ or $a_1(x)$ invertible in $R_{n,q}$.

Proof. Using Lemma 3.5 we can get a random flattening $a(x) = a_0(x) + a_1(x)B$. Then if we assume $q = p_1 p_2 \cdots p_t$ is the product of district primes, we can then check if either $a_0(x)$ or $a_1(x)$ is coprime to $x^m + 1$ modulo p_i for all i. This can be done by checking if they share a common root, by evaluating $a_0(x)$ at all the roots of $x^n + 1$ modulo p_i, i.e. taking an FFT of $a_0(x)$ in R_{n,p_i} and then checking if any component is zero. If it is not coprime, we can simply apply Lemma 3.5 again to get a new one and check if it is coprime. We will argue next why with high probability we should get coprime with few attempts. □

Heuristic Assumption 3.7. *In the proof of Lemma 3.6 we expect to find a coprime $a_0(x)$ or $a_1(x)$ after a very small number of attempts.*

Argument: For any p_j, let $\alpha_0, \alpha_1, ..., \alpha_{n-1}$ be the roots of $x^n + 1$ modulo p_j, which have the form $\alpha_i = \omega^{2i+1}$ $0 \le i < n$ where ω has order $2n$. The FFT of a polynomial $v(x) \in R_{n,q}$ has the form $(v(\alpha_0), v(\alpha_1), ..., v(\alpha_{n-1}))$. If $v(x) = \sum_{i=0}^{n-1} v_i x^i$ is uniform random then the probability that

$$v(\alpha) = v_0 + v_1\alpha + v_2\alpha^2 + \cdots + v_{n-1}\alpha^{n-1} = 0 \pmod{p_j}$$

is $1/p_j$. Thus the probability that any root would evaluate to zero would be less than n/q_j. Over all p_j, $1 \le j \le t$, the probability that any root would evaluate to zero for any p_j would be less than $\left(\frac{n}{p_1} + \frac{n}{p_2} + \cdots + \frac{n}{p_t}\right)$. When we use this in our scheme we will take $q = p_1p_2$ each of size around 2^{40} and $n = 2^{13}$, so this is $\approx 2^{14}/2^{40} = 2^{-26}$. Where the heuristic assumption comes in is that $v(x)$ in our case is not uniform random but rather the outcome of the random flattening process. Since it is random but not uniform random, the probability is harder to specify. □

We now define

$$G = \begin{pmatrix} g^t & 0 \\ 0 & g^t \end{pmatrix}$$

a $(2\ell) \times 2$ matrix, called a gadget matrix. By Lemma 3.6, every $(a(x), b(x)) \in R_{n,q}^2$ can be written as

$$(a(x), b(x)) = \mathbf{u}(x)G \tag{3}$$

where $\mathbf{u}(x) \in R_n^{2\ell}$ is the random flattening of $(a(x), b(x))$ with $||\mathbf{u}(x)||_\infty \le B$. We define

$$(a(x), b(x)) \lhd G^{-1} = \mathbf{u}(x).$$

The reader should be warned that G is not a square matrix, so it has no inverse, here we just use G^{-1} as an operator that acts from the right on $(a(x), b(x))$, a row vector of two polynomials with coefficients in $\mathbb{Z}_q$ (of large size), to get $\mathbf{u}(x) = (a(x), b(x)) \lhd G^{-1}$, a random row vector of 2ℓ polynomials each with coefficients at most B (of small size). This is a nice trick of trading element size for dimension. For example, when $B = 3$ and $\ell = 4$, we have

$$G = \begin{pmatrix} 1 & 3 & 3^2 & 3^3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 3^2 & 3^3 \end{pmatrix}^t,$$

and

$$(5 + 35x, -14) \lhd G^{-1} = (-1 - x, -1, 1 + x, x, 1, 1, 1, -1) \in R_n^8.$$

since $5 = 3^2 - 3 - 1$, $35 = 3^3 + 3^2 - 1$, and $-14 = -3^3 + 3^2 + 3 + 1$. By definition, we have

$$(\mathbf{v} \lhd G^{-1})G = \mathbf{v}, \quad \text{for every } \mathbf{v} \in R_{n,q}^2. \tag{4}$$

External product. For any row vector $\mathbf{v} \in R_{n,q}^2$ and any $A \in R_{n,q}^{2\ell \times 2}$ (which denotes $(2\ell) \times 2$ matrices with entries in $R_{n,q}$), their external product is defined as

$$\mathbf{v} \odot A = (\mathbf{v} \lhd G^{-1})A \in R_{n,q}^2,$$

which is a random vector in $R_{n,q}^2$, since $\mathbf{v} \lhd G^{-1}$ is a random row vector of length 2ℓ and A is an $(2\ell \times 2)$ matrix. This definition can be extended to define the product of any two $(m\ell) \times m$ matrices (to get another $(m\ell) \times m$ matrix), as originally defined by Gentry, Sahai and Waters (2013 [34]). Recently, Chillotti et al (2016 [23, 24]) observed that, for bootstrapping, it is better to use this external product. From the definition, the external product is right distributive, that is, for any two matrices $A, B \in R_{n,q}^{(2\ell)\times 2}$, we have

$$\mathbf{v} \odot (A + B) \equiv \mathbf{v} \odot A + \mathbf{v} \odot B \pmod{(x^n + 1, q)},$$

where all three terms use the same $v \triangleleft G^{-1}$. However, they are not equal if one computes each term independently (unless $v \triangleleft G^{-1}$ is deterministic). Also, it is not left distributive, i.e., for two vectors $\mathbf{v}_1, \mathbf{v}_2 \in R_{n,q}^2$,

$$(\mathbf{v}_1 + \mathbf{v}_2) \odot A \not\equiv \mathbf{v}_1 \odot A + \mathbf{v}_2 \odot A \pmod{(x^n + 1, q)},$$

in general, since the operator G^{-1} is not linear when acting on $\mathbf{v}$.

GSW ciphers. Let $s(x) = \sum_{i=0}^{n-1} s_i x^i$, where $s_i \in \{0, 1\}$, be an n-bit secret key of a user. For any $m(x) \in R_n$ (say with small coefficients), a GSW cipher for $m(x)$ with error size τ is of the form

$$\mathrm{GSW}_{\mathbf{s}}(m(x)) = A + m(x)G \in R_{n,q}^{(2\ell)\times 2} \tag{5}$$

where $A \in R_{n,q}^{2\ell\times 2}$ and each row of A is an RLWE sample (chosen independent randomly) so that

$$A(-s(x), 1)^t \equiv \mathbf{w}(x) \bmod (x^n + 1, q)$$

where $\mathbf{w}(x) \in R_n^{2\ell}$ with $||\mathbf{w}(x)||_\infty \le \tau$. The next lemma is observed by Chillotti et al (2016 [23]), here we make the error bound explicit in our error model.

Lemma 3.8. *Let $m_0, m_1 \in R_n$ be any two polynomials. For any $RE_{\mathbf{s}}(m_0)$ with error polynomial $||w_0(x)||_\infty \le \tau_0$ and any $GSW_{\mathbf{s}}(m_1)$ with error vector $||\mathbf{w}(x)||_\infty \le \tau_1$, we have*

$$RE_{\mathbf{s}}(m_0) \odot GSW_{\mathbf{s}}(m_1) = RE_{\mathbf{s}}(m_0 m_1),$$

and $RE_{\mathbf{s}}(m_0 m_1)$ has error polynomial $\mathbf{h}\mathbf{w}(x)^t + m_1 w_0(x)$ where $\mathbf{h}$ is the random fattening of $RE_{\mathbf{s}}(m_0)$.

Proof. By assumption, we may let

$$\mathrm{RE}_{\mathbf{s}}(m_0) = \mathbf{v} + m_0 D_q(0, 1) \in R_{n,q}^2, \qquad \mathrm{GSW}_{\mathbf{s}}(m_1) = A + m_1 G \in R_{n,q}^{(2\ell)\times 2},$$

where $\mathbf{v} \in R_{n,q}^2$ and $A \in R_{n,q}^{2\ell\times 2}$ satisfying, modulo $(x^n + 1, q)$,

$$\mathbf{v}(-s(x), 1)^t \equiv w_0(x), \qquad A(-s(x), 1)^t \equiv \mathbf{w}(x)^t, \tag{6}$$

and $w_0(x) \in R_n$ and $\mathbf{w}(x) = (w_1(x), \ldots, w_{2\ell}(x)) \in R_n^{2\ell}$ with $||w_0(x)||_\infty \le \tau_0$ and $||w_i(x)||_\infty \le \tau_1$ for $1 \le i \le 2\ell$. Let $\mathbf{h} = \mathrm{RE}_{\mathbf{s}}(m_0(x)) \triangleleft G^{-1} = (h_1, \ldots, h_{2\ell}) \in R_n^{2\ell}$, with $||\mathbf{h}||_\infty \le B$. Computing modulo $(x^n + 1, q)$, we have

$$\begin{aligned}
\mathrm{RE}_{\mathbf{s}}(m_0(x)) \odot \mathrm{GSW}_{\mathbf{s}}(m_1(x)) &\equiv \mathbf{h}(A + m_1 G) = \mathbf{h}A + m_1 \mathbf{h} G \\
&\equiv \mathbf{h}A + m_1([\mathbf{v} + m_0 D_q(0, 1)] \triangleleft G^{-1})G \\
&\equiv \mathbf{h}A + m_1[\mathbf{v} + m_0 D_q(0, 1)] \\
&\equiv (\mathbf{h}A + m_1 \mathbf{v}) + m_0 m_1 D_q(0, 1) \in R_{n,q}^2,
\end{aligned}$$

where, in the second last equation, we used the property of G^{-1} from (4). The error polynomial is, using (6), $(\mathbf{h}A + m_1\mathbf{v})(-s(x), 1)^t \equiv \mathbf{h}\,\mathbf{w}(x)^t + m_1 w_0(x)$. □

4 Encryption schemes

We present two encryption schemes based on the RLWE problem [45]: one using private keys and the other using public keys. We use rounding and modulus reduction. We note that the technique of modulus reduction has been used in [16, 14, 15] and rounding was introduced in LWR in (2012, [9]). Brakerski [13] suggested in a comment to also use rounding to reduce ciphertext sizes in FHE schemes, but before [32] such techniques had not led to FHE schemes with such small expansion factors. We shall encrypt a message polynomial $m(x) = \sum_{i=0}^{n-1} m_i x^i$, where $m_i \in [0, 2^k)$. Using all these techniques, and by carefully choosing error distributions, we achieve a cipher expansion of $1 + \frac{6}{k}$ for encryption with private keys and $2 + \frac{12}{k} + \frac{\log_2(n)}{2k}$ for encryption with public keys. Let r be a power of 2, $D_r = r/2^{k+2}$ and $D_q = \lfloor q/2^{k+2} \rfloor$.
Secret key. Randomly pick $\mathbf{s} = (s_0, s_1, \dots, s_{n-1}) \in \{0,1\}^n$ of hamming weight at most $n/8$, and let $s(x) = \sum_{i=0}^{n-1} s_i x^i$.
Public key. A corresponding public key in $R_{n,q}^2$ is with respect to a larger modulus $q \geq 2^7 rn$ and is generated as $pk = (k_0(x), k_1(x))$ where $k_0(x) \in R_{n,q}$ is chosen uniform randomly and

$$k_1(x) := k_0(x)s(x) + e(x) \text{ mod } (x^n + 1, q)$$

with $e(x) \in R_n$ bounded uniform random st $||e(x)||_\infty < D_q/(512n)$.
Pseudo-random number generator P. We also need a pseudo-random number generator in order to reduce ciphertext size under encryption with private keys. Suppose P is a function that can expand any n-bit sequence $u \in \{0,1\}^n$ (deterministically) into a sequence of 0's and 1's of length $n\lceil \log_2(r) \rceil$, denoted by $P(u)$. The sequence $P(u)$ can be uniquely converted into a polynomial in $R_{n,r}$, denoted by $P(u,x)$. For example, one can use SHAKE-128 [51], or the lightweight generator [7]. However, the function P needs not to have a strong cryptographic property, but only needs to be statistically uniform, that is, when $u \in \{0,1\}^n$ is uniform random, $P(u,x)$ should be nearly uniform random in $R_{n,r}$. The security of our encryption scheme depends on the RLWE problem in $R_{n,r}$ and that $P(u,x)$ is nearly uniform random in $R_{n,r}$.

4.1 Private-key Encryption

To encrypt a message $m(x)$ under secret key $\mathbf{s}$, our scheme is in Figure 1.

Lemma 4.1. *Let $(a(x), b(x)) \in R_{n,r}^2$ be as computed in Figure 1. Then there exists $w_3(x) \in R_n$ with $||w_3(x)||_\infty < \frac{D_r}{4}$ so that*

$$2^{t-k-4}b(x) - s(x)a(x) \equiv w_3(x) + m(x)D_r \textit{ mod } (x^n + 1, r).$$

In particular, where $r = 2^{k+6}\sqrt{n}$, $(u, \boldsymbol{v})$ returned in Step 4 has $(k+6)n$ bits and represents an RLWE cipher $RE_{\mathbf{s}}(m(x))$ with error size $< 4\sqrt{n}$.

Proof. By Step 3, since the coefficients of $b_1(x)$ are between 0 and $r-1$, we have $b_1(x) = 2^{t-k-4}b(x) + b_0(x)$ for some $b_0 \in R_n$ with $||b_0(x)||_\infty < 2^{t-k-4}$. By Step 2, we have $2^{t-k-4}b(x) - s(x)a(x) \equiv -b_0(x) + w(x) + m(x)D_r \text{ mod } (x^n + 1, r)$. Note that $r = 2^{t+1}$ and $D_r = 2^{t-k-1}$, so

$$||-b_0(x) + w(x)||_\infty \leq ||b_0(x)||_\infty + ||w(x)||_\infty < 2^{t-k-4} + \frac{D_r}{8} = \frac{D_r}{4}.$$

Therefore, the lemma holds with $w_3(x) = w(x) - b_0(x)$. □

```
            **Private-key Encryption** : $\mathrm{RE}_{\mathbf{s}}(m(x))$
Input:      $s(x) = \sum_{i=0}^{n-1} s_i x^i$ where $s_i \in \{0,1\}$, an $n$-bit secret key,
            $m(x) = \sum_{i=0}^{n-1} m_i x^i$, where $m_i \in [0, 2^k)$, a $kn$-bit message,
            $t := \lceil \log_2(r) \rceil - 1$, hence $2^t < r \le 2^{t+1}$,
            $P : \{0,1\}^n \to \{0,1\}^{n(t+1)}$, a pseudo-random number generator.
Output:     $(u, \mathbf{v}) \in \{0,1\}^n \times \{0,1\}^{(k+5)n}$
Step 1.     Pick $u \in \{0,1\}^n$ uniform randomly, and compute
               $a(x) := P(u, x) \in R_{n,r}$.
Step 2.     Pick $w(x) \in R_n$ uniform randomly with $\|w(x)\|_\infty \le D_r/8$, and
            compute
            $b_1(x) := a(x)s(x) + w(x) + m(x)D_r \bmod (x^n + 1, r)$
            (so that each coefficient of $b_1(x)$ is between 0 and $r - 1$).
Step 3.     Taking the highest $k + 5$ bits for each coefficient of $b_1(x)$:
                 $b(x) := \lfloor b_1(x)/2^{t-k-4} \rfloor$.
            Let $\mathbf{v} \in (\{0,1\}^{k+5})^n$ denote the bit representation of $b(x)$.
Step 4.     Return $(u, \mathbf{v})$.
```

Figure 1:

4.2 Public-key Encryption

The scheme is presented in Figure 2.

Lemma 4.2. *Suppose $n \ge 512$, $r = 2^{t+1} \ge 2^{k+6}\sqrt{n}$ and $q \ge 2^7 rn$. Let $(a(x), b(x)) = RE_{pk}(m(x)) \in R^2_{n,r}$. Then with probability $\ge 1 - n \cdot 2^{-190}$, we have*

$$2^{t-k-5} b(x) - s(x)a(x) \equiv w_3(x) + m(x)D_r \ \mathit{mod}\ (x^n + 1, r).$$

for some $w_3(x) \in R_n$ with $||w_3(x)||_\infty < \frac{D_r}{4}$. In particular, when $r = 2^{k+6}\sqrt{n}$, each ciphertext $RE_{pk}(m(x))$ has $n(2k + 12 + \frac{1}{2}log_2(n))$ bits and the error, i.e. each coefficient of $w_3(x)$ is random in $(-4\sqrt{n}, 4\sqrt{n})$.

Proof. By Step 3, we have $a(x) = ra_1(x)/q + v_1(x)$, and $2^{t-k-5}b(x) = rb_1(x)/q + v_0(x) + m(x)D_r$, where $v_i(x) \in \mathbb{R}[x]$ with degree $< n$ for $i = 0$ and 1, $||v_1(x)||_\infty \le 1/2$ and $||v_0(x)||_\infty < 2^{t-k-5}$. By $k_1(x) = k_0(x)s(x) + e(x) \bmod (x^n + 1, q)$ and $a_1(x)$, $b_1(x)$, there exist polynomials $h_1(x), h_2(x) \in \mathbb{Z}[x]$ so that

$$b_1(x) - s(x)a_1(x) = u(x)e(x) + w_2(x) - s(x)w_1(x) + h_1(x)(x^n + 1) + qh_2(x).$$

Let $w(x) = u(x)e(x) + w_2(x) - s(x)w_1(x)$. Then

$$\begin{aligned} 2^{t-k-5}b(x) - s(x)a(x) &= \frac{r}{q}(b_1(x) - s(x)a_1(x)) + v_0(x) - s(x)v_1(x) + m(x)D_r \\ &= \frac{r}{q}w(x) + v_0(x) - s(x)v_1(x) + m(x)D_r \\ &+ \frac{r}{q}h_1(x)(x^n + 1) + rh_2(x) \\ &\equiv w_3(x) + m(x)D_r \bmod (x^n + 1, r) \end{aligned}$$

where $w_3(x) = \frac{r}{q}w(x) + v_0(x) - s(x)v_1(x)$. We need to estimate the coefficient size of other terms in $w_3(x)$(when reduced modulo $x^n + 1$). Since $u(x) = \sum_{i=0}^{n-1} u_i$ where $u_i \in \{-1, 0, 1\}$,

	Public-key Encryption : $\mathrm{RE}_{pk}(m(x))$
Input:	$pk = (k_0(x), k_1(x)) \in R_{n,q}^2,\ \ t := \lceil \log_2(r) \rceil - 1$, hence $2^t < r \leq 2^{t+1}$
	$m(x) = \sum_{i=0}^{n-1} m_i x^i$: a kn-bit message where each $m_i \in [0, 2^k)$,
Output:	$(a(x), b(x)) \in R_{n,r}^2$
Step 1.	Pick $u(x) \in R_n$ with each coefficient random in $\{-1, 0, 1\}$,
	Pick $w_1(x) \in R_n$ randomly with $\lvert\lvert w_1(x) \rvert\rvert_\infty \leq D_q/(64n)$,
	Pick $w_2(x) \in R_n$ randomly with $\lvert\lvert w_2(x) \rvert\rvert_\infty \leq D_q/256$.
Step 2.	Compute:
	$a_1(x) := k_0(x)u(x) + w_1(x) \mod (x^n + 1, q)$,
	$b_1(x) := k_1(x)u(x) + w_2(x) \mod (x^n + 1, q)$.
	(Both $a_1(x)$ and $b_1(x)$ have coefficients in $[0, q-1]$.)
Step 3.	Modulus reduction and rounding:
	$a(x) := \left\lfloor \frac{r}{q} a_1(x) \right\rceil$, $b(x) := \left\lfloor \frac{1}{2^{t-k-5}} (\frac{r}{q} b_1(x) + m(x) D_r) \right\rceil$.
	(Each coefficient of $b(x)$ is in $[0, 2^{k+6} - 1]$, hence has $k+6$ bits.)
Step 4.	Return $(a(x), b(x))$.

Figure 2:

we have

$$||u(x)e(x)||_\infty \leq \sum_{i=0}^{n-1} ||u_i x^i e(x)||_\infty = \sum_{i=0}^{n-1} ||e(x)||_\infty = n||e(x)||_\infty \leq n \frac{D_q}{512n} = \frac{D_q}{512}.$$

Similarly, since $s_i \in \{0, 1\}$ and $\mathrm{Ham}(\mathbf{s}) \leq \frac{1}{8}n$, we obtain $||s(x)w_1(x)||_\infty \leq \frac{1}{8}n||w_1(x)||_\infty \leq \frac{1}{8}n \cdot \frac{D_q}{64n} = \frac{D_q}{512}$. Therefore,

$$||w(x)||_\infty \leq ||u(x)e(x)||_\infty + ||w_2(x)||_\infty + ||s(x)w_1(x)||_\infty \leq \frac{D_q}{512} + \frac{D_q}{256} + \frac{D_q}{512} = \frac{D_q}{128}.$$

Expanding terms of $s(x)v_1(x)$, we obtain

$$\begin{aligned} s(x)v_1(x) = \sum_{i=0}^{n-1} s_i x^i \sum_{l=0}^{n-1} v_{1,l} x^l &= \sum_{i=0}^{n-1} \sum_{l=0}^{n-1} s_i v_{1,l} x^{i+l} \\ &= \sum_{k=0}^{n-1} (\sum_{i=0}^{k} s_i v_{1,k-i} - \sum_{i=k+1}^{n-1} s_i v_{1,k+n-i}) \cdot x^k \\ &= \sum_{k=0}^{n-1} \sum_{i=0}^{n-1} c_{k,i} s_i v_{1,(k-i) \mathrm{mod\ n}} \cdot x^k \\ &= \sum_{k=0}^{n-1} \sum_{i=0}^{n-1} s'_{k,i} v_{1,j} \cdot x^k \pmod{x^n + 1, q}. \end{aligned}$$

where $s'_{k,i} = c_{k,i} s_i$ for $c_{k,i} \in \{-1, 1\}$ and $j = (k-i) \mathrm{mod}\ n$ for some $v_{1,j} \in R$ with $|v_{1,j}| \leq \frac{1}{2}$. Thus $||\mathbf{s}'_k|| = ||\mathbf{s}|| \leq \frac{1}{8}n$ for $0 \leq k \leq n-1$. Moreover, we know that $E(v_{1,j}) = 0$ for $0 \leq j \leq n-1$ and all $v_{1,j}$ are independent, implying that $v_{1,j} \sim subG(\frac{1}{4})$. By Corollary 1.7

in [54], for $\sum_{i=0}^{n-1} s'_{k,i}v_{1,j}$ with bound $\frac{23}{\sqrt{2}}||\mathbf{s}'_k||_2 \leq \frac{23}{8}\sqrt{n}$, we obtain

$$\begin{aligned}P(|\sum_{i=0}^{n-1} s'_{k,i}v_{1,j}| > \frac{23}{8}\sqrt{n}) &\leq P(|\sum_{i=0}^{n-1} s'_{k,i}v_{1,j}| > \frac{23}{\sqrt{2}}||\mathbf{s}'_k||_2) \\ &< 2 \cdot e^{-132.5} < 2^{-190}.\end{aligned}$$

Since $||s(x)v_1(x)||_\infty = \max_k |\sum_{i=0}^{n-1} s'_{k,i}v_{1,j}|$, we know

$$\begin{aligned}P(\max_k |\sum_{i=0}^{n-1} s'_{k,i}v_{1,j}| \leq \frac{23}{8}\sqrt{n}) &= 1 - P(\exists k', s.t. |\sum_{i=0}^{n-1} s'_{k',i}v_{1,j}| > \frac{23}{8}\sqrt{n}) \\ &\geq 1 - n \cdot 2^{-190}.\end{aligned}$$

Thus we know with probability $\geq 1 - n \cdot 2^{-190}$,

$$||v_0(x) - s(x)v_1(x)||_\infty \leq ||v_0(x)||_\infty + ||s(x)v_1(x)||_\infty \leq 2^{t-k-5} + \frac{23}{8}\sqrt{n}.$$

Since $n \geq 512$, it follows that with probability $\geq 1 - n \cdot 2^{-190}$,

$$\begin{aligned}||w_3(x)||_\infty &\leq \frac{r}{q}||w(x)||_\infty + ||v_0(x) - s(x)v_1(x)||_\infty \\ &< \frac{r}{128q}D_q + 2^{t-k-5} + \frac{23}{8}\sqrt{n} \\ &< \frac{D_r}{128} + \frac{D_r}{2^4} + \frac{23}{8}\sqrt{n} \\ &= \frac{D_r}{128} + \frac{D_r}{2^4} + \frac{23}{8} \cdot \frac{D_r}{2^4} = \frac{D_r}{4}\end{aligned}$$

where the last inequality is from $r \geq 2^{k+6}\sqrt{n}$ and $D_r = r/2^{k+2}$.

When $r = 2^{k+6}\sqrt{n}$, we know that $\frac{D_r}{4} = \frac{r}{2^{k+4}} = \frac{2^{k+6}}{2^{k+4}}\sqrt{n} = 4\sqrt{n}$, which completes the proof. □

4.3 Decryption, ciphertext expansion, and unpacking

Decryption. To decrypt a ciphertext $(a(x), b(x))$ from $\mathrm{RE}_{\mathbf{s}}(m(x))$ or $\mathrm{RE}_{pk}(m(x))$, the user computes

$b_1(x) := 2^{t-k-4}b(x) - s(x)a(x) \bmod (x^n + 1, r)$, or $b_1(x) := 2^{t-k-5}b(x) - s(x)a(x) \bmod (x^n + 1, r)$,

and $m_1(x) = \lfloor b_1(x)/D_r \rceil$. Then $m_1(x) = m(x)$, the reason is that $b_1(x) \equiv w(x) + m(x)D_r \bmod (x^n + 1, r)$ for some $w(x) \in R_n$ with $||w(x)||_\infty < D_r/4$.

4.3.1 Ciphertext expansion under private key

The ciphertext size under private key encryption is $n(k + 6)$ bits, and encrypts kn message bits. Thus the ciphertext expansion ratio is

$$\frac{n(k+6)}{kn} = 1 + \frac{6}{k}.$$

Thus asymtotically as $k \to \infty$ we get the optimal ciphertext expansion of 1. For the values of $k = 2, 3, 4, 5$ that we suggest for practical use in this scheme we, we get expansion ratios of 4, 3, 2.5, 2.2 respectively as shown in Figure 13.

4.3.2 Ciphertext expansion under public key

As proved in Lemma 4.2 the expansion under public keys when $r = 2^{k+6}\sqrt{n}$, is

$$\frac{n(2k + 12 + \frac{1}{2}\log_2(n))}{kn} = 2 + \frac{12}{k} + \frac{\log_2(n)}{2k}.$$

For $n = 2^{12}$ and the values of $k = 2, 3, 4, 5$ that we suggest for practical use in this scheme we, we get expansion ratios of 11, 8, 6.5, 5.6 respectively as shown in Figure 13.

Unpacking. Given an RLWE cipher $\mathrm{RE}_{\mathbf{s}}(m(x)) = \mathbf{c}$ for $m(x) = \sum_{i=0}^{n-1} m_i x^i$ we want to extract LWE ciphers for each coefficient in the message polynomial. Suppose $\mathbf{c}$ is from private-key encryption. Then $\mathbf{c}$ is of the form $\mathbf{c} = (u, v)$ where $u \in \{0,1\}^n$ and $v \in \left(\{0,1\}^5\right)^n$. Apply the pseudo random number generator P to u to get a polynomial $a(x) = P(u, x) \in R_{n,r}$, and convert v into a polynomial $b(x) \in R_{n,r}$. By Lemma 4.1, we have

$$2^{t-k-4}b(x) \equiv a(x)s(x) + \left(\sum_{i=0}^{n-1} m_{k,i}x^i\right) D_r + w(x) \bmod (x^n + 1, r),$$

where $\|w(x)\|_\infty < D_r/4$. In general, for any $a(x) = a_0 + a_1x + \cdots + a_{m-1}x^{m-1} \in R_{m,q}$ we define

$$\mathrm{Extract}(a(x), i) = (a_i, a_{i-1}, \ldots, a_0, -a_{m-1}, -a_{m-2}, \ldots, -a_{m-(n-1-i)}) \in \mathbb{Z}_q^n.$$

This is used to get the coefficients of $a(x)s(x) \bmod (x^m + 1)$. Note that, modulo $x^m + 1$,

$$\begin{aligned} a(x)s(x) = \sum_{k=0}^{n-1}\sum_{j=0}^{m-1} s_k a_j x^{k+j} \equiv \sum_{i=0}^{m-1} &[(s_0a_i + s_1a_{i-1} + \cdots + s_ia_0) - (s_{i+1}a_{m-1} \\ &+ s_{i+2}a_{m-2} + \cdots + s_{n-1}a_{m-(n-1-i)})]x^i. \end{aligned}$$

Hence, for $0 \le i \le n$, the i-th coefficient of $a(x)s(x) \bmod (x^m + 1)$ is equal to the inner product of $\mathbf{s}$ with $\mathrm{Extract}(a(x), i)$. It follows that for $0 \le i \le n-1$, an LWE cipher for m_i is

$$E_{\mathbf{s}}(m_i) = (\mathrm{Extract}(a(x), i), 2^{t-k-4}b_i) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$$

with error size $< D_r/4$, where b_i is the coefficient of x^i in $b(x)$.

Next suppose $\mathbf{c}_k$ is from public-key encryption. By Lemma 4.2, $\mathbf{c}$ is of the form $(a(x), b(x)) \in R_{n,r}^2$ so that

$$2^{t-k-5}b(x) \equiv a(x)s(x) + \left(\sum_{i=0}^{n-1} m_i x^i\right) D_r + w(x) \bmod (x^n + 1, r),$$

where $\|w(x)\|_\infty < D_r/4$. Hence, for $0 \le i \le n-1$, an LWE cipher for m_i is

$$E_{\mathbf{s}}(m_i) = (\mathrm{Extract}(a(x), i), 2^{t-k-5}b_i) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$$

with error size $< D_r/4$, where b_i is the coefficient of x^i in $b(x)$.

5 Multi-bit Homomorphic operations

In the previous section we saw how data stored efficiently in RLWE ciphers can be unpacked to LWE ciphers, and in this section we will show how homomorphic operations can be done on those LWE ciphers. We follow the approach in Ducas and Micciancio (2015 [29]) and

Chillotti et al. (2016 [23]), however, we do not need to perform key switch as they do. We incorporate a technique of Biasse and Ruiz (2015 [11]) which is further developed by Carpov, Izabachène, and Mollimard (2018 [17]) for doing arbitrary function lookups.

To give a high level description, we take LWE ciphers in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size bounded by $4\sqrt{n}$, which come from the unpacking of RLWE ciphers. The error bound of $4\sqrt{n}$ is very large with respect to r so only one homomorphic addition or subtraction can be done at this step. The ciphertexts are then homomorphically lifted to $R_{m,Q} = \mathbb{Z}[x]/(x^m+1, Q)$ by being embedded in the exponents of RLWE ciphertexts modulo a much larger Q and mixed with the bootstrapping key in a way that homomorphically decrypts them. Further arbitrary functions can then be applied to produce the desired arithmetic operation, and finally the ciphertexts are mapped back down to $\mathbb{Z}_r^n \times \mathbb{Z}_r$ by a modulus reduction. The resulting ciphertexts still have error bounded by the same starting bound of $4\sqrt{n}$ with high probability. Thus, they can continue to be operated on any practical number of times. An overview diagram of scheme is shown in Figure 3.

$$\begin{tikzcd}
 & R^2_{m,Q} \arrow[r, "function"] & R^2_{m,Q} \arrow[r, "extract"] & \mathbb{Z}_Q^n \times \mathbb{Z}_Q \arrow[dll, "mod\ reduction"] \\
R^2_{n,r} \arrow[r, leftrightarrow, "unpack", "pack"'] & \mathbb{Z}_r^n \times \mathbb{Z}_r \arrow[u, "HomLift"] & &
\end{tikzcd}$$

Figure 3: Boostrapping operation, $R_{n,r} := \frac{\mathbb{Z}[x]}{(x^n+1,r)}$.

5.1 Homomorphic Lifting

Suppose we are given LWE ciphers $\mathbf{E}_s(x_i) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ for $x_i \in \{0,..,2^k-1\}$ with error size $< D_r/4$. Let $\mathbf{s} = (s_0, \ldots, s_{n-1}) \in \{0,1\}^n$, representing an n-bit secret key of a user. Suppose $r \geq 2^{k+6}\sqrt{n}$ is a power of 2, Q is much bigger than r (to be determined later) and

$$m = r/2, \quad D_r = \lfloor r/2^{k+2} \rfloor, \quad D_Q = \lfloor Q/2^{k+2} \rfloor.$$

Also take $B^\ell \geq Q > B$ (we shall take $\ell = 2$ in this paper). We shall work in the rings

$$R_m = \mathbb{Z}[x]/(x^m+1), \quad R_{m,Q} = \mathbb{Z}[x]/(x^m+1, Q).$$

Define a bootstrapping key to be $bk = (C_0, \ldots, C_{n-1})$ where

$$C_i = \mathrm{GSW}_{\mathbf{s}}(s_i) = A_i + s_i G \in R_{m,Q}^{(2\ell)\times 2}, \quad 0 \leq i \leq n-1, \tag{7}$$

where $A_i \in R_{m,Q}^{(2\ell)\times 2}$ is a GSW sample (chosen randomly and independently by the owner of $\mathbf{s}$) with certain error size τ_{bk} (to be determined later). Such a bootstrapping key for the user with the secret key $\mathbf{s}$ is made public and can be used by anyone else to compute homomorphic operations of ciphertexts.

Suppose $\mathrm{E}_{\mathbf{s}}(x_1) = (\mathbf{a}_1, b_1)$ and $\mathrm{E}_{\mathbf{s}}(x_2) = (\mathbf{a}_2, b_2)$ where $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{Z}_r^n$ and

$$b_i \equiv \langle \mathbf{s}, \mathbf{a}_i \rangle + x_i D_r + e_i \pmod{r},$$

for some $e_i \in \mathbb{Z}$ with $|e_i| < D_r/4$ for $i = 1, 2$. We note that

$$b_1 + b_2 \equiv \langle \mathbf{s}, \mathbf{a}_1 + \mathbf{a}_2 \rangle + (x_1 + x_2) D_r + e_1 + e_2 \pmod{r}, \tag{8}$$

$$b_1 - b_2 \equiv \langle \mathbf{s}, \mathbf{a}_1 - \mathbf{a}_2 \rangle + (x_1 - x_2) D_r + e_1 - e_2 \pmod{r}. \tag{9}$$

We let $\mathbf{u} \in \mathbb{Z}_r^n \times \mathbb{Z}_r$, y, e be defined any one of the following three ways,

(i) $(u_0, \dots, u_{n-1}) = \mathbf{a}_1 + \mathbf{a}_2 \in \mathbb{Z}_r^n$, $u_n = b_1 + b_2 \in \mathbb{Z}_r$, $y = x_1 + x_2$, and $e = e_1 + e_2 \in \mathbb{Z}$,

(ii) $(u_0, \dots, u_{n-1}) = \mathbf{a}_1 - \mathbf{a}_2 \in \mathbb{Z}_r^n$, $u_n = b_1 - b_2 \in \mathbb{Z}_r$, $y = x_1 - x_2$, and $e = e_1 - e_2 \in \mathbb{Z}$,

(iii) $(u_0, \dots, u_{n-1}) = \mathbf{a}_1 \in \mathbb{Z}_r^n$, $u_n = b_1 \in \mathbb{Z}_r$, $y = x_1$, and $e = e_1 \in \mathbb{Z}$.

In any case, $|e| < D_r/2$ and the equations (8) and (9) become $u_n \equiv \sum_{i=0}^{n-1} s_i u_i + yD_r + e \pmod{r}$. Let $w = u_n - \sum_{i=0}^{n-1} s_i u_i$ and the equation becomes

$$w \equiv yD_r + e \pmod{r}. \tag{10}$$

Now that the error has grown to be $< D_r/2$ no more additions can be made. This is because the modulus r is small compared to the size of the error. We will use our Homomorphic Lifting operation to lift this ciphertext to a larger modulus Q where more additions can be performed. Following Ducas and Micciancio (2015 [29]), we use the group homomorphism from the additive subgroup $(\mathbb{Z}_r, +)$ to the following multiplicative group of $R_{m,Q} = \mathbb{Z}[x]/(x^m + 1, Q)$:

$$\langle x \rangle = \{x^i : 0 \le i \le r-1\} \equiv \{1, x, \dots, x^{m-1}, -1, -x, \dots, -x^{m-1}\},$$

mapping $i \in \mathbb{Z}_r$ to $x^i \in R_{m,Q}$. For any subset $T \subseteq \mathbb{Z}_r$, let

$$t(x) := \sum_{i \in T} x^i \in R_{m,Q}.$$

For example, if $r = 20$, $m = 10$ and $T = \{1, 2, -4, 17\}$, then

$$t(x) = x + x^2 + x^{-4} + x^{17} \equiv x + x^2 - x^6 - x^7 (\text{mod } x^m + 1).$$

For this $t(x)$, its coefficient at x^2 is 1, its coefficient at $x^{m+2} = x^{12}$ is -1 (since $x^2 \equiv -x^{12}$), and its coefficient at x^3 is 0 since none of 3 and $m+3$ is in T. Also note that, if $T = \{w, w+m\}$, then $t(x) = x^w + x^{w+m} \equiv x^w + (-1) \cdot x^w \equiv 0 \pmod{x^m + 1}$. Hence we should avoid using any subset T that contains w and $m + w$ for some w. We let $T := \{i \in \mathbb{Z} : |i| < D_r/2\}$. Then by (10) we have $x^{-w} \equiv x^{-yD_r} x^{-e} \pmod{x^m + 1}$ and multiplying by $t(x)$ which we call the *error encoding polynomial* we get

$$t(x)x^{-w} \equiv t(x)x^{-e}x^{-yDr} \pmod{x^m + 1}. \tag{11}$$

Now we will show how $t(x)x^{-w}$ can be computed under encryption using the boostrapping key, $bk = (C_0, \dots, C_{n-1})$ from equation (7). Note that, for any $z \in \{0, 1\}$ and $u \in \mathbb{Z}$, we have the identity

$$x^{zu} = 1 + (x^u - 1)z. \tag{12}$$

Let $C = \mathrm{GSW}_{\mathbf{s}}(z) \in R_{m,Q}^{(2\ell)\times 2}$ be any GSW cipher. One can map $zu \in \mathbb{Z}_r$ to x^{zu}, then to a GSW cipher: $G + (x^u - 1)C \in R_{m,Q}^{(2\ell)\times 2}$. We describe the Homomorphic Lifting (HomLift) operation in Figure 4. The operation is a random mapping

$$\mathrm{HomLift} : (\mathbb{Z}_r^n \times \mathbb{Z}_r) \times \left\{ R_{m,Q}^{(2\ell)\times 2} \right\}^n \to R_{m,Q}^2$$

$$(\mathbf{u}, bk) \mapsto \mathrm{RE}_{\mathbf{s}}(t(x)x^{-u_n + \sum_{k=0}^{u_i s_i}}).$$

In Step 2, A is updated n times and we can prove that for the final $A = (A_0, A_1)$, A_0 will have uniform random and independent coefficients.

```
Homomorphic Lifting Algorithm : $\mathrm{HomLift}_{bk}(\mathbf{u})$
Input:    $bk = (C_0, \ldots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$: bootstrapping key,
          $\mathbf{u} = (u_0, \ldots, u_{n-1}, u_n) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$
Output:   $\mathrm{RE}_{\mathbf{s}}(t(x)x^{-u_n + \sum_{k=0}^{n-1} u_i s_i}) \in R_{m,Q}^2$
Step 1.   Initialization:
              $t(x) := \sum_{i\in T} x^i$ where $T := \{i \in \mathbb{Z} : -D_r < i < D_r\}$,
              $A := (0, t(x)x^{-u_n} D_Q) \in R_{m,Q}^2$.
Step 2.   For $k$ from 0 to $n-1$ do (randomness involved here)
              $A := A \odot (G + (x^{u_k} - 1)C_k)$.
Step 3.   Return $A$.
```

Figure 4: Homomorphic Lifting Operations

Lemma 5.1. *Taking $m = 2^t$ with $t > 0$, in Step 2 of the Homomorphic Lifting Algorithm, for any initial $A = (A_0, A_1)$ in $R_{m,Q}^2$, we can guarantee that for the final $A' = (A_0', A_1')$, A_0' will have uniform random and independent coefficients.*

Proof. According to Step 2 of the Homomorphic Lifting Algorithm, for $i = 0, 1, \cdots, n-1$, $A = A \odot (G + (x^{u_i} - 1) \cdot C_i)$ where $u_i \in \mathbb{Z}_r$ and

$$C_i = \begin{pmatrix} a_{1i}(x) & a_{1i}(x)s(x) + e_{1i}(x) \\ a_{2i}(x) & a_{2i}(x)s(x) + e_{2i}(x) \\ a_{3i}(x) & a_{3i}(x)s(x) + e_{3i}(x) \\ a_{4i}(x) & a_{4i}(x)s(x) + e_{4i}(x) \end{pmatrix} + s_i G \bmod Q.$$

By the definition of the external product $\odot$ and random flattening, we know that

$$A \odot M = (A_0, A_1) \odot M = (A_{0,L}, A_{0,H}, A_{1,L}, A_{1,H}) \cdot M$$

where $A_0 = A_{0,L} + A_{0,H} \cdot B$ and $A_1 = A_{1,L} + A_{1,H} \cdot B$ in $R_{m,Q}$. Plugging in the form of $G + (x^{u_i} - 1) \cdot C_i$ and denoting the new A by $A' = (A_0', A_1')$, we obtain

$$\begin{aligned} A_0' &= A_{0,L} \cdot ((x^{u_i} - 1)(a_{1i}(x) + s_i) + 1) + A_{0,H} \cdot ((x^{u_i} - 1)(a_{2i}(x) + s_i B) + B) \\ &\quad + A_{1,L} \cdot (x^{u_i} - 1)a_{3i}(x) + A_{1,H} \cdot (x_i^u - 1)a_{4i}(x) \\ &= (x^{u_i} - 1)\left[A_{0,L}a_{1,i}(x) + A_{0,H}a_{2,i}(x) + A_{1,L}a_{3,i} + A_{1,H}a_{4,i}\right] + \gamma \end{aligned}$$

where $\gamma = (x^{u_i} - 1)s_i(A_{0,L} + A_{0,H}B) + (A_{0,L} + A_{0,H}B)$.

If $u_i = 0$, we know that $x^{u_i} - 1 = 0$, implying that $A_0' = A_{0,L} + A_{0,H} \cdot B = A_0$ and $A_1' = A_{1,L} + A_{1,H} \cdot B = A_1$. Thus A remains unchanged during this update.

Claim: If $u_i \neq 0$, we claim that $x^{u_i} - 1$ is invertible in $R_{m,Q}$.

To show this claim, we show $x^{u_i} - 1$ and $x^m + 1$ do not share any common roots. For $x^m + 1$, $x^m \equiv -1 \pmod{Q}$ thus all roots have order exactly $2m$. On the other hand, $x^{u_i} \equiv 1 \pmod{Q}$ so all the roots of $x^{u_i} - 1$ have order dividing u_i and $u_i < r = 2m$, thus they have no common roots. This proves the claim.

We can apply Lemma 3.6 to the initial $A = (A_0, A_1)$ to obtain that at least one of $A_{i,L}$ and $A_{i,H}$, $i = 0, 1$ is invertible in $R_{m,Q}$. WLOG, we assume $A_{0,L}$ is invertible; thus

$$A_0' = (x^{u_i} - 1)A_{0,L}^{-1}\left[a_{1,i}(x) + A_{0,L}^{-1}A_{0,H}a_{2,i}(x) + A_{0,L}^{-1}A_{1,L}a_{3,i} + A_{0,L}^{-1}A_{1,H}a_{4,i}\right] + \gamma.$$

Now $a_{1,i}(x)$ has uniform random independent coefficients and is independent of all the other terms in this expression for A_0'. Thus the result of the sum

$$S := \left[a_{1,i}(x) + A_{0,L}^{-1}A_{0,H}a_{2,i}(x) + A_{0,L}^{-1}A_{1,L}a_{3,i} + A_{0,L}^{-1}A_{1,H}a_{4,i}\right]$$

is uniform random with independent coefficients. Since $(x^{u_i}-1)A_{0,L}^{-1}$ is invertible, $(x^{u_i}-1)A_{0,L}^{-1}S$ is uniform random and with independent coefficients and remains so after adding γ. Since u is an LWE cipher, least one value $u_i \neq 0$ with extremely high probability. $\square$

Next we estimate the error bound on the final ciphertext of the HomLift algorithm. The following lemma is due to Chillotti et. al. [23] but we make the error bound explicit for our model.

Lemma 5.2 (Homomorphic Lifting)**.** *Let $bk = C_0, ..., C_{n-1}$ be a bootstrapping key, $\mathbf{u} = (u_0, ..., u_{n-1}, u_n) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$. Suppose $RE_s(m(x)) \in R_{m,Q}^2$ is a trivial encryption and each C_k, $0 \le k \le n-1$, has error size at most τ_{bk}. Then*

$$RE_s(m(x)) \odot (G + (x^{u_0}-1)C_0) \odot \cdots \odot (G + (x^{u_{n-1}}-1)C_{n-1}) \equiv RE_s\left(m(x)x^{\sum_{k=0}^{n-1} u_k s_k}\right)$$

with error bounded by $15 \cdot 2B\tau_{bk}\sqrt{2\ell mn}$ with probability $1 - m2^{-161}$. In particular, we can let $m(x) = t(x)x^{-u_n}D_Q$ to get $RE_{\mathbf{s}}(t(x)x^{-u_n+\sum_{k=0}^{n-1} u_k s_k})$.

Proof. By assumption for $0 \le k \le n-1$, $C_k = \mathrm{GSW}_{\mathbf{s}}(s_k) = A_k + s_k G$ where $A_k \in R_{m,Q}^{(2\ell)\times 2}$ and $A_k(-s(x), 1)^t \equiv \mathbf{w}^{(k)}(x) \bmod (x^n+1, Q)$ with $\mathbf{w}^{(k)}(x) \in R_m$ and $||\mathbf{w}^{(k)}(x)||_\infty \le \tau_{bk}$. Then using the identity in (12),

$$G + (x^{u_k}-1)C_k = (x^{u_k}-1)A_k + (1 + (x^{u_k}-1)s_k)G = (x^{u_k}-1)A_k + x^{u_k s_k}G = \mathrm{GSW}_{\mathbf{s}}(x^{u_k s_k}),$$

and the error polynomial is $(x^{u_k}-1)A_k(-s(x), 1)^t \equiv (x^{u_k}-1)\mathbf{w}^{(k)}(x)$. So the product we need to compute becomes

$$RE_s(m(x)) \odot GSW_{\mathbf{s}}(x^{u_0 s_0}) \odot \cdots \odot GSW_{\mathbf{s}}(x^{u_{n-1}s_{n-1}}). \tag{13}$$

Let $R_k := RE_s(m(x)) \odot GSW_{\mathbf{s}}(x^{u_0 s_0}) \odot \cdots \odot GSW_{\mathbf{s}}(x^{u_k s_k})$ have error term $e_k(x)$ and flattening $\mathbf{h}^{(k)} \in R_{m,Q}^{(2\ell)}$ so in computing the $(k-2)^{th}$ product $R_k \odot GSW_{\mathbf{s}}(x^{u_{k+1}s_{k+1}}) \equiv RE_s\left(m(x)x^{\sum_{j=0}^{k+1} u_j s_j}\right)$ by Lemma 3.8 the new error term is $\mathbf{h}^{(k)}(x^{u_{k+1}}-1)\mathbf{w}^{k+1}(x)^t + x^{u_{k+1}s_{k+1}}e_k(x)$. In the full product (13) the error term is the following, letting $\psi_\alpha(x) := x^{\sum_{j=\alpha}^{n-1} u_j s_j}$,

$$\psi_0 w^{(-1)} + \psi_1(x^{u_0}-1)\mathbf{h}^{(-1)}\mathbf{w}^{(0)t} + \psi_2(x^{u_1}-1)\mathbf{h}^{(0)}\mathbf{w}^{(1)t} + \cdots + \psi_n(x^{u_{n-1}}-1)\mathbf{h}^{(n-2)}\mathbf{w}^{(n-1)t} \tag{14}$$

where $w^{(-1)}$ and $\mathbf{h}^{(-1)}$ are the error term and flattening of $RE_{\mathbf{s}}(m(x))$. Splitting this into two parts for the sake of independence,

$$\psi_0 w^{(-1)} + \sum_{k=1}^{n} \psi_k x^{u_{k-1}}\mathbf{h}^{(k-2)}\mathbf{w}^{(k-1)t} - \sum_{k=1}^{n} \psi_k \mathbf{h}^{(k-2)}\mathbf{w}^{(k-1)t}, \tag{15}$$

we will analyze the first $n+1$ terms together and the latter n terms together. In analyzing the former, we can drop the $\psi_k x^{u_{k-1}}$ since it just rotates the coefficients and look just at

$$w^{(-1)} + \sum_{k=1}^{n} \mathbf{h}^{(k-2)}\mathbf{w}^{(k-1)t}. \tag{16}$$

To analyze one inner product $\mathbf{h}^{(k)}\mathbf{w}^{(k+1)t}$ we will drop the superscripts to ease notation and write

$$\begin{aligned}
\mathbf{h}\mathbf{w}^t &\equiv \sum_{i=1}^{2\ell} h_i(x)w_i(x) \text{ where } h_i(x), w_i(x) \in R_{m,Q} \\
&\equiv \sum_{i=1}^{2\ell} \left(\sum_{j=0}^{m-1} h_{ij}x^j\right)\left(\sum_{\gamma=0}^{m-1} w_{i\gamma}x^\gamma\right) \\
&\equiv \sum_{i=1}^{2\ell}\sum_{j=0}^{m-1}\sum_{\gamma=0}^{m-1} h_{ij}w_{i\gamma}x^{\gamma+j} \\
&\equiv \sum_{i=1}^{2\ell}\sum_{j=0}^{m-1} \underbrace{[(h_{i0}w_{ij} + \cdots h_{ij}w_{i0}) - (h_{i,j+1}w_{i,n-1} + \cdots + h_{i,n-1}w_{i,n-(n-1-j)})]}_{m-1 \text{ terms}} x^j. \\
&\equiv \sum_{j=0}^{m-1}\sum_{\gamma=1}^{2\ell(m-1)} h_\gamma \tilde{w}_\gamma x^j.
\end{aligned}$$

where in the last line, since the w_{ij}'s come from a distribution that is symmetric about zero, the $-w_{ij}$'s can be rewritten as $\tilde{w}_{ij}$ which are still iid.

Now each coefficient of x^j in the sum in (16) has the form

$$\sum_{\gamma=1}^{n(2\ell(m-1))+1} h_\gamma \tilde{w}_\gamma.$$

Note that among the error terms in the boostrapping key there are $2\ell n(m-1)$ iid coefficients; and these are all represented in these w_γ's here. The one other term comes from the error term of the initial $\mathrm{RE}_{\mathbf{s}}(m(x))$, which is just zero when it is a trivial ciphertext. Thus this is the sum of $N := n(2\ell(m-1))$ iid bounded uniform random variables $|w_\gamma| \le \tau_{bk}$ or also $\mathrm{subG}(\tau_{bk}^2)$. However, the h_γ coefficients are not independent of the w_γ's in that in (16) $\mathbf{h}^{(i)}$ depends on all the previous $\mathbf{h}^{(j)}, \mathbf{w}^{(j)}$, $j < i$. Thus we cannot directly apply Cor 1.7 of [54] to say the resulting sum is sub-Gaussian, but we can use the result form [56, Corollary 2.4] to say this,

$$(\text{Coef of } x^j) \sim \mathrm{subG}(\tau_{bk}^2 ||\mathbf{h}||_2^2).$$

To bound the error, we need to consider bounding all m coefficients at once and do so using Thm 1.14 of [54],

$$\mathrm{Prob}\left[\max_{0 \le j \le m-1} |\text{Coef of } x^j| < t\right] \le 2m \exp\left(\frac{-t^2}{2\tau_{bk}^2 ||\mathbf{h}||_2^2}\right).$$

In particular, to get a bound with very high probability we choose a bound of 15 standard deviations, $t = 15(\tau_{bk}||\mathbf{h}||_2) \le 15(\tau_{bk})\sqrt{NB^2} \le 15 \cdot B\tau_{bk}\sqrt{2\ell mn} =: \tau_1$, so that the probability of any coefficient exceeding this bound is less than $2m\exp(-15^2/2) \le m2^{-161}$. Similarly for the last n terms of Equation (15), N will be $N := n(2\ell(m-1))$ and the probability that any coefficient exceeds $15 \cdot B\tau_{bk}\sqrt{2\ell mn} =: \tau_2$ is at most $m2^{-161}$. Thus the probability that any coefficient in (15) exceeds $\tau_1 + \tau_2 = 15 \cdot 2B\tau_{bk}\sqrt{2\ell mn}$ is at most $m2^{-161}$. □

5.2 Function Lookup.

Now for any function of the following three forms

(i) $f : [0, 2^{k+1}) \to [0, 2^k)$,

(ii) $f : (-2^k, 2^k) \to [0, 2^k)$,

(iii) $f : [0, 2^k) \to [0, 2^k)$,

we want to find $E_{\mathbf{s}}(f(y))$. The three domains correspond to the three choices (i,ii, iii) for $\mathbf{u}$, y, and e in Section (5.1).

Using the technique of Carpov et. al. [17] we define the *function encoding polynomial* as

$$F(x) := \sum_{i \in M} f(i) x^{iD_r}.$$

Multiplying equation (11) by $F(x)$ we get

$$\Phi_f(x) := t(x)x^{-w}F(x) \equiv t(x)x^{-e}x^{-yD_r}F(x) \pmod{x^m + 1, Q}. \tag{17}$$

Now we claim that the constant term of this polynomial is $f(y) \pmod{Q}$, the function value of which we want an encryption.

Lemma 5.3. *For $y = x_1 + x_2 \in [0, 2^{k+1})$ and for any function $f : [0, 2^{k+1}) \to [0, 2^k)$ the constant term of $\Phi_f(x)$ is $f(y)$. Similarly, for $y = x_1 - x_2 \in (-2^k, 2^k)$ and $f : (-2^k, 2^k) \to [0, 2^k)$, the constant term of $\Phi_f(y)$ is $f(y)$. Similarly for $y = x_1 \in [0, 2^k)$ and $f : [0, 2^k) \to [0, 2^k)$.*

Proof. **Case 1.** For the first case of $f : [0, 2^{k+1}) \to [0, 2^k)$ we have by assumption,

$$\Phi_f(x) \equiv t(x)x^{-e}x^{-yD_r} \sum_{i \in [0, 2^{k+1})} f(i)x^{iD_r} \pmod{x^m + 1}.$$

The polynomial $t(x)$ has a term x^e which can cancel with x^{-e}. $F(x)$ has a term $f(y)x^{yD_r}$ which can cancel with x^{-yD_r} to give $f(y)$. Thus, in $\Phi_f(x)$ there is the term

$$x^e \cdot x^{-e} \cdot x^{-yD_r} \cdot f(y)x^{yD_r} \equiv f(y) \pmod{x^m + 1, q}.$$

We just need to show that this is the only constant term in $\Phi_f(x)$. In other words, if $|e_1| < D_r/2$ and $i \in [0, 2^{k+1})$ so that x^{e_1} is a term in $t(x)$ and $f(i)x^{iD_r}$ is a term in $F(x)$ with either $e_1 \neq e$ or $i \neq y$, we need to show that we cannot get $e_1 - e + (i - y)Dr \equiv 0 \pmod{m}$. With either $e_1 \neq e$ or $i \neq y$, we have

$$\begin{aligned} 0 \neq |(e_1 - e) + (i - y)D_r| &\leq |e_1 - e| + |(i - y)D_r| \\ &< D_r + (2^{k+1} - 1)D_r \\ &= 2^{k+1}D_r \\ &= r/2 = m. \end{aligned}$$

Thus, $e_1 - e + (i - y)D_r \not\equiv 0 \pmod{m}$. So the only term contributing to the constant term of $\Phi_f(x)$ is $f(y)$.

Case 2. In the second case of $f : (-2^k, 2^k) \to [0, 2^k)$ we have,

$$\Phi_f(x) \equiv t(x)x^{-e}x^{-yD_r} \sum_{i \in (-2^k, 2^k)} f(i)x^{iD_r} \pmod{x^m + 1}.$$

The polynomial $t(x)$ has a term x^e which can cancel with x^{-e}. $F(x)$ has a term $f(y)x^{yD_r}$ which can cancel with x^{-yD_r} to give $f(y)$. Thus, in $\Phi_f(x)$ there is the term

$$x^e \cdot x^{-e} \cdot x^{-yD_r} \cdot f(y)x^{yD_r} \equiv f(y) \pmod{x^m + 1, Q}.$$

We just need to show that this is the only constant term in $\Phi_f(x)$. In other words, if $|e_1| < D_r/2$ and $i \in (-2^k, 2^k)$ so that x^{e_1} is a term in $t(x)$ and $f(i)x^{iD_r}$ is a term in $F(x)$ with either $e_1 \neq e$ or $i \neq y$, we need to show that we cannot get

$$e_1 - e + (i-y)D_r \equiv 0 \pmod{m}.$$

We have $(i-y) \in [-2^{k+1}+2, 2^{k+1}-2]$, and with either $e_1 \neq e$ or $i \neq y$,

$$\begin{aligned} 0 \neq |(e_1 - e) + (i-y)D_r| &\leq |e_1 - e| + |(i-y)D_r| \\ &< D_r + (2^{k+1}-2)D_r \\ &= 2^{k+1}D_r - D_r \\ &< r/2 = m. \end{aligned}$$

Thus, $e_1 - e + (i-y)D_r \not\equiv 0 \pmod{m}$. So the only term contributing to the constant term of $\Phi_f(x)$ is $f(y)$.

Case 3. The proof is similar to Cases 1 and 2. □

Now we will show how $\Phi_f(x)$ can be computed under encryption using the bootstrapping key, bk, and $\mathrm{E}_{\mathbf{s}}(x_1)$ and $\mathrm{E}_{\mathbf{s}}(x_2)$. Recall,

$$x^{-w} \equiv x^{-u_n} x^{\sum_{i=0}^{n-1} s_i u_i},$$

so an untrusted party performing the HE computations with bk and $\mathbf{u} = (u_0, ..., u_n)$ can compute a HomLift to get $\mathrm{RE}_{\mathbf{s}}(t(x)x^{-u_n}x^{\sum_{k=0}^{n-1} u_k s_k})$ with the error bound given by Lemma 5.2. Getting $F(x)$ into the product can be done by observing that

$$\mathrm{RE}_{\mathbf{s}}(\Phi_f(x)) = \mathrm{RE}_{\mathbf{s}}(t(x)x^{-u_n}x^{\sum_{k=0}^{n-1} u_k s_k}) \cdot F(x) \tag{18}$$

with the error term $F(x)w(x)$. At this point the final error bound will partially depend on the choice of the function f, which will vary by operation being performed. But we can consider a worst case bound on the range of f to be $[0, 2^k)$, and since the three possible domains $M = [0, 2^{k+1})$, $M = (-2^k, 2^k)$ and $M = [0, 2^k)$ have the size bounded by 2^{k+1} we can proceed with a general worst case analysis. The error term is

$$F(x)w(x) = \sum_{i \in M} f(i)x^{iD_r} \sum_{k=1}^{m} w_k x^k \equiv \sum_{i \in M} \sum_{k=1}^{m} f(i) w_k x^{k+iD_r} \pmod{x^m+1, Q}.$$

Taking a worst case bound and using the bound on $w(x)$ from Lemma 5.2 we have all the coefficients are bounded by

$$||F(x)w(x)||_\infty \leq (2^k \cdot 2^{k+1})(15 \cdot 2B\tau_{bk}\sqrt{2\ell mn}) = 15(2^{2k+2})B\tau_{bk}\sqrt{2\ell mn} \tag{19}$$

still with probability less than $1 - m2^{-161}$. Thus we get $\mathrm{RE}_{\mathbf{s}}(\Phi_f(x))$ with this error bound.

In (18) we can denote the parts of the $\mathrm{RE}_{\mathbf{s}}$ ciphers as

$$(a(x)F(x), b(x)F(x)) = (a(x), b(x)) \cdot F(x).$$

From Lemma 5.1, we know that $a(x)$ is uniform random with independent coefficients. If $F(x)$ is invertible in $R_{m,Q}$, then $a(x)F(x)$ will also be uniform random in $R_{m,Q}$ with independent coefficients. In almost all cases, $F(x)$ will be invertible, and if ever we needed to use an $F(x)$ that was not invertible, we could encrypt $F(x)$ as a GSW cipher using a public key from the boostrapping key and then perform an external product. Lemma 5.1 would then guarantee that the resulting first component of $\mathrm{RE}_{\mathbf{s}}(\Phi_f(x))$ would be uniform random with independent coefficients.

Next, we will show how to extract an LWE cipher in $\mathbb{Z}_Q^n \times \mathbb{Z}_Q$ for the constant term of $\Phi_f(x)$, which is $f(y)$ by Lemma 5.3. We denote

$$\mathrm{RE}_{\mathbf{s}}(\Phi_f(x)) = (a(x), b(x)) = \left(\sum_{i=0}^{m-1} a_i x^i, \sum_{i=0}^{m-1} b_i x^i\right).$$

As in Section 4.3.2, for $a(x) = a_0 + a_1 x + \cdots + a_{m-1}x^{m-1} \in R_m$ we define

$$\mathrm{Extract}(a(x), i) = (a_i, a_{i-1}, \ldots, a_0, -a_{m-1}, -a_{m-2}, \ldots, -a_{m-(n-1-i)}) \in \mathbb{Z}_Q^n.$$

This is used to get the coefficients of $a(x)s(x) \bmod (x^m+1)$. Note that, modulo x^m+1,

$$\begin{aligned} a(x)s(x) &= \sum_{k=0}^{n-1}\sum_{j=0}^{m-1} s_k a_j x^{k+j} \\ &\equiv \sum_{i=0}^{m-1} [(s_0 a_i + s_1 a_{i-1} + \cdots + s_i a_0) \\ &\qquad -(s_{i+1}a_{m-1} + s_{i+2}a_{m-2} + \cdots + s_{n-1}a_{m-(n-1-i)})]x^i. \end{aligned}$$

Hence, for $0 \le i \le n$, the i-th coefficient of $a(x)s(x) \bmod (x^m+1)$ is equal to the inner product of $\mathbf{s}$ with $\mathrm{Extract}(a(x), i)$.

Let $\mathbf{a} := \mathrm{Extract}(a(x), 0)$ and $\mathbf{c} := (\mathbf{a}, b_0)$. It follows that

$$b_0 \equiv \langle \mathbf{s}, \mathbf{a}\rangle + f(y)\tilde{D}_Q + v \pmod Q$$

where $|v| \le \tau$ where τ is the bound in equation (19). So $\mathbf{E}_s(f(y)) = (\mathbf{a}, b_0) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$ is an LWE cipher with error bounded by τ with probability $1 - m2^{-161}$.

Now we will use a modulus reduction to take $\mathbf{E}_s(f(y)) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$ to an LWE cipher in $\mathbb{Z}_r^n \times \mathbb{Z}_r$. For the modulus reduction Lemma 5.4 we need $\tau \le Q\sqrt{n}/r$ which if we solve for Q is

$$15(2^{2k+2})Br\tau_{bk}\sqrt{2\ell m} \le Q. \tag{20}$$

Taking Q at least this big, we get after the modulus reduction $\mathbf{E}_s(f(y)) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ has error bounded by $4\sqrt{n}$. We give the details of the modulus reduction next.

5.3 Modulus reduction

We describe how an LWE ciphertext over $\mathbb{Z}_Q$ can be converted to an LWE ciphertext over $\mathbb{Z}_r$ where r is much smaller than Q in our FHE scheme. This technique of modulus reduction is used in [14, 15, 16].

Lemma 5.4. ***(Modulus reduction: deterministic rounding.)*** *If $2^{k+2}|n$, $n \ge 2^{10}$, $x \in \{0, 1, \cdots, 2^k - 1\}$, $Q, r \in \mathbb{Z}^+$ s.t. Q is odd, $r \ge 2^{k+6}\sqrt{n}$, $Q \ge 2^{k+2}\cdot r$, $e \in \mathbb{Z}$ with $|e| \le \tau = \frac{Q\sqrt{n}}{r}$, $D_r = \lfloor r/2^{k+2}\rfloor$, $D_Q = \lfloor Q/2^{k+2}\rfloor$, given $\mathbf{a} \in \mathbb{Z}_Q^n$ uniformly random with independent components, $\mathbf{s} \in \{0,1\}^n$ with $Ham(\mathbf{s}) \le \rho n$ and*

$$b \equiv \langle \mathbf{s}, \mathbf{a}\rangle + e + xD_Q \pmod Q,$$

let

$$b' = \lfloor rb/Q\rceil, \quad \mathbf{a}' = \lfloor r\mathbf{a}/Q\rceil,$$

computed component wise, then

$$b' \equiv \langle \mathbf{s}, \mathbf{a}'\rangle + e' + xD_r \pmod r$$

for some $e' \in \mathbb{Z}$ s.t. $|e'| < 4\sqrt{n}$ with probability $\ge 1 - 2^{-(\frac{17.7}{\rho \ln 2} - 1)}$.

Proof. Note that

$$b' = \frac{rb}{Q} + \epsilon_0, \quad a_i' = \frac{ra_i}{Q} + \epsilon_i, \quad 1 \le i \le n,$$

for some $\epsilon_i \in \mathbb{R}$ with $|\epsilon_i| < 1/2$ for $0 \le i \le n$. The fact that $|\epsilon_i| \neq 1/2$ comes from that Q is odd. Let $\gcd(r, Q) = d$, $Q = q_1 d$ and $r = r_1 d$. Then $ra_i/Q = r_1 a_i/q_1$. Since all the a_i are independently uniform random in $\mathbb{Z}_Q$, we know all the $r_1 a_i$ are independently uniform in $\mathbb{Z}_{q_1}$, implying that $\frac{ra_i}{Q}$ are also independently uniform random on $\frac{1}{q_1}\mathbb{Z}/(q_1\mathbb{Z})$, (by which we mean the set of fractions consisting of the standard representatives of $\mathbb{Z}_{q_1}$ all divided by q_1). By choosing the closest integer to $\frac{ra_i}{Q}$, we know that $E(\epsilon_i) = 0$ and $|\epsilon_i| < 1/2$ for $1 \le i \le n$, implying that $\epsilon_i \sim \text{subG}(1/4)$ and are independent. Since $b = \mathbf{s}\,\mathbf{a}^t + e + xD_Q + Qy$ for some integer y, we have

$$\frac{rb}{Q} = \sum_{i=1}^{n} s_i \frac{ra_i}{Q} + \frac{re}{Q} + x\frac{rD_Q}{Q} + ry,$$

hence $b' = \sum_{i=1}^n s_i a_i' + xD_r + e' + ry$, where e' is an integer and

$$e' = \epsilon_0 - \sum_{i=1}^{n} s_i \epsilon_i + x\left(\frac{rD_Q}{Q} - D_r\right) + \frac{re}{Q}.$$

If $2^{k+2}|n$ and $Q \ge 2^{k+2} \cdot r$, then in fact $D_r = r/2^{k+2}$. Noticing that

$$|\frac{r}{Q}D_Q - D_r| = \frac{r}{Q}|D_Q - \frac{Q}{r}D_r| = \frac{r}{Q}|\lfloor Q/2^{k+2}\rfloor - Q/2^{k+2}| \le \frac{r}{Q} \le \frac{1}{2^{k+2}},$$

we know that

$$|x(\frac{r}{Q}D_Q - D_r)| \le |x| \cdot \frac{1}{2^{k+2}} < 2^k \cdot \frac{1}{2^{k+2}} = 1/4.$$

We apply Corollary 1.7 in [54] for $\sum_{i=1}^n s_i\epsilon_i$ with bound $\frac{2.975}{\sqrt{\rho}}\|\mathbf{s}\| \le 2.975\sqrt{n}$ to obtain

$$P(|\sum_{i=1}^{n} s_i\epsilon_i| > 2.975\sqrt{n}) = P(|\sum_{i=1}^{n} s_i\epsilon_i| > \frac{2.975}{\sqrt{\rho}}\|\mathbf{s}\|) < 2 \cdot e^{-\frac{17.7}{\rho}} < 2 \cdot 2^{-\frac{17.7}{\rho \ln 2}} = 2^{-(\frac{17.7}{\rho \ln 2} - 1)}.$$

Thus with probability $\ge 1 - 2^{-(\frac{17.7}{\rho \ln 2} - 1)}$, since $n \ge 2^{10}$,

$$\begin{aligned} |e'| &< 1/2 + 2.975\sqrt{n} + 1/4 + \sqrt{n} \\ &< 3/4 + 3.975\sqrt{n} \\ &< 0.025\sqrt{n} + 3.975\sqrt{n} \\ &= 4\sqrt{n} \end{aligned}$$

which completes the proof. □

Our generic bootstrapping algorithm is described in Figure 5, for computing one HomLift followed by an independent number of function lookups.

Theorem 5.5. *Suppose a bootstrapping key bk has error size at most τ_{bk}, r is divisible by 2^{k+2} and*

$$r \ge 2^{k+6}\sqrt{n}, \quad Q \ge 15(2^{2k+2})Br\tau_{bk}\sqrt{2\ell m}$$

Then, for any LWE cipher $E_{\mathbf{s}}(y) = \mathbf{u} \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size $< D_r/2$ the bootstrapping algorithm in Figure 5 outputs random LWE ciphers $E_{\mathbf{s}}(f_j(y))$, $\in \mathbb{Z}_r^n \times \mathbb{Z}_r$, all with error $< D_r/4 = 4\sqrt{n}$ for any lookup functions $f_j : M \to [0, 2^k)$.

```
Generic Bootstrapping Algorithm : $\mathrm{BT}_{bk}(\mathbf{u})$
Input:    $bk = (C_0, \ldots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$: bootstrapping key,
          $\mathbf{u} \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{u} = \mathrm{E}_{\mathbf{s}}(y)$ .
Output:   $E_{\mathbf{s}}(f_j(y))$, for $1 \le j \le \alpha$ for $f_j : M \to [0, 2^k)$ where
          $M = [0, 2^{k+1})$ or $(-2^k, 2^k)$ or $[0, 2^k)$.
Step 1.   HomLift:
              $A := \mathrm{HomLift}_{bk}(\mathbf{u}, bk)$
Step 2.   Function lookup: For $j$ from 1 to $\alpha$ do
              $F_j(x) := \sum_{i\in M} f(i)x^{iD_r}$, $1 \le j \le \alpha$,
              $A_j := A \cdot F_j(x)$
Step 3.   Extract: suppose $A_j = (a_j(x), \sum_{i=0}^{m-1} b_{ji}x^i) \in R_{m,Q}^2$. Set
              $\mathbf{a}_j := (\mathrm{Extract}(a_j(x), 0), b_{j0}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 4.   Modulus reduction: For $j$ from 1 to $\alpha$ do
              $\mathbf{c}_j := \lfloor r\mathbf{a}_j/Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 5.   Return $\mathbf{c}_j$, $1 \le j \le \alpha$.
```

Figure 5: Boostrapping Algorithm

Proof. Let $w = u_n - \sum_{i=0}^{n-1} s_i u_i$. By Lemma 5.2 after Step 1, $A = \mathrm{RE}_{\mathbf{s}}(t(x)x^{-w})$ with error bounded by $15 \cdot 2B\tau_{bk}\sqrt{2\ell mn}$ with probability $1 - m2^{-161}$.

At Step 2, letting $A = (a(x), b(x)) \in R_{m,Q}$, we have

$$b_j(x) \equiv a_j(x)s(x) + t(x)x^{-w}D_Q + v(x) \mod (x^m + 1, Q), \tag{21}$$

and $v(x) \in R_m$ with $||v(x)||_\infty \le 15 \cdot 2B\tau_{bk}\sqrt{2\ell mn}$ with probability $1 - m2^{-161}$. Now letting $A_j = (a(x)F_j(x), b(x)F_j(x))$, we have

$$b_j(x)F_j(x) \equiv a_j(x)F_j(x)s(x) + t(x)x^{-w}F_j(x)D_Q + v(x)F_j(x) \mod (x^m + 1, Q), \tag{22}$$

with by equation (19), $||v(x)F(x)||_\infty \le 15(2^{2k+2})B\tau_{bk}\sqrt{2\ell mn}$ with probability $1 - m2^{-161}$. In Step 3 let

$$\mathbf{u}_j := \mathrm{Extract}(a_j(x), 0).$$

We have

$$b_{j0} \equiv \langle \mathbf{s}, \mathbf{u}_j\rangle + c_j D_Q + v_{j0} \pmod Q,$$

where c_j is the constant term of $t(x)x^{-w}F_j(x)$, which by Lemma 5.3 is $f_j(y)$ and $|v_{j0}| \le 15(2^{2k+2})B\tau_{bk}\sqrt{2\ell mn}$ with probability $1 - m2^{-161}$.

At Step 5, for $\rho = 1/8$, we apply the modulus reduction in Lemma 5.4 to $\mathbf{a}_j$ to get LWE ciphers in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size $< 4\sqrt{n} = D_r/4$ for probability greater than $1 - 2^{-143}$. To apply this modulus reduction, we need $\mathbf{a}_j$ to have uniform random independent entries, and this follows from Lemma 5.1. □

5.4 Select k-bit Arithmetic Operations

Using the HomLift procedure and a variety of novel functions and relations we are able to perform many useful arithmetic operations homomorphically.

Addition (mod $\mathbf{p \le 2^k}$). To homomorphically add $x_1, x_2 \in [0, 2^k)$ modulo $p \le 2^k$, we take $y = x_1 + x_2$ in Equation (10) and use the function

$$f : [0, 2^{k+1}) \to [0, p) \subset \mathbb{Z},$$

$$y \mapsto y \pmod{p}.$$

Then for two ciphertext of x_1 and x_2 we can perform the addition using one HomLift and f:

$$\text{Addmodp}(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) = E_{\mathbf{s}}(f(x_1 + x_2)) = E_{\mathbf{s}}(x_1 + x_2 \pmod{p}).$$

The algorithm is show in Figure 6.

```
            **Addition (mod p ≤ 2^k)** : Addmodp(E_s(x_1), E_s(x_2), p)
Input:      $bk = (C_0, \dots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$ : bootstrapping key,
            $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{v}_i = E_{\mathbf{s}}(x_i)$ for $x_1, x_2 \in [0, 2^k)$.
Output:     $\mathrm{E}_{\mathbf{s}}(x_1 + x_2 \pmod{p})$
Step 1.     Compute $\mathbf{u} := \mathbf{v}_1 + \mathbf{v}_2 = (u_0, \dots, u_{n-1}, u_n) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 2.     HomLift:
                $A := \text{HomLift}(\mathbf{u}, bk)$
Step 3.     Function lookup:
                $F(x) := \sum_{i \in M} f(i) x^{iD_r}$, with $M = [0, 2^{k+1})$
                and $f(y) = y \pmod{p}$.
                $A := A \cdot F(x)$
Step 4.     Extract: suppose $A = (a(x), \sum_{i=0}^{m-1} b_i x^i) \in R_{m,Q}^2$. Set
                $\mathbf{a} := (\text{Extract}(a(x), 0), b_0) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 5.     Modulus reduction:
                $\mathbf{c} := \lfloor r\mathbf{a}/Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.     Return $\mathbf{c}$.
```

Figure 6:

Subtraction (mod $\mathbf{p \le 2^k}$) To homomorphically subtract two k bit integers $x_1, x_2 \in [0, 2^k)$ modulo $p \le 2^k$, we take $y = x_1 - x_2$ and use the function

$$f : (-2^k, 2^k) \to [0, p) \subset \mathbb{Z}$$

$$y \mapsto y \pmod{p}.$$

Then for two ciphertext of x_1 and x_2 we can perform the following using one HomLift and f

$$\text{Submodp}(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) = E_{\mathbf{s}}(f(x_1 - x_2)) = E_{\mathbf{s}}(x_1 - x_2 \pmod{p}).$$

Addition in $\mathbb{Z}$ For $x_1, x_2 \in [0, 2^k)$, $x_1 + x_2 = y_0 + y_1 2^k \in \mathbb{Z}$ where y_0 has k bits and y_1 has one bit. Since the sum of x_1 and x_2 in $\mathbb{Z}$ will be a $k+1$ bit message, if we want to store that result in a ciphertext will need to store it across two ciphertext, since each ciphertext is the encryption of only k message bits. We will need two functions to do this, f_0 will extract the lower k bits,

$$f_0 : [0, 2^{k+1}) \to \mathbb{Z},$$

$$y \mapsto y \pmod{2^k} = y_0.$$

Another function f_1 will extract the 1 highest bit, $f_1 : [0, 2^{k+1}) \to \mathbb{Z}$, $y \mapsto \lfloor y/2^k \rfloor = y_1$.

Then for ciphertexts of x_1 and x_2 we can get an encryption of $x_1 + x_2$ in $\mathbb{Z}$ spread across two ciphertexts using one bootstrapping and two function lookups,

$$\begin{aligned}\text{Addin}\mathbb{Z}(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) &= (E_{\mathbf{s}}(f_0(x_1 + x_2)), E_{\mathbf{s}}(f_1(x_1 + x_2))) \\ &= (E_{\mathbf{s}}(y_0), E_{\mathbf{s}}(y_1)).\end{aligned}$$

Multiplication (mod $\mathbf{p < 2^k}$ odd). To multiply $x_1, x_2 \in [0, 2^k)$ modulo $p < 2^k$, p odd, and get one cipher for $x_1 \cdot x_2 \pmod p \in [0, 2^k)$, the key will be to use the following identity which holds in $\mathbb{Z}$

$$x_1 \cdot x_2 = \left(\frac{x_1 + x_2}{2}\right)^2 - \left(\frac{x_1 - x_2}{2}\right)^2. \tag{23}$$

If we consider this equation modulo p for p odd, 2 will have an inverse and (23) becomes

$$x_1 \cdot x_2 \equiv ((x_1 + x_2)2^{-1})^2 - (x_1 - x_2)2^{-1})^2 \pmod p.$$

Note that if p were even, 2 would not have an inverse modulo p and this equation would not be well defined. We will find intermediate ciphertexts for $z_1 := \left(\frac{x_1+x_2}{2}\right)^2 \pmod p$ and $z_2 := \left(\frac{x_1-x_2}{2}\right)^2 \pmod p$, using two HomLifts. First for $z_1 := \left(\frac{x_1+x_2}{2}\right)^2 \pmod p$ we have $x = x_1 + x_2 \in [0, 2^{k+1})$. We will use the function

$$f_1 : [0, 2^{k+1}) \to [0, p) \subset \mathbb{Z},$$

$$f_1(x) = \left(x \cdot 2^{-1}\right)^2 \pmod p.$$

Then one HomLift and one function lookup gives a ciphertext $E_s(z_1)$.

Now to get the intermediate ciphertext for z_2 we will perform another HomLift of $u := \mathbf{E}_s(x_1) - \mathbf{E}_s(x_2)$ and use the function

$$f_2 : (-2^k, 2^k) \to [0, p) \subset \mathbb{Z},$$

$$f_2(x) = \left(x(2^{-1})\right)^2 \pmod p.$$

Thus one HomLift and one function lookup gives a ciphertext $E_s(z_2)$.

Finally, we will need to combine these two ciphertexts $E_s(z_1)$ and $E_s(z_2)$ to get one for $x_1 \cdot x_2 \equiv z_1 - z_2 \pmod p$. We have $z_1 - z_2 \in (-2^k, 2^k)$, and we will use the function

$$f_3 : (-2^k, 2^k) \to [0, p) \subset \mathbb{Z},$$

$$f_3(x) = x \pmod p.$$

Now the one final HomLift and function lookup gives us a ciphertext for $x_1 \cdot x_2 \equiv z_1 - z_2 \pmod p$. In total, we needed three HomLifts to do this multiplication modulo $p < 2^k$ odd. The procedure is shown in Figure 7.

Multiplication (mod $\mathbf{2^k}$). To take two cipher for $x_1, x_2 \in [0, 2^k)$ and get one ciphers for $x_1 \cdot x_2 \pmod{2^k} \in [0, 2^k)$, we cannot use the identity in (23) modulo 2^k because 2 would not have an inverse, but we can use the following modified identity that holds modulo 2^k

$$x_1 \cdot x_2 \equiv \left\lfloor \frac{(x_1 + x_2)^2 \pmod{2^{k+2}}}{4} \right\rfloor - \left\lfloor \frac{(x_1 - x_2)^2 \pmod{2^{k+2}}}{4} \right\rfloor \pmod{2^k}. \tag{24}$$

Using this we can compute a cipher of k bits for $z_1 := \left\lfloor \frac{(x_1+x_2)^2 \pmod{2^{k+2}}}{4} \right\rfloor$ using a HomLift of $y = x_1 + x_2$ and the function

$$f_1 : [0, 2^{k+1}) \to [0, 2^k),$$

$$f_1(x) = \left\lfloor \frac{x^2 \pmod{2^{k+2}}}{4} \right\rfloor.$$

```
**Mult (mod p < 2^k odd)** : Multmodpodd($\mathrm{E}_{\mathbf{s}}(x_1), \mathrm{E}_{\mathbf{s}}(x_2), p$)
Input:    $bk = (C_0, \dots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$: bootstrapping key,
          $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{v}_i = E_{\mathbf{s}}(x_i)$ for $x_1, x_2 \in [0, 2^k)$.
          $p < 2^k$ odd.
Output:   $\mathrm{E}_{\mathbf{s}}(x_1 \cdot x_2 \pmod{p})$
Step 1.   Compute $\mathbf{u} := \mathbf{v}_1 + \mathbf{v}_2$, $\tilde{\mathbf{u}} := \mathbf{v}_1 - \mathbf{v}_2 \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 2.   HomLift:
              $A_1 := \mathrm{HomLift}(\mathbf{u}, bk)$,   $A_2 := \mathrm{HomLift}(\tilde{\mathbf{u}}, bk)$
Step 3.   Function lookup:
              $A_1 := A_1 \cdot F_1(x)$,   $A_2 := A_2 \cdot F_2(x)$
Step 4.   Extract: For $j = 1, 2$ suppose $A_j = (a_j(x), \sum_{i=0}^{m-1} b_{ji} x^i) \in R_{m,Q}^2$. Set
              $\mathbf{a}_j := (\mathrm{Extract}(a_j(x), 0), b_{j0}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 5.   Modulus reduction: For $j = 1, 2$ do
              $\mathbf{c}_j := \lfloor r\mathbf{a}_j / Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.   Compute $\mathbf{u} := \mathbf{c}_1 - \mathbf{c}_2$
Step 7.   HomLift:
              $A := \mathrm{HomLift}(\mathbf{u}, bk)$,
Step 8.   Function lookup:
              $A := A \cdot F_3(x)$,
Step 9.   Extract: suppose $A = (a(x), \sum_{i=0}^{m-1} b_i x^i) \in R_{m,Q}^2$. Set
              $\mathbf{a} := (\mathrm{Extract}(a(x), 0), b_0) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 10.  Modulus reduction:
              $\mathbf{c} := \lfloor r\mathbf{a} / Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 11.  Return $\mathbf{c}$.
```

Figure 7:

We also compute a cipher of k bits for $z_2 := \left\lfloor \frac{(x_1 - x_2)^2 \pmod{2^{k+2}}}{4} \right\rfloor$ by a HomLift of $y = x_1 - x_2 \in (-2^k, 2^k)$, and the function

$$f_2 : (-2^k, 2^k) \to [0, 2^k),$$

$$f_2(x) = \left\lfloor \frac{x^2 \pmod{2^{k+2}}}{4} \right\rfloor.$$

Finally, we do another HomLift of $z_1 - z_2$ and use the function

$$f_3 : (-2^k, 2^k) \to [0, 2^k),$$

$$f_3(x) = x \pmod{2^k},$$

to get a cipher $E_{\mathbf{s}}(x_1 \cdot x_2 \pmod{2^k})$. In total we needed three HomLifts to do this multiplication modulo 2^k.

Multiplication in $\mathbb{Z}$ Given ciphers $\mathrm{E}_{\mathbf{s}}(x_1)$ and $\mathrm{E}_{\mathbf{s}}(x_2)$ for $x_1, x_2 \in [0, 2^k)$, we want to compute their product in $\mathbb{Z}$, which can be express as

$$x_1 \cdot x_2 = y_0 + y_1 2^k$$

where $y_0 \in [0, 2^k)$ and $y_1 \in [0, 2^k - 1)$. We will need to store the result y_0 and y_1 encrypted in two separate ciphertexts. Using our multiplication mod 2^k we can get a ciphertext for y_0

as $y_0 \equiv x_1 \cdot x_2 \pmod{2^k}$, and using two of the same HomLifts but different lookup functions for our multiplication mod $p = 2^k - 1$, we get an intermediate ciphertext $\tilde{y_1} := x_1 \cdot x_2 \pmod{2^k - 1}$. This is equivalent to the following

$$\begin{aligned}\tilde{y_1} &= x_1 \cdot x_2 \pmod{2^k - 1}\\ &= y_0 + y_1 2^k \pmod{2^k - 1}\\ &\equiv y_0 + y_1 \pmod{2^k - 1},\end{aligned}$$

since $2^k \equiv 1 \pmod{2^k - 1}$. Computing y_0 and $\tilde{y_1}$ takes 4 HomLifts. Finally we get a ciphertext for y_1 by $y_1 = \tilde{y_1} - y_0 \pmod{2^k - 1}$. This brings the total HomLifts to 5. Two pairs of HomLifts can be done in parallel, leaving the number of sequential HomLifts at 3. The procedure is show in Figure 11 where the lookup functions are defined as follows.

1. $f_1 : [0, 2^{k+1}) \to [0, 2^k)$, $f_1(x) = \left\lfloor \frac{x^2 \pmod{2^{k+2}}}{4} \right\rfloor$.
2. $f_2 : (-2^k, 2^k) \to [0, 2^k)$, $f_2(x) = \left\lfloor \frac{x^2 \pmod{2^{k+2}}}{4} \right\rfloor$.
3. $f_3 : [0, 2^{k+1}) \to [0, 2^k - 1)$, $f_3(x) = \left(x(2^{-1})\right)^2 \pmod{2^k - 1}$.
4. $f_4 : (-2^k, 2^k) \to [0, 2^k - 1)$, $f_4(x) = \left(x(2^{-1})\right)^2 \pmod{2^k - 1}$.
5. $f_5 : (-2^k, 2^k) \to [0, 2^k)$, $f_5(x) = x \pmod{2^k}$.
6. $f_6 : (-2^k, 2^k) \to [0, 2^k)$, $f_6(x) = x \pmod{2^k - 1}$.
7. $f_7 = f_6$.

Recall that specifying the lookup function along with its domain is sufficient to define the corresponding function encoding polynomial.

Field inverse in $\mathbb{F}_p$ for $p < 2^k$. The preceding operations have all been binary, combining two ciphertext in some way. These remaining operations are unitary operations performed on a single ciphertext. For $p < 2^k$ prime, $\mathbb{Z}_p$ is a field and as such every element except for 0 has a multiplicative inverse. We can homomorphically compute the inverse $x \in [0, p)$ as follows. We let $\mathbf{u} = \mathbf{E}_s(x)$ and perform a HomLift following by a function lookup using

$$\begin{aligned} f &: (0, p) \to (0, p)\\ f(i) &= i^{-1} \pmod{p}.\end{aligned}$$

Computing $i^{-1} \pmod{p}$ is in general best done using the extended Euclidean algorithm, but for our use here since k is not large f could be computed using a simple table lookup. At this level, we must leave it to the designer of homomorphic circuit to avoid calling this function for 0 which has no inverse. Having this operations completes all the basic field operations in $\mathbb{F}_p$. The procedure is giving in Figure 8 and takes one HomLift. This same operation works if p is not prime, but in that case there will be more elements that do not have inverses, since $\mathbb{Z}_p$ would not be a field. In particular, $x \in \mathbb{Z}_p$ is invertible if $\gcd(x, p) = 1$.

Power operation ($\mathbf{mod\ p \le 2^k}$). For the ciphertext of $x \in [0, 2^k)$ we show how to compute x^α for any power $\alpha \in \mathbb{Z}^+$ using only one HomLift. We perform a HomLift of $u := \mathbf{E}_s(x)$ and then use the function lookup,

$$\begin{aligned} f &: [0, 2^k) \to [0, 2^k)\\ f(i) &= i^\alpha \pmod{p}.\end{aligned}$$

```
            $\mathbb{F}_p$ **Inverse** : Inversemodp($\mathrm{E}_{\mathbf{s}}(x), p$) where $p \le 2^k$ is prime.
Input:      $bk = (C_0, \ldots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$: bootstrapping key,
            $\mathbf{u} \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{u} = E_{\mathbf{s}}(x)$ for $x \in (0, 2^k)$.
Output:     $\mathrm{E}_{\mathbf{s}}(x^{-1} \pmod{p})$
Step 1.     HomLift:
                $A := \mathrm{HomLift}(\mathbf{u}, bk)$
Step 3.     Function lookup:
                $F(x) := \sum_{i \in M} f(i) x^{iD_r}$, with $M = (0, 2^k)$
                and $f(i) = i^{-1} \pmod{p}$.
                $A := A \cdot F(x)$
Step 4.     Extract: suppose $A = (a(x), \sum_{i=0}^{m-1} b_i x^i) \in R_{m,Q}^2$. Set
                $\mathbf{a} := (\mathrm{Extract}(a(x), 0), b_0) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 5.     Modulus reduction:
                $\mathbf{c} := \lfloor r\mathbf{a}/Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.     Return $\mathbf{c}$.
```

Figure 8:

The procedure is giving in Figure 9 and takes one HomLift.

RELU. The RELU function (Rectified Linear Units) is one of the most common machine learning activation functions. For creating neural nets that can operate on encrypted data homomorphically this is a very desirable function. The RELU problem is to compute the following max function

$$x^+ := \max\{0, x\}.$$

There are potentially several domains that x may be chosen from. If we specify to $x \in (-2^k, 2^k) \subset \mathbb{Z}$ we can solve this. In particular, consider $x \in (-2^k, 2^k)$, then

$$x^+ := \begin{cases} 0 \text{ if } x \in (-2^k, 0) \\ x \text{ if } x \in [0, 2^k). \end{cases}$$

If we are given $\mathbf{E}_s(x)$ for $x \in (-2^k, 2^k)$, we can perform a HomLift of $u := \mathbf{E}_s(x)$ and use the lookup function $f(x) = x^+$, and the function encoding polynomial

$$F(x) = \sum_{i \in (-2^k, 2^k)} f(i) x^{iD_r}$$

to get $\mathbf{E}_s(x^+)$. The procedure is giving in Figure 10 and takes one HomLift.

5.5 Combining operations

In this paper, we do not get into the details of how to combine these operations when creating a circuit. We do point out that care will need to be taken when managing ciphertexts that store the higher and lower bits of an integer resulting from the operations of addition/multiplication in $\mathbb{Z}$.

```
            **Power (mod p ≤ 2^k)**: $\text{Powermodp}(\text{E}_{\mathbf{s}}(x), \alpha, p)$
Input:      $bk = (C_0, \ldots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$: bootstrapping key,
            $\mathbf{u} \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{u} = E_{\mathbf{s}}(x)$ for $x \in [0, 2^k)$. $p \leq 2^k$, $\alpha \in \mathbb{Z}^+$
Output:     $\text{E}_{\mathbf{s}}(x^{\alpha} \pmod{p})$.
Step 1.     HomLift:
                $A := \text{HomLift}(\mathbf{u}, bk)$
Step 3.     Function lookup:
                $F(x) := \sum_{i \in M} f(i) x^{iD_r}$, with $M = [0, 2^k)$
                and $f(i) = i^{\alpha} \pmod{p}$.
                $A := A \cdot F(x)$
Step 4.     Extract: suppose $A = (a(x), \sum_{i=0}^{m-1} b_i x^i) \in R_{m,Q}^2$. Set
                $\mathbf{a} := (\text{Extract}(a(x), 0), b_0) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 5.     Modulus reduction:
                $\mathbf{c} := \lfloor r\mathbf{a}/Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.     Return $\mathbf{c}$.
```

Figure 9:

```
            **RELU** : $\text{RELU}(\text{E}_{\mathbf{s}}(x))$
Input:      $bk = (C_0, \ldots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$: bootstrapping key,
            $\mathbf{u} \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{u} = E_{\mathbf{s}}(x)$ for $x \in (-2^k, 2^k)$.
Output:     $\text{E}_{\mathbf{s}}(x^+)$.
Step 1.     HomLift:
                $A := \text{HomLift}(\mathbf{u}, bk)$
Step 3.     Function lookup:
                $F(x) := \sum_{i \in M} f(i) x^{iD_r}$, with $M = (-2^k, 2^k)$
                and $f(i) = i^+$.
                $A := A \cdot F(x)$
Step 4.     Extract: suppose $A = (a(x), \sum_{i=0}^{m-1} b_i x^i) \in R_{m,Q}^2$. Set
                $\mathbf{a} := (\text{Extract}(a(x), 0), b_0) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 5.     Modulus reduction:
                $\mathbf{c} := \lfloor r\mathbf{a}/Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.     Return $\mathbf{c}$.
```

Figure 10:

```
            Mult in $\mathbb{Z}$ : MultinZ$(\mathrm{E}_{\mathbf{s}}(x_1), \mathrm{E}_{\mathbf{s}}(x_2))$
Input:      $bk = (C_0, \ldots, C_{n-1}) \in \left\{R_{m,Q}^{(2\ell)\times 2}\right\}^n$ : bootstrapping key,
            $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{v}_i = E_{\mathbf{s}}(x_i)$ for $x_1, x_2 \in [0, 2^k)$.
Output:     $\mathrm{E}_{\mathbf{s}}(y_0)$, $\mathrm{E}_{\mathbf{s}}(y_1)$ st $x_1 \cdot x_2 = y_0 + y_1 2^k$
Step 1.     Compute $\mathbf{u} := \mathbf{v}_1 + \mathbf{v}_2$, $\tilde{\mathbf{u}} := \mathbf{v}_1 - \mathbf{v}_2 \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 2.     HomLift:
                $A :=$ HomLift$(\mathbf{u}, bk)$,   $\tilde{A} :=$ HomLift$(\tilde{\mathbf{u}}, bk)$
Step 3.     Function lookup:
                $A_1 := A \cdot F_1(x)$,   $A_2 := \tilde{A} \cdot F_2(x)$
                $A_3 := A \cdot F_3(x)$,   $A_4 := \tilde{A} \cdot F_4(x)$
Step 4.     Extract: For $j = 1$ to 4, suppose $A_j = (a_j(x), \sum_{i=0}^{m-1} b_{ji} x^i)$.
            Set
                $\mathbf{a}_j :=$ (Extract$(a_j(x), 0), b_{j0}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 5.     Modulus reduction: For $j$ from 1 to 4 do
                $\mathbf{c}_j := \lfloor r\mathbf{a}_j / Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.     Compute $\bar{\mathbf{u}} := \mathbf{c}_1 - \mathbf{c}_2$,   $\ddot{\mathbf{u}} := \mathbf{c}_3 - \mathbf{c}_4$
Step 7.     HomLift:
                $\bar{A} :=$ HomLift$(\bar{\mathbf{u}}, bk)$,   $\ddot{A} :=$ HomLift$(\ddot{\mathbf{u}}, bk)$
Step 8.     Function lookup:
                $A_5 := \bar{A} \cdot F_5(x)$,   $A_6 := \ddot{A} \cdot F_6(x)$
Step 9.     Extract: For $j = 5, 6$ suppose $A_j = (a_j(x), \sum_{i=0}^{m-1} b_{ji} x^i)$. Set
                $\mathbf{a}_j :=$ (Extract$(a_j(x), 0), b_{j0}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 10.    Modulus reduction: For $j = 5, 6$
                $\mathbf{c}_j := \lfloor r\mathbf{a}_j / Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 11.    Compute $\hat{\mathbf{u}} := \mathbf{c}_6 - \mathbf{c}_5$,
Step 12.    HomLift:
                $\hat{A} :=$ HomLift$(\hat{\mathbf{u}}, bk)$,
Step 13.    Function lookup:
                $A_7 := \hat{A} \cdot F_7(x)$,
Step 14.    Extract: For $j = 7$, suppose $A_j = (a_j(x), \sum_{i=0}^{m-1} b_{ji} x^i)$. Set
                $\mathbf{a}_j :=$ (Extract$(a_j(x), 0), b_{j0}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$,
Step 15.    Modulus reduction: For $j = 7$ do
                $\mathbf{c}_j := \lfloor r\mathbf{a}_j / Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 16.    Return: $c_5 = \mathbf{E}_s(y_0)$ and $c_7 = \mathbf{E}_s(y_1)$.
            Lookup functions:
                $f_1 : [0, 2^{k+1}) \to [0, 2^k)$, $f_1(x) = \left\lfloor \frac{x^2 \pmod{2^{k+2}}}{4} \right\rfloor$.
                $f_2 : (-2^k, 2^k) \to [0, 2^k)$, $f_2(x) = \left\lfloor \frac{x^2 \pmod{2^{k+2}}}{4} \right\rfloor$.
                $f_3 : [0, 2^{k+1}) \to [0, p)$, $f_3(x) = \left(x(2^{-1})\right)^2 \pmod{p}$.
                $f_4 : (-2^k, 2^k) \to [0, p)$, $f_4(x) = \left(x(2^{-1})\right)^2 \pmod{p}$.
                $f_5 : (-2^k, 2^k) \to [0, 2^k)$, $f_5(x) = x \pmod{2^k}$.
                $f_6 : (-2^k, 2^k) \to [0, 2^k)$, $f_6(x) = x \pmod{2^k - 1}$.
                $f_7 = f_6$
```

Figure 11: Homomorphic integer multiplication of $x_1, x_2 \in [0, 2^k)$ with output stored in two ciphertexts: $x_1 \cdot x_2 = y_0 + y_1 2^k \in \mathbb{Z}$ where both y_0 and y_1 have k bits, MultinZ$(E_{\mathbf{s}}(x_1), E_{\mathbf{s}}(x_2)) = (E_{\mathbf{s}}(y_0), E_{\mathbf{s}}(y_1))$.

6 Fully homomorphic encryption scheme

6.1 Parameter Conditions

We shall assume that $k \in \mathbb{Z}^+$, $n \geq 1024$ is a power of 2, r is a power of 2, $m = r/2$, and

$$\ell = 2, \quad r \geq 2^{k+6}\sqrt{n}, \quad q \geq 2^7 rn.$$

For B and Q we need them to satisfy the assumptions of Theorem 5.5,

$$15(2^{2k+2})Br\tau_{bk}\sqrt{2\ell m} \leq Q < B^2.$$

This implies $15(2^{2k+2})r\tau_{bk}\sqrt{2\ell m} \leq B' < B$. We will also take $Q = B \cdot B'$ with both B, B' prime for Lemma 3.6. In practice to make the implementation more efficients, we will choose B, B' such that $r|(B-1)$ and $r|(B'-1)$. This will allow more efficient FFTs when parallelizing using the CRT. Let

$$R_{n,r} = \mathbb{Z}[x]/(x^n+1, r), \quad R_{n,q} = \mathbb{Z}[x]/(x^n+1, q), \quad R_{m,Q} = \mathbb{Z}[x]/(x^m+1, Q),$$

$$D_r = \lfloor r/2^{k+2} \rfloor, \quad D_q = \lfloor q/2^{k+2} \rfloor, \quad D_Q = \lfloor Q/2^{k+2} \rfloor.$$

Each user generates a secret key $s \in \{0,1\}^n$ and a public key as described in Section 4.
Bootstrapping key (standard).

A corresponding bootstrapping key $bk = (C_0, C_1, \ldots, C_{n-1})$ is generated as follows. For each $0 \leq i \leq n-1$ do the following:

- pick $a_{ji}(x) \in R_{m,Q}$ uniform random and independent, for $1 \leq j \leq 4$,
- pick $e_{ji}(x) \in R_m$ bounded uniform random and independent with

$$\tau_{bk} = ||e_{ji}(x)||_\infty \leq 2\sqrt{n}, \qquad 1 \leq j \leq 4,$$

- Compute $b_{ji}(x) := a_{ji}(x)s(x) + e_{ji}(x) \bmod (x^m+1, Q)$, for $1 \leq j \leq 4$,
- Set

$$C_i := \begin{pmatrix} a_{1i}(x) & b_{1i}(x) \\ a_{2i}(x) & b_{2i}(x) \\ a_{3i}(x) & b_{3i}(x) \\ a_{4i}(x) & b_{4i}(x) \end{pmatrix} + s_i G \bmod Q.$$

Bootstrapping key (smaller). Now we introduce some size optimizations for storing the boostrapping key. We do so by storing each of the four RLWE ciphertexts that correspond to s_i for each $0 \leq i \leq n-1$ in a more compact form. Let $t := \lceil \log_2(Q) \rceil - 1$, hence $2^t < Q \leq 2^{t+1}$, and assume we have a PRG $P : \{0,1\}^m \rightarrow \{0,1\}^{4tm-2t}$. Sample $u_i \in \{0,1\}^m$ uniform random and then use the PRG to expand it to represent the polynomials $a_{3i}(x), a_{4i}(x)$ and all the terms of $a_{1i}(x), a_{2i}(x)$ except their constant terms, call these $\tilde{a}_{1i}(x), \tilde{a}_{2i}(x)$. Randomly sample the constant terms $\tilde{a}_{1i0}, \tilde{a}_{2i0}$ separately from $\mathbb{Z}_Q$. Then let

$$a_{1i}(x) := \tilde{a}_{1i}(x) + \tilde{a}_{1i0} + s_i$$

and

$$a_{2i}(x) := \tilde{a}_{2i}(x) + \tilde{a}_{2i0} + s_i B$$

with constant terms denoted as

$$a_{1i0} := \tilde{a}_{1i0} + s_i$$

and

$$a_{2i0} := \tilde{a}_{2i0} + s_i B.$$

Now using the bootstrapping key encryption subroutine in Figure 12 (proof similar to private key encryption) to round the RLWE ciphertexts, we get that the bootstrapping key can be defined as

$$C_i := \begin{pmatrix} a_{1i}(x) & \mathrm{BT}_s(a_{1i}(x), 0) \\ a_{2i}(x) & \mathrm{BT}_s(a_{2i}(x), 0) \\ a_{3i}(x) & \mathrm{BT}_s(a_{3i}(x), s_i) \\ a_{4i}(x) & \mathrm{BT}_s(a_{4i}(x), s_i B) \end{pmatrix} \bmod Q.$$

But we only need to store the seed u and the constant terms a_{1i0} and a_{2i0}. Thus we can recreate C_i from the following

$$\{u, a_{1i0}, a_{2i0}, \mathrm{BT}_s(a_{1i}(x), 0), \mathrm{BT}_s(a_{2i}(x), 0), \mathrm{BT}_s(a_{3i}(x), s_i), \mathrm{BT}_s(a_{4i}(x), s_i B)\}$$

which is $m + 2t + 4(t-5)m$ bits. Thus the entire boostrapping key C_i, $0 \le i \le n-1$, can be recreated from $n(m + 2t + 4(t-5)m)$ bits.

	Bootstrapping key Subroutine : $\mathrm{BT}_{\mathbf{s}}(a(x), m(x))$
Input:	$s(x) = \sum_{i=0}^{n-1} s_i x^i$ where $s_i \in \{0,1\}$, an n-bit secret key,
	$m(x)$ message to encrypt, $a(x)$ uniform random
	$t := \lceil \log_2(Q) \rceil - 1$, hence $2^t < Q \le 2^{t+1}$,
Output:	$\mathbf{v} \in \{0,1\}^{(t-5)m}$
Step 1.	Pick $w(x) \in R_m$ uniform randomly with $\|w(x)\|_\infty \le \sqrt{n} = 2^6$, and
	$b_1(x) := a(x)s(x) + w(x) + m(x) \bmod (x^m + 1, Q)$
	(so that each coefficient of $b_1(x)$ is between 0 and $Q-1$).
Step 2.	Taking the highest $t-5$ bits for each coefficient of $b_1(x)$:
	$b(x) := \lfloor b_1(x)/2^5 \rfloor$.
	Let $\mathbf{v} \in (\{0,1\}^{t-5})^m$ denote the bit representation of $b(x)$.
Step 2.	Return $\mathbf{v}$.

Figure 12:

Lemma 6.1. *Let $(a(x), b(x)) \in R_{m,Q}^2$ be as computed in Figure 12, then there exists $w_3(x) \in R_m$ with $||w_3(x)||_\infty < 2\sqrt{n} = 2^7$ so that*

$$2^5 b(x) - s(x)a(x) \equiv w_3(x) + m(x)B \ \textit{mod} \ (x^m + 1, Q).$$

Proof. By Step 1, since the coefficients of $b_1(x)$ are between 0 and $Q-1$, we have

$$b_1(x) = 2^5 b(x) + b_0(x)$$

for some $b_0 \in R_m$ with $||b_0(x)||_\infty < 2^6$. By Step 2, we have

$$2^5 b(x) - s(x)a(x) \equiv -b_0(x) + w(x) + m(x)B \bmod (x^m + 1, Q).$$

Thus,

$$|| - b_0(x) + w(x)||_\infty \le ||b_0(x)||_\infty + ||w(x)||_\infty < \sqrt{n} + \sqrt{n} = 2\sqrt{n}.$$

Therefore, the lemma holds with $w_3(x) = w(x) - b_0(x)$. □

6.2 Suggested Parameters Sizes

The main parameters affecting performance and security that need to be fixed in choosing concrete parameters are n and k. To meet the necessary security constraints analyzed more fully in the next section we will need $n \geq 2^{12}$. With $n = 2^{12}$ we can consider several possible values of k. In Figures 13, we list some parameters that satisfy the conditions above. The row for c_s gives the ciphertext expansion ratio under private-key encryption, that is, the bit size of a ciphertext of an n-bit message divided by n; the row for c_{pk} gives the ciphertext expansion ratio under public-key encryption. The row for bk indicates the bit size of bootstrapping keys. Note that the size of the boostrapping key grows exponentially as k increases, and this is a reason in practice to choose a small k.

n	2^{12}	2^{12}	2^{12}	2^{12}	2^{12}
k	1	2	3	4	5
$r = 2^{k+6}\sqrt{n}$	2^{13}	2^{14}	2^{15}	2^{16}	2^{17}
$m = r/2$	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
$q \geq 2^7 rn$	32 bits	33 bits	34 bits	35 bits	36 bits
Q	71 bits	78 bits	85 bits	92 bits	99 bits
c_s	7	4	3	2.5	2.2
c_{pk}	20	11	8	6.5	5.6
bk transmit	1128 MB	2491 MB	5452 MB	11844 MB	25,568 MB
λ_1 BKZ	1000+	1000+	1000+	1000+	1000+
λ_2 BKZ	838	798	762	728	696
λ_3 BKZ	249	219	196	177	162

Figure 13: Suggested Parameters: The row of c_s gives the cipher expansion under private-key encryption, the row of c_{pk} is for cipher expansion for public-key encryption, and the row bk is for the size of bootstrapping keys. The row λ_i is the security estimate for ciphertexts of Type i in Section 7

Failure probability The probability of failure has been bounded at each step, the largest the probability of failure has grown to is less than 2^{-140}.

7 Security analysis

In this section, we give a brief analysis of the security of our homomorphic encryption scheme. Although the LWE and RLWE problems have hardness tied to worst case lattice problems, we still need to estimate the concrete complexity of all current attacks for our proposed parameters. This is an active and ongoing important area of research, particularly in light of the NIST post-quantum cryptography process.

In our scheme, according to Figure 13, n is a power of 2 and we have RLWE ciphertexts over $\mathbb{Z}_q$ for three choices of q:

Type 1. $q = r = 2^{k+6}\sqrt{n}$ and the error size is bounded by $4\sqrt{n}$: corresponding to ciphertexts in $R_{n,q}^2$ from private-key and public-key encryptions of the original data (see Lemmas 4.1 and 4.2);

Type 2. $q \approx 2^7 rn$ and the error size is bounded by $D_q/(512n) \geq 4\sqrt{n}$: corresponding to the public key $pk = (k_0(x), k_1(x)) \in R_{n,q}^2$;

Type 3. $q = Q \approx 2^{7k+32} n^{1.5} \tau_{bk}^2$ and the error size is bounded by $\tau_{bk} = 2\sqrt{n}$: corresponding to a bootstrapping key $C_i \in R_{m,Q}^2$, $1 \leq i \leq n$ and the extractions into $\mathbb{Z}_Q^n \times \mathbb{Z}_Q$.

Number theoretic attacks.

The RLWE problem over the above two rings and more general rings of the form $\mathbb{Z}[x]/(f(x), q)$ have been studied in [30, 31, 18, 19, 20, 21] using algebraic number theory. They present several attacks that show many weak instances of the general rings. However, their attacks do not apply to the two rings used by our scheme. In fact, one of the main ideas of the attacks is to test if $f(x)$ modulo q has a factor of small degree and the roots of the factor have a small multiplicative order. For our two rings, when n is a power of 2, $x^n + 1$ has all roots of order $2n$ modulo any $q > 2$. Similarly for $x^m + 1$. Hence the number theoretic attacks can not be applied effectively to our rings.

Lattice basis reduction attacks. The most powerful attacks on RLWE/LWE is to use the lattice basis reduction algorithm (LLL) due to Lenstra, Lenstra and Lovasz (1982, [42]); see [50] for its practical performance and [48, 49] for its recent improvements. There is also a BKZ variation [57, 22], which uses SVP oracles [47, 36, 58, 59, 40, 41, 10]. There are several approaches that can reduce LWE problems over $\mathbb{Z}_q$ to lattice problems over $\mathbb{Z}$, including the SIS method [1, 46, 43], the BKW method [12, 3, 4, 5, 28, 35, 39], the bounded distance decoding (BBD) method [38, 43, 44, 8].

The paper by Albrecht, Player and Scott [6] gives a nice survey on these methods and give concrete complexity analysis; they also have an LWE estimator (`bitbucket.org/malb/lwe-estimator`) that is also used to give the security estimates for the HE Standards document [2]. These LWE estimates are based on discrete Gaussian error and so do not apply directly to our bounded uniform distribution. However, the known lattice reduction attacks do not make use of the particular error distribution; but rather, their performance depends on the standard deviation of the error distribution. Thus, it has become common to use the LWE estimator even when the error distribution is not Gaussian, e.g. in the NewHope Round 2 NIST submission the error is binomial.

We use a bounded uniform distribution with error bound $\tau_{bk} = 2\sqrt{n}$ for the boostrapping key and $\tau = 4\sqrt{n}$ for all other ciphertexts. The variance of a discrete bounded uniform distribution on $[a, b]$ is

$$\frac{(b-a+1)^2 - 1}{12}.$$

Thus the variance our our τ bounded error distribution is $\frac{(2\tau+1)^2-1}{12}$. Our code for using the estimator for our parameters follows.

```
load("https://bitbucket.org/malb/lwe-estimator/raw/HEAD/estimator.py")
#For estimating the security of Type 1 ciphers
n = 2^12; q = 2^13; #q=2^{13,14,15,16,17}
tau = 4 * sqrt(n); var = ((2*tau +1)^2 - 1 )/12; stddev = sqrt(var); alpha =
    alphaf(sigmaf(stddev), q)
_ = estimate_lwe(n, alpha, q, reduction_cost_model=BKZ.sieve)

#For estimating the security of Type 2 ciphers
n = 2^12; q = 2^32; # q = 2^{32,33,34,35,36}
tau = 4 * sqrt(n); var = ((2*tau +1)^2 - 1 )/12; stddev = sqrt(var); alpha =
    alphaf(sigmaf(stddev), q)
_ = estimate_lwe(n, alpha, q, reduction_cost_model=BKZ.sieve)

#For estimating the security of Type 3 ciphers
n = 2^12; q = 2^100; # q = 2^{71,78,85,92,99}
tau = 2 * sqrt(n); var = ((2*tau +1)^2 - 1 )/12; stddev = sqrt(var); alpha =
    alphaf(sigmaf(stddev), q)
_ = estimate_lwe(n, alpha, q, reduction_cost_model=BKZ.sieve)
```

8 Conclusions

We presented a fully homomorphic encryption scheme with a small cipher expansion and k-bit arithmetic operations. The scheme is suitable for practical applications in distributed networks of computers, including IoTs, blockchains and cloud servers and can protect function privacy and can be used in many applications including outsourced computing. On the theoretical side, this is the first FHE scheme to achieve asymptotically a ciphertext expansion factor of 1, and to our knowledge demonstrate k-bit multiplication done in the integers as in Figure 11. Moreover, through more study of sub-Gaussian properties, we have been able to avoid using an Independence Heuristic as in the Chillotti et al. [25] TFHE schemes.

References

[1] M. Ajtai, *Generating hard instances of lattice problems (extended abstract)*, Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '96, ACM, 1996, pp. 99–108.

[2] Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Jeffrey Hoffstein, Kristin Lauter, Satya Lokam, Daniele Micciancio, et al., *Homomorphic encryption standard*, (2018).

[3] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret, *On the complexity of the BKW algorithm on LWE*, Des. Codes Cryptography **74** (2015), no. 2, 325–354.

[4] Martin R. Albrecht, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret, *Lazy modulus switching for the BKW algorithm on LWE*, Public-Key Cryptography – PKC 2014 (Berlin, Heidelberg) (Hugo Krawczyk, ed.), Springer Berlin Heidelberg, 2014, pp. 429–445.

[5] Martin R. Albrecht, Robert Fitzpatrick, and Florian Göpfert, *On the efficacy of solving LWE by reduction to unique-SVP*, Information Security and Cryptology – ICISC 2013

(Cham) (Hyang-Sook Lee and Dong-Guk Han, eds.), Springer International Publishing, 2014, pp. 293–310.

[6] Martin R Albrecht, Rachel Player, and Sam Scott, *On the concrete hardness of learning with errors*, Journal of Mathematical Cryptology **9** (2015), no. 3, 169–203.

[7] Riham AlTawy, Raghvendra Rohit, Morgan He, Kalikinkar Mandal, Gangqiang Yang, and Guang Gong, *sliscp: Simeck-based permutations for lightweight sponge cryptographic primitives*, Selected Areas in Cryptography – SAC 2017 (Cham) (Carlisle Adams and Jan Camenisch, eds.), Springer International Publishing, 2018, pp. 129–150.

[8] Shi Bai and Steven D. Galbraith, *Lattice decoding attacks on binary LWE*, pp. 322–337, Springer International Publishing, Cham, 2014.

[9] Abhishek Banerjee, Chris Peikert, and Alon Rosen, *Pseudorandom functions and lattices*, Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2012, pp. 719–737.

[10] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven, *New directions in nearest neighbor searching with applications to lattice sieving*, Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms (Philadelphia, PA, USA), SODA '16, Society for Industrial and Applied Mathematics, 2016, pp. 10–24.

[11] Jean-François Biasse and Luis Ruiz, *Fhew with efficient multibit bootstrapping*, Progress in Cryptology – LATINCRYPT 2015 (Cham) (Kristin Lauter and Francisco Rodríguez-Henríquez, eds.), Springer International Publishing, 2015, pp. 119–135.

[12] Avrim Blum, Adam Kalai, and Hal Wasserman, *Noise-tolerant learning, the parity problem, and the statistical query model*, J. ACM **50** (2003), no. 4, 506–519. MR 2146884

[13] Zvika Brakerski, *Fully homomorphic encryption without modulus switching from classical gapsvp*, Proceedings of the 32Nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012 - Volume 7417 (New York, NY, USA), Springer-Verlag New York, Inc., 2012, pp. 868–886.

[14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan, *(Leveled) fully homomorphic encryption without bootstrapping*, ACM Trans. Comput. Theory **6** (2014), no. 3, Art. 13, 36. MR 3255281

[15] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé, *Classical hardness of learning with errors*, STOC'13—Proceedings of the 2013 ACM Symposium on Theory of Computing, ACM, New York, 2013, pp. 575–584. MR 3210819

[16] Zvika Brakerski and Vinod Vaikuntanathan, *Efficient fully homomorphic encryption from (standard) LWE*, SIAM Journal on Computing **43** (2014), no. 2, 831–871.

[17] Sergiu Carpov, Malika Izabachène, and Victor Mollimard, *New techniques for multi-value homomorphic evaluation and applications.*, IACR Cryptology ePrint Archive **2018** (2018), 622.

[18] Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren, *Provably weak instances of Ring-LWE revisited*, Proceedings of the 35th Annual International Conference on Advances in Cryptology — EUROCRYPT 2016 - Volume 9665 (New York, NY, USA), Springer-Verlag New York, Inc., 2016, pp. 147–167.

[19] Hao Chen, Kristin Lauter, and Katherine E. Stange, *Security considerations for galois non-dual RLWE families*, Cryptology ePrint Archive, Report 2016/193, 2016, https://eprint.iacr.org/2016/193.

[20] Hao Chen, Kristin E. Lauter, and Katherine E. Stange, *Attacks on the search-RLWE problem with small error*, Cryptology ePrint Archive, Report 2015/971, 2015, https://eprint.iacr.org/2015/971.

[21] Yao Chen, Benjamin M. Case, Shuhong Gao, and Guang Gong, *Error analysis of weak Poly-LWE instances*, Cryptography and Communications, 2017, pp. 411–426.

[22] Yuanmi Chen and Phong Q. Nguyen, *BKZ 2.0: better lattice security estimates*, Advances in cryptology—ASIACRYPT 2011, Lecture Notes in Comput. Sci., vol. 7073, Springer, Heidelberg, 2011, pp. 1–20. MR 2934994

[23] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène, *Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds*, Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22, Springer, 2016, pp. 3–33.

[24] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène, *Improving TFHE: faster packed homomorphic operations and efficient circuit bootstrapping*, Cryptology ePrint Archive, Report 2017/430, 2017, https://eprint.iacr.org/2017/430.

[25] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene, *Tfhe: Fast fully homomorphic encryption over the torus.*, IACR Cryptology ePrint Archive **2018** (2018), 421.

[26] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène, *Tfhe: Fully homomorphic encryption over the torus*, Poster at Third Homomorphic Encryption Standards workshop, 2018.

[27] ———, *TFHE: Fast fully homomorphic encryption library*, August 2016, https://tfhe.github.io/tfhe/.

[28] Alexandre Duc, Florian Tramèr, and Serge Vaudenay, *Better algorithms for LWE and LWR*, Advances in Cryptology – EUROCRYPT 2015 (Berlin, Heidelberg) (Elisabeth Oswald and Marc Fischlin, eds.), Springer Berlin Heidelberg, 2015, pp. 173–202.

[29] Léo Ducas and Daniele Micciancio, *FHEW: bootstrapping homomorphic encryption in less than a second*, Advances in cryptology—EUROCRYPT 2015. Part I, Lecture Notes in Comput. Sci., vol. 9056, Springer, Heidelberg, 2015, pp. 617–640. MR 3344940

[30] Kirsten Eisenträger, Sean Hallgren, and Kristin Lauter, *Weak instances of PLWE*, Selected Areas in Cryptography – SAC 2014 (Cham) (Antoine Joux and Amr Youssef, eds.), Springer International Publishing, 2014, pp. 183–194.

[31] Yara Elias, Kristin E. Lauter, Ekin Ozman, and Katherine E. Stange, *Provably weak instances of Ring-LWE*, Advances in Cryptology – CRYPTO 2015 (Berlin, Heidelberg) (Rosario Gennaro and Matthew Robshaw, eds.), Springer Berlin Heidelberg, 2015, pp. 63–92.

[32] Shuhong Gao, *Efficient fully homomorphic encryption scheme*, Cryptology ePrint Archive, Report 2018/637, 2018, `https://eprint.iacr.org/2018/637`.

[33] Craig Gentry, *Fully homomorphic encryption using ideal lattices*, Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009, 2009, pp. 169–178.

[34] Craig Gentry, Amit Sahai, and Brent Waters, *Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based*, Advances in Cryptology–CRYPTO 2013, Springer, 2013, pp. 75–92.

[35] Qian Guo, Thomas Johansson, and Paul Stankovski, *Coded-BKW: Solving LWE using lattice codes*, pp. 23–42, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[36] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé, *Algorithms for the shortest and closest lattice vector problems*, Coding and Cryptology (Berlin, Heidelberg) (Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, eds.), Springer Berlin Heidelberg, 2011, pp. 159–190.

[37] Wassily Hoeffding, *Probability inequalities for sums of bounded random variables*, Journal of the American statistical association **58** (1963), no. 301, 13–30.

[38] Ravi Kannan, *Minkowski's convex body theorem and integer programming*, Math. Oper. Res. **12** (1987), no. 3, 415–440.

[39] Paul Kirchner and Pierre-Alain Fouque, *An improved BKW algorithm for LWE with applications to cryptography and lattices*, pp. 43–62, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[40] Thijs Laarhoven and Benne de Weger, *Faster sieving for shortest lattice vectors using spherical locality-sensitive hashing*, Progress in Cryptology – LATINCRYPT 2015 (Cham) (Kristin Lauter and Francisco Rodríguez-Henríquez, eds.), Springer International Publishing, 2015, pp. 101–118.

[41] Thijs Laarhoven, Michele Mosca, and Joop van de Pol, *Finding shortest lattice vectors faster using quantum search*, Des. Codes Cryptography **77** (2015), no. 2-3, 375–400.

[42] A. K. Lenstra, H. W. Lenstra, and L. Lovász, *Factoring polynomials with rational coefficients*, Mathematische Annalen **261** (1982), no. 4, 515–534.

[43] Richard Lindner and Chris Peikert, *Better key sizes (and attacks) for LWE-based encryption*, Topics in cryptology—CT-RSA 2011, Lecture Notes in Comput. Sci., vol. 6558, Springer, Heidelberg, 2011, pp. 319–339.

[44] Mingjie Liu and Phong Q. Nguyen, *Solving BDD by enumeration: an update*, Topics in cryptology—CT-RSA 2013, Lecture Notes in Comput. Sci., vol. 7779, Springer, Heidelberg, 2013, pp. 293–309. MR 3082022

[45] Vadim Lyubashevsky, Chris Peikert, and Oded Regev, *On ideal lattices and learning with errors over rings*, Advances in cryptology—EUROCRYPT 2010, Lecture Notes in Comput. Sci., vol. 6110, Springer, Berlin, 2010, pp. 1–23. MR 2660480

[46] Daniele Micciancio and Oded Regev, *Lattice-based cryptography*, pp. 147–191, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[47] Daniele Micciancio and Panagiotis Voulgaris, *Faster exponential time algorithms for the shortest vector problem*, Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010, 2010, pp. 1468–1480.

[48] Daniele Micciancio and Michael Walter, *Practical, predictable lattice basis reduction*, Advances in cryptology—EUROCRYPT 2016. Part I, Lecture Notes in Comput. Sci., vol. 9665, Springer, Berlin, 2016, pp. 820–849. MR 3516393

[49] Arnold Neumaier and Damien Stehlé, *Faster LLL-type reduction of lattice bases*, Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2016, Waterloo, ON, Canada, July 19-22, 2016, 2016, pp. 373–380.

[50] Phong Q. Nguyen and Damien Stehlé, *LLL on the average*, Algorithmic Number Theory, 7th International Symposium, ANTS-VII, Berlin, Germany, July 23-28, 2006, Proceedings (Florian Hess, Sebastian Pauli, and Michael E. Pohst, eds.), Lecture Notes in Computer Science, vol. 4076, Springer, 2006, pp. 238–256.

[51] National Institute of Standards and Technology, *FIPS PUB 202 SHA-3 standard: Permutation-based hash and extendable-output functions*, 2015.

[52] Oded Regev, *On lattices, learning with errors, random linear codes, and cryptography*, Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005, 2005, pp. 84–93.

[53] Oded Regev, *On lattices, learning with errors, random linear codes, and cryptography*, J. ACM **56** (2009), no. 6, 34:1–34:40.

[54] Philippe Rigollet, *18. s997: High dimensional statistics*, Lecture Notes), Cambridge, MA, USA: MIT Open-CourseWare (2015).

[55] R L Rivest, L Adleman, and M L Dertouzos, *On data banks and privacy homomorphisms*, Foundations of Secure Computation, Academia Press (1978), 169–179.

[56] Anonymous SameAuthors, *A note on sub-gaussians*, Included in supplemental material, to appear concurrently, 2019.

[57] C. P. Schnorr and M. Euchner, *Lattice basis reduction: Improved practical algorithms and solving subset sum problems*, Mathematical Programming **66** (1994), no. 1, 181–199.

[58] Xiaoyun Wang, Mingjie Liu, Chengliang Tian, and Jingguo Bi, *Improved Nguyen-Vidick heuristic sieve algorithm for shortest vector problem*, Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2011, Hong Kong, China, March 22-24, 2011, 2011, pp. 1–9.

[59] Feng Zhang, Yanbin Pan, and Gengran Hu, *A three-level sieve algorithm for the shortest vector problem*, Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers, 2013, pp. 29–47.