EFFICIENT FULLY HOMOMORPHIC ENCRYPTION SCHEME

SHUHONG GAO

Abstract. Since Gentry discovered in 2009 the first fully homomorphic encryption scheme, the last few years have witnessed dramatic progress on designing more efficient homomorphic encryption schemes, and some of them have been implemented for applications. The main bottlenecks are in bootstrapping and large cipher expansion (the ratio of the size of ciphertexts to that of messages). 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) to 13ms. This paper presents a compact fully homomorphic encryption scheme that has the following features: (a) its cipher expansion is 6 with private-key encryption and 20 with public-key encryption; (b) all ciphertexts after any number (unbounded) of homomorphic bit operations have the same size and are always valid with the same error size; (c) its security is based on the LWE and RLWE problems (with binary secret keys) and the cost of breaking the scheme by the current approaches is at least 2^{160} bit operations. The scheme protects function privacy and provides a simple solution for secure two-party computation and zero knowledge proof of any language in NP.

Keywords. Fully homomorphic encryption, data security, lattices, learning with errors (LWE) problem, ring learning with errors (RLWE) problem, NP problems, secure two-party computation, zero knowledge proof, fast Fourier transforms (FFT).

1. Introduction

As cloud computing, internet of things (IoT) and blockchain technology become increasingly prevalent, there is an urgent need to protect the privacy of massive volumes of sensitive data collected or stored in computer networks or cloud servers, as many of the networks or servers can be vulnerable to external and internal threats such as malicious hackers or curious insiders. In the case of blockchain technology, it is the privacy issue that prevents widespread enterprise adoption of blockchains, despite the transparency and immutability that the technology offers. The traditional encryption schemes can provide privacy protection of data but do not allow for the performance of analytics on encrypted data without decryption first. The Holy-Grail of cryptography is to have a practical encryption scheme that has the following properties:

(a) encrypted data can be stored anywhere (e.g. untrusted clouds, blockchains, or personal computers at home or at a hacker's control);
(b) any third party (including cloud servers, hackers, miners or insiders) can perform searching or analytics of an arbitrary function on the encrypted data to get search results in encrypted form, however, only the data owner (who has the secret decoding key) can decode the encrypted search results;

Date: May 21, 2018.

The work presented in this paper was partially supported by the National Science Foundation under grants CCF-1407623, DMS-1403062 and DMS-1547399. Thanks go to the Center of Applied Mathematics, Tianjin University, China, for hosting the author in Summer 2017, and to American Institute of Mathematics (AIM), San Jose, for hosting the author in several SQuaRE meetings in the last few years with Qi Cheng, J. Maurice Rojas and Daqing Wan.

(c) an adversary can access all the encrypted data and use all the available computing powers in the world but still can not compute any information of the original data in reasonable time (say 100 years).

An encryption scheme having all the three properties is called a fully homomorphic encryption (FHE) scheme. If (b) is satisfied only for a class of searches (but not all possible searches), then it is called a somewhat homomorphic encryption (SHE) scheme. In comparison, traditional cryptosystems (e.g. RSA, AES, elliptic curve cryptosystems, etc [55]) have properties (a) and (c), but not (b) which is the most challenging part of designing an FHE scheme.

The idea of homomorphic encryption was first proposed in 1978 by Rivest, Adleman and Dertouzos [99]. Only in 2009 was the breakthrough discovery made by Gentry [60] for the first fully homomorphic encryption scheme. His work has inspired an explosive surge of research on homomorphic encryption schemes, and the design blueprint of his original scheme has been serving as a prototype for designing more efficient schemes since then. In Gentry's scheme, random noise is added to ciphertexts (for security reason), the noise grows quickly when one performs homomorphic operations on the ciphertexts, and the size of the new ciphertexts may grow as well. Only a limited number of homomorphic operations can be applied before the noise gets too big to destroy the message encoded in the ciphertexts. Gentry proposes the novel idea of performing homomorphic decoding and bootstrapping to reduce the noise size, then more homomorphic operations can be performed, and continue in this fashion (of bootstrapping whenever needed), any number of homomorphic operations can be performed as needed for an arbitrary search function on the encrypted ciphertexts, hence yielding a fully homomorphic encryption scheme.

To design a practical homomorphic encryption scheme, one has to solve three problems:

(i) **Cipher expansion problem**. The size ratio of the ciphertexts vs the original data must be small so that communications and storage are not too expensive, and the size of new ciphertexts obtained under homomorphic operations should be independent of the complexity of every search function;

(ii) **Time efficiency problem**. The cost of homomorphic computing of an arbitrary function f should be proportional to the complexity of f itself, and the overhead factor should only depend on the security parameter λ;

(iii) **Security problem**. The security of the scheme must be based on hard mathematical problems, and for a scheme deigned for a given security parameter λ (say $\lambda = 120, 160$, or 200), the cost of breaking the scheme should be at least 2^λ (in bit operations).

On the security problem, Regev [97, 98] proved in 2005 an important theorem on the hardness of the learning with error (LWE) problem. Thanks to Regev's work and [25, 88], homomorphic encryption schemes based on the LWE and RLWE problems now have a solid security foundation.

In the last nine years or so, dramatic progress on the time efficiency problem has been made by Gentry and his colleagues and many other researchers around the world. In fact, three generations of homomorphic encryption schemes were developed, each of them has its advantages and disadvantages. The first generation is based on ideal lattices and approximate gcd problem of integers, see [60, 48, 101, 61, 62, 43, 63, 64, 38, 102]. The second generation is based on LWE and RLWE problems, and several novel techniques are developed, including modulus reduction, key switch and re-linearization, for mitigating noise growth; see [21, 27, 26, 22, 23, 54, 73]. The BGV scheme [22] is implemented in HElib [72], and the FV scheme [54] is implemented in SEAL [33]. There is also another scheme combining the idea from NTRU [75] and the RLWE problem, see [103, 86, 20]. The third generation refers to the GSW scheme [65, 28] which is based on RLWE and approximate

eigenvalues; also a novel technique called flattening is invented to better control noise growth. Though bootstrapping is still prohibitively slow, these schemes can be implemented as leveled schemes to compute functions that have designed depth of multiplications, and they can handle patches of many bits simultaneously. The SEAL implementation [33] is already being tested for private genome analysis [78]. In all these schemes, however, the ciphertext expansion is still too large, ranging from a few hundreds to tens of thousands (depending on the designed depth).

On bootstrapping, a recent breakthrough is made by Ducas and Micciancio (2015 [51]), who use the GSW scheme [65] 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 [39], 2017[40]), who reduce the bootstrapping time down to 13ms (for one homomorphic bit operation). On cipher expansion, it is stated in [39] that their ciphertext size is still 400,000 times that of the original data, and in [40], this expansion factor is claimed to be reduced to 64 (for message block length 500, but no detail is given).

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

(1) its cipher expansion is 6 under private-key encryption and $10+\log_2(n)$ under public-key encryption where n (a power of 2) denotes the message block length, all ciphertexts are computed modulo r where $r = 16n$, and the noise size is bounded by $n-1$;

(2) its bootstrapping procedure needs only a bootstrapping key and does the following: for any two LWE ciphers $E_{\mathbf{s}}(x_1)$ and $E_{\mathbf{s}}(x_2) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ with noise size bounded by $n-1$ where $x_1, x_2 \in \{0,1\}$, it produces three random LWE ciphers:

$$E_{\mathbf{s}}(x_1 \wedge x_2), \quad E_{\mathbf{s}}(x_1 \vee x_2), \quad E_{\mathbf{s}}(x_1 \oplus x_2) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$$

with noise size still bounded by $n-1$ (no failure at all), and the total time for the bootstrapping procedure (for $n = 512$) is estimated to be about 130ms, namely 10 times that of [40];

(3) its security is based on the LWE and RLWE problems [97, 98, 88] (with binary secret keys) and, for any message block length $n \geq 512$, the cost of breaking the scheme by the current approaches is at least 2^{160} bit operations.

Note that, in the bootstrapping of [51, 39, 40], a new LWE cipher produced by bootstrapping may be invalid, with probability about 2^{-33} (for $n = 500$). That is a very small probability, so useful for computing many functions, however, it can not be applied to functions that require more than 2^{33} bit operations (unless increasing n). In our scheme, the LWE ciphers after bootstrapping are always in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size bounded by $n-1$, hence always valid (no failure at all). This means that one can perform bootstrapping any number (unbounded) of times and all the new LWE ciphers are still in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ with the same error size $n-1$. Due to this compactness, the computed ciphertexts do not leak any information on which function is computed, hence the scheme automatically provides function privacy.

Another important feature of our FHE scheme is that the ciphertexts resulted from homomorphic computing are independent random. This is extremely important in designing protocols for secure multi-party computation and for zero knowledge proof. We shall give more details about this in Section 6.

Organization of the paper. In Section 2 we present notations, basic concepts and techniques. We describe the LWE and RLWE problems and the related ciphers. We also present modulus reduction, randomized flattening, external product, etc., 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 3, we present our bootstrapping procedure, which is

modified from those of [51, 39]. In Section 4, we present our encryption schemes and show how a pseudocode for a general function f can be computed homomorphically, particularly how to handle "if-statement", "for-loop" and "while-loop". We also show how to pack LWE ciphers into RLWE ciphers, hence reduce the ciphertext size of the final result. In Section 5, we present security analysis of our scheme, and in Section 6, we mention some use cases, including secure two-party computation and zero knowledge proof of any language in NP.

2. Basic concepts and techniques

2.1. **Cyclotomic rings and norms.** Let q be a positive integer and $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$, the ring of integers modulo q. For an integer $n \geq 1$, let

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

where $\mathbb{R}$ denotes the field of real numbers and (x^n+1, q) denotes the ideal of $\mathbb{Z}[x]$ generated by x^n+1 and q, namely

$$(x^n+1, q) = \{u(x)(x^n+1) + v(x)q : \ u(x), v(x) \in \mathbb{Z}[x]\}.$$

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 ℓ-norm and ∞-norm as

$$\|f(x)\|_\ell = \left(\sum_{i=0}^{d} |f_i|^\ell\right)^{1/\ell}, \quad \|f(x)\|_\infty = \max_{0\leq i\leq d} |f_i|.$$

We shall use the cases when $\ell = 1, 2$ and ∞. However, for $f(x) = \sum_{i=0}^{d} f_i x^i \in \mathbb{R}_n$, we define its norm as follows. First find the unique $h(x) \in \mathbb{R}[x]$ so that $\deg g(x) < n$ 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 define

$$\|f(x)\| = \|h(x)\|,$$

where the norm $||\cdot||$ stands for any one of the ℓ-norm or ∞-norm (similarly below). For example, in $\mathbb{R}_4 = \mathbb{R}[x]/(x^4+1)$, we have $\|10x^4 + x + 8\|_\infty = \|x-2\|_\infty = 2$ and

$$\|(x-1)^4\|_1 = \|x^4 - 4x^3 + 6x^2 - 4x + 1\|_1 = \| - 4x^3 + 6x^2 - 4x\|_1 = 14.$$

For any real number c and for any $u(x), v(x) \in \mathbb{R}_n$, we have $\|cu(x)\| = |c|\|u(x)\|$, and if $u(x) \equiv v(x) \bmod (x^n+1)$ then $\|u(x)\| = \|v(x)\|$. Also, the usual triangle inequality still holds, that is,

$$\|u(x) + v(x)\| \leq \|u(x)\| + \|v(x)\|.$$

For $m \geq 1$, elements in R_n^m are viewed as row vectors of length m, similarly for $\mathbb{R}_n^m$ and $R_{n,q}^m$. For $\mathbf{u} = (u_1(x), \ldots, u_m(x)) \in R_n^m$, define

$$\|\mathbf{u}\|_p = \left(\sum_{i=1}^{m} \|u_i(x)\|_p^p\right)^{1/p}, \quad \|\mathbf{u}\|_\infty = \max_{1\leq i\leq m} \|u_i(x)\|_\infty.$$

Also, for any real number z, the function $\lfloor z \rceil$ denotes the integer closest to z. For example, $\lfloor 1.6 \rceil = 2$, $\lfloor -0.4 \rceil = 0$, however, $\lfloor -1.5 \rceil = -2$ or -1, either is fine. For any vector $\mathbf{v} \in \mathbb{R}^n$ (or any polynomial $\mathbf{v} \in \mathbb{R}_n$ with degree $< n$), $\lfloor \mathbf{v} \rceil$ is the vector (or the polynomial) when $\lfloor \cdot \rceil$ is applied to each entry (or each coefficient) of $\mathbf{v}$.

2.2. **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 $b > 0$, by b-bounded uniform random variable X on $\mathbb{Z}$, we mean X is uniform random on the integers i with $|i| \le b$, 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 $\alpha/\sqrt{2\pi}$. A random variable X over $\mathbb{R}$ is called sub-Gaussian with parameter α if $E(X) = 0$ and its moment generating function satisfies

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

If X is sub-Gaussian with parameter α, then its tails are dominated by a Gaussian of parameter α, i.e.,

$$\text{Prob}(|X| \ge t) \le 2\exp(-\pi(t/\alpha)^2), \quad \text{for all } t \ge 0.$$

A b-bounded random variable is always sub-Gaussian with parameter $b\sqrt{2\pi}$, a sum of independent sub-Gaussian random variables on $\mathbb{R}$ is still sub-Gaussian. 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; see [36].

Let $q > 1$ be any integer. Each probability distribution on $\mathbb{R}$ induces a discretized distribution on $\mathbb{Z}_q$ as follows. Pick a random number $z \in \mathbb{R}$ according to the given distribution, compute $y := \lfloor qz \rceil \bmod q$ so that y is between $-q/2$ and $q/2$, and return y. For example, when $q = 11$, $\lfloor q \cdot 1.6 \rceil = \lfloor 17.6 \rceil = 18 \equiv -4 \bmod q$, however $\lfloor q \cdot (-1.5) \rceil = \lfloor -16.5 \rceil = -17$ or -16 (then reduce by q), either is fine. We shall use discretized Gaussian distributions on $\mathbb{Z}_q$. For a Gaussian random variable X with parameter α, its discretized Gaussian over $\mathbb{Z}_q$ is sub-Gaussian with parameter $c\alpha q$ for some small constant $c \ge 1$.

2.3. **LWE problem, LWE ciphers, and modulus reduction.** Regev (2005 [97, 98]) introduced the learning with error (LWE) problem over $\mathbb{Z}_q$. Let χ be a probabilistic distribution on $\mathbb{Z}$, 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 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 $\mathbb{Z}_q^n \times \mathbb{Z}_q$. We call χ the error distribution of the LWE problem.

Theorem 2.1 (Regev [97, 98])**.** *Suppose the error distribution χ is a discretized Gaussian distribution on $\mathbb{Z}_q$ with parameter $\alpha > 0$ and $\alpha q \ge 2\sqrt{n}$. Then solving the LWE problem over $\mathbb{Z}_q$ is at least as hard as solving some approximate shortest vector problem for lattices of dimension n (under quantum reduction).*

Note that Brakerski et al [25] removed the quantum reduction requirement and proved that solving n-dimensional LWE problem over $\mathbb{Z}_q$ is at least as hard as solving a worst-case lattice problem in dimension $\sqrt{n}$. It is still an open question whether $\sqrt{n}$ in the lattice problem can be replaced by $O(n)$, which is desirable in applications. Also, a binary-LWE problem over $\mathbb{F}_q$ is the LWE-problem over $\mathbb{F}_q$ where the secret vector $\mathbf{s} \in \mathbb{F}_q^n$ is a vector in $\{0,1\}^n$. Brakerski et al [25] and Micciancio and Peikert [88] proved the binary-LWE problem

is also hard. For this reason, many cryptosystems in the literature use binary secret vectors $\mathbf{s}$, and we shall follow this tradition as well.

This breakthrough result shows that the average complexity of solving LWE problem is bounded below by the worst-case complexity of lattice problems. Hence LWE problem is believed to be hard on average, thus forms the security foundation of many post-quantum cryptosystems (that are secure even if quantum computers can be built). There are two minor problems in using Gaussian distributions in practice: one is that Gaussian distribution is relatively expensive to generate (compared to bounded uniform distributions as we use below); another is that there is a small probability of decoding failure for new ciphers computed from bootstrapping. In this paper, we present a compact FHE that never have decoding failure for both fresh and computed ciphertexts.

Error model for the current paper. We shall use bounded uniform distributions for errors, that is, we fixed some bound $\tau > 0$ and choose e uniform randomly in $[-\tau, \tau]$. In our choice of public keys and bootstrapping key (described later), the τ will be about n, and in the fresh ciphertexts of the original data, we use $\tau = n-1$. Also, in all the ciphertexts from our bootstrapping operations or from homomorphic computing of an arbitrary function, the error size is always bounded by $\tau = n-1$. Our error width $2(n-1)$ is much bigger than those used in all the proposed homomorphic schemes in the current literature, and is also much bigger than $2\sqrt{n}$ as required by Regev's Theorem above.

Our error distribution is not discretized Gaussian, hence we can not apply Theorem 2.1 directly. However, we argue that our error distribution could be better in the sense that the LWR problem may be harder. We present two heuristic reasons. First, suppose one uses a discretize Gaussian $D_{\alpha,q}$ on $\mathbb{Z}_q$ with parameter $\alpha > 0$ so that $\alpha q \approx 2\sqrt{n}$. Then, for a random e between $-q/2$ and $q/2$ chosen according to $D_{\alpha,q}$, the inequality $|e| \le c\sqrt{n}$ holds with high probability (for any constant $c \ge 3$). Note that all the known attacks on the LWE problem based on lattice basis reduction become less effective when the error width increases. Since our schemes use a much bigger error width than $2\sqrt{n}$, our LWE problem should not be easier. Secondly, we argue from information theory point of view. Note that a uniform distribution on $[-n, n]$ has a much higher entropy than that of a discretized Gaussian on $\mathbb{Z}_q$ with standard deviation $c\sqrt{n}$ where $c > 0$ is a constant [44]. In computing $b := \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q$, more information (bits) of $\langle \mathbf{s}, \mathbf{a} \rangle$ is destroyed by e in our case. Hence it is harder to reconstruct $\mathbf{s}$ from a list of samples $(\mathbf{a}, b)$ from our error distribution.

Conjecture 2.2. *Suppose the error distribution χ in the LWE problem over $\mathbb{Z}_q$ is τ-bounded uniform distribution where $\tau = cn$ for some constant $c > 0$. Then solving the LWE problem over $\mathbb{Z}_q$ is at least as hard as solving some approximate shortest vector problem for lattices of dimension n.*

We should note that it is shown in [88, 30] that the LWE problem with bounded uniform distributions for errors is hard when the number of samples is limited (linear in n). However, the above conjecture is still an open when the number of samples are allowed to be any polynomial in n.

LWE ciphers. Regev [97, 98] also introduced a cryptosystem based on the LWE problem. Let $\mathbf{s} \in \mathbb{Z}_q^n$ be a secret key, $D_q = \lfloor q/4 \rfloor$ and $1 \le \tau < D_q/2$. To encrypt a message bit $x \in \{0, 1\}$, pick $\mathbf{a} \in Z_q^n$ 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. Regev originally used $D = \lfloor q/2 \rfloor$, we use $q/4$ so that we can perform homomorphic bit operations on ciphertexts via bootstrapping (see below).

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 $-q/2 < b_1 \le q/2$, and $x_1 := \lfloor b_1/D_q \rceil$. Then $x = x_1$. The reason is that

$$b_1 \equiv b - \langle \mathbf{s}, \mathbf{a} \rangle \equiv e + xD_q \pmod q,$$

and $|e| \le \tau < D_q/2$ implies that $-q/2 < e + xD_q \le q/2$, hence $b_1 = e + xD_q$ (as real numbers) and $b_1/D_q = e/D_q + x$ with $|e/D_q| < 1/2$.

Modulus reduction. Next 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. This technique of modulus reduction is used in [23, 25, 27]. Here we show how precisely the error size changes, depending only on the entropy (i.e. the bit size) of $\mathbf{s}$.

Lemma 2.3. *Let $\mathbf{s}, \mathbf{a} \in \mathbb{Z}_q^n$, $e \in \mathbb{Z}$ with $|e| \le \tau$, $x \in \{0,1\}$, $D_r = \lfloor r/4 \rfloor$, and*

$$b \equiv \langle \mathbf{s}, \mathbf{a} \rangle + e + xD_q \pmod q.$$

(a) *Suppose $\tau \le q(n-3)/(2r)$, $q \ge 4r$, and each entry of $\mathbf{s}$ is in $\{0,1\}$. 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 $e' \in \mathbb{Z}$ with $|e'| < n$.

(b) *Let $\ell = \lceil \log_2 q \rceil$ and $q \ge 16$. Suppose $\tau \le q(n\ell - 5)/(2r)$ and $\mathbf{s} \in \mathbb{Z}_q^n$ is arbitrary. Then there exist $\mathbf{s}' \in \{0,1\}^{n\ell}$, $\mathbf{a}' \in \mathbb{Z}_r^{n\ell}$ and $b' \in \mathbb{Z}_r$ so that*

$$b' \equiv \mathbf{s}'(\mathbf{a}')^t + e' + xD_r \pmod r,$$

for some $e' \in \mathbb{Z}$ with $|e'| < n\ell$.

Proof. For part (a), 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 $|\epsilon_i| \le 1/2$ for $0 \le i \le n$. 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

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

We claim that $|\frac{rD_q}{q} - D_r| < 1$. To see this, let $q = 4q_1 + q_0$ and $r = 4r_1 + r_0$ where $0 \le r_0 < 4$ and $0 \le q_0 < 4$. Then

$$\frac{rD_q}{q} - D_r = \frac{(4r_1 + r_0)q_1}{4q_1 + q_0} - r_1 = \frac{r_1 + \frac{r_0}{4}}{1 + \frac{q_0}{q_1}} - r_1 = \frac{\frac{r_0}{4} - q_0\frac{r_1}{q_1}}{1 + \frac{q_0}{q_1}},$$

which has absolute value < 1 whenever $q_1 \ge 4r_1$, hence true when $q \ge 4r$. As $s_i \in \{0,1\}$, $x \in \{0,1\}$, $|\epsilon_i| \le 1/2$ and $|e| \le \tau \le q(n-5)/(2r)$, it follows that

$$|e'| < (n+1)\frac{1}{2} + 1 + \frac{r\tau}{q} \le \frac{n+3}{2} + \frac{n-3}{2} = n.$$

For part (b), we write each entry of $\mathbf{s}$ in binary representation:

$$s_i = \sum_{j=0}^{\ell-1} s_{ij} 2^j,$$

where $s_{ij} \in \{0,1\}$ for $1 \le i \le n$ and $0 \le j \le \ell - 1$. Then

$$\langle \mathbf{s}, \mathbf{a} \rangle = \sum_{i=1}^{n} s_i a_i = \sum_{i=1}^{n} \sum_{j=0}^{\ell-1} s_{ij}(a_i 2^j).$$

Apply part (a) to the case with $\mathbf{s}$ replaced by $\mathbf{s}' = (s_{ij}) \in \{0,1\}^{n\ell}$. This completes the proof. □

Remarks. (i) In the proof of part (a), one can randomize the rounding so that ϵ_i is random subGaussian; see Lemma 5 in Ducas and Micciancio (2015 [51]). Then the probability that $|\epsilon_0 - \sum_{i=1}^{n} s_i \epsilon_i| \ge t\sqrt{w+1}$ is at most $2\exp(-2t^2)$ where w is the number of nonzero entries in $\mathbf{s}$. Hence, if $\tau \le 10q\sqrt{n}/r$, then the probability that $|e'| \ge 20\sqrt{n}$ is at most $2\exp(-200)$, which is extremely small. This would allow one to use a small $q = O(\sqrt{n})$ for encryption, but there would be a very small probability of decoding failure. We shall use a bigger $q = O(n)$ and make sure that there is no decoding failure at all in our encryption and bootstrapping schemes, hence allows us to perform any number of homomorphic bit operations without decoding failure.

(ii) Part (b) and its proof shows that an LWE problem with arbitrary $\mathbf{s} \in \mathbb{Z}_q^n$ can be reduced to an LWE problem with binary $\mathbf{s}' \in \{0,1\}^{n\ell}$ over $\mathbb{Z}_q$ as well as over $\mathbb{Z}_r$ for any $r < q$. Hence the LWE problem with binary secret is as hard as an arbitrary secret with dimension reduced by a factor of $\log_2(q)$. In the following, we shall only use binary $\mathbf{s}$.

2.4. **RLWE ciphers.** Lyubashevsky, Peikert and Regev (2010 [87]) introduced the ring learning with error (RLWE) problem in order to get more efficient encryption schemes. Let

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

that is, polynomials are computed modulo $x^n + 1$ and all the coefficients modulo q. Each element of $R_{n,q}$ (or R_n) is of the form $a(x) = \sum_{i=0}^{n-1} a_i x^i$, where $a_i \in \mathbb{Z}$, representing an n-tuple $(a_0, a_1, \ldots, a_{n-1}) \in \mathbb{Z}_q^n$ (or in $\mathbb{Z}^n$). Let $s(x) \in R_{n,q}$ be any secret key. An RLWE sample is of the form $(a(x), b(x)) \in \mathbb{R}_{n,q}^2$ 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 $\mathbf{v} \in \mathbb{R}_{n,q}^2$ 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.

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 $(x^n + 1, q)$.

2.5. **Gadget matrix, external product and GSW ciphers.** In order to perform homomorphic multiplication, Gentry, Sahai, and Waters (2013 [65]) introduced the idea of 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. Let B and ℓ be positive integers so that $B^\ell \geq q$. Let

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

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

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

where $a_i \in \mathbb{Z}$ has small size. For example, we can let $-B/2 < a_i \leq B/2$, then $(a_0, a_1, \ldots, a_{\ell-1})$ is unique. This is good for applications where new ciphertexts from homomorphic computing do not need to be random. However there are applications (say zero knowledge proof, see below) where it is important that new ciphertexts are uniformly random (in some sense). Hence we shall allow $|a_i|$ to be as big as $2B$. The following lemma is straightforward to prove.

Lemma 2.4 (Random Flattening)**.** *Suppose $B^\ell \geq q$ and $a \in \mathbb{Z}$. For $0 \leq i \leq \ell - 1$, pick $x_i \in \mathbb{Z}$ with $|x_i| \leq 3B/2$ uniform randomly and independently, and let*

$$a - (x_0 + x_1 B + \cdots + x_{\ell-1} B^{\ell-1}) \equiv y_0 + y_1 B + \cdots + y_{\ell-1} B^{\ell-1} \pmod{q},$$

where $|y_i| \leq B/2$ for $0 \leq i \leq \ell - 1$. Set $a_i = x_i + y_i$ for $0 \leq i \leq \ell - 1$. Then $(a_0, a_1, \ldots, a_{\ell-1})$ is uniform random solution to

$$a \equiv a_0 + a_1 B + \cdots + a_{\ell-1} B^{\ell-1} \pmod{q},$$

with $|a_i| \leq 2B$ for $0 \leq i \leq \ell - 1$.

We can extend this to any list of elements in $\mathbb{Z}_q$. Thus every polynomial $a(x) \in R_{n,q}$ can be written as

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

where $a_i(x) \in R$ is independently uniform random with $||a_i(x)||_\infty \leq 2B$ for $0 \leq i \leq \ell - 1$. Define

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

an $(2\ell) \times 2$ matrix, called a gadget matrix. 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 uniform random with $||\mathbf{u}(x)||_\infty \leq 2B$. We define

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

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 right on $(a(x), b(x))$, a row vector of two polynomials with coefficients in $\mathbb{Z}_q$ (of large size), to get $\mathbf{u}(x) = (a(x), b(x)) \triangleleft G^{-1}$, a random row verctor of 2ℓ polynomials each with coefficients at most $2B$ (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

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

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

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

which will be crucial in bootstrapping below.

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

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

which is a random vector in $R_{n,q}^2$, since $\mathbf{v} \triangleleft G^{-1}$ is a random row vector of length 2ℓ and A is an $(2\ell \times 2)$ matrix. This definition can be extended to define 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 [65]). Recently, Chillotti et al (2016 [39, 40]) 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 $s_i \in \{0, 1\}$, representing 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 $||\mathbf{w}(x)||_\infty \le \tau$. An RLWE cipher for $m(x)$ with error size τ is of the form

$$\mathrm{RE}_{\mathbf{s}}(z) = \mathbf{v} + m(x)D_q(0, 1) \in R_{n,q}^2 \tag{6}$$

where $\mathbf{v} \in R_{n,q}^2$ is an RLWE sample so that

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

where $e(x) \in R_n$ with $||e(x)||_\infty \le \tau$. The next lemma is observed by Chillotti et al (2016 [39]), here we make the error bound explicit in our error model.

Lemma 2.5. *Let $m_0, m_1 \in R_n$ be any two polynomials. For any $RE_{\mathbf{s}}(m_0)$ with error size τ_0 and any $GSW_{\mathbf{s}}(m_1)$ with error size τ_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 size at most $||m_1||_1\tau_0 + 4Bn\ell\tau_1$.

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 $(x^n + 1, q)$,

$$\mathbf{v}(-s(x), 1)^t \equiv w_0(x), \qquad A(-s(x), 1)^t \equiv \mathbf{w}(x)^t, \tag{7}$$

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

$$h = \mathrm{RE}_{\mathbf{s}}(m_0(x)) \triangleleft G^{-1} = (h_1, \ldots, h_{2\ell}) \in R_n^{2\ell},$$

with $||h||_\infty \le 2B$. Computing modulo (x^n+1, q), we have

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

where, in the second last equation, we used the property of G^{-1} from (4). This proves the first part of the lemma.

It remains to estimate the error size. The error polynomial is, using (7),

$$(hA + m_1\mathbf{v})(-s(x), 1)^t \equiv h\,\mathbf{w}(x)^t + m_1 w_0(x) \pmod{(x^n+1, q)}.$$

Let $m_1 = \sum_{i=0}^{n-1} m_{1i} x^i \in R_n$ and $h_j = \sum_{i=0}^{n-1} h_{ji} x^i$ where $h_{ji} \in \mathbb{Z}$ with $|h_{ji}| \le 2B$ for $0 \le i \le n-1$ and $1 \le j \le 2\ell$. Note that

$$h\,\mathbf{w}(x)^t + m_1 w_0(x) = \sum_{j=1}^{2\ell}\sum_{i=0}^{n-1} h_{ji} x^i w_j(x) + \sum_{i=0}^{n-1} m_{1i} \cdot x^i w_0(x).$$

Hence

$$\begin{aligned}||h\,\mathbf{w}(x)^t + m_1 w_0(x)||_\infty &\le \sum_{j=1}^{2\ell}\sum_{i=0}^{n-1} |h_{ji}| \cdot ||x^i w_j(x)||_\infty + \sum_{i=0}^{n-1} |m_{1i}| \cdot ||x^i w_0(x)||_\infty \\ &\le \sum_{j=1}^{2\ell}\sum_{i=0}^{n-1} 2B\tau_1 + \sum_{i=0}^{n-1} |m_{1i}|\,\tau \le (2\ell)n2B\tau_1 + ||m_1||_1\,\tau_0.\end{aligned}$$

This completes the proof. □

3. Homomorphic bit operations

Suppose we are given any two LWE ciphers in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ for $x_1, x_2 \in \{0,1\}$ with error size $< D_r/4$. We want to compute LWE ciphers for

$$x_1 \wedge x_2, \quad x_1 \vee x_2, \quad x_1 \oplus x_2,$$

still with error size $< D_r/4$, where $x_1 \wedge x_2 = x_1 \cdot x_2$ (AND gate), $x_1 \vee x_2 = x_1$ OR x_2 (OR gate) and $x_1 \oplus x_2 = x_1 + x_2 \pmod 2$ (XOR gate, exclusive OR). In this section, we show how to compute all these new ciphers simultaneously by one bootstrapping operation. Note that a cipher for $(x_1$ NAND $x_2) = 1 - x_1 \wedge x_2$ can be obtained from that of $x_1 \wedge x_2$ trivially, so we just need to focus on the above three bit operations. We follow the approach in Ducas and Micciancio (2015 [51]) and Chillotti et al. (2016 [39]), however, we do not need to perform key switch as they do.

Let $\mathbf{s} = (s_0, \ldots, s_{n-1}) \in \{0,1\}^n$, representing an n-bit secret key of a user. Suppose $r \ge 2n$ and is divisible by 8, Q is much bigger than r (to be determined later), $B^\ell \ge Q$ and

$$m = r/2, \quad D_r = \lfloor r/4 \rfloor, \quad \tilde{D}_Q = \lfloor Q/8 \rfloor.$$

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

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

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 (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

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

and (in $\mathbb{Z}$),

$$x_1 + x_2 = 2(x_1 \wedge x_2) + (x_1 \oplus x_2).$$

Hence

$$b_1 + b_2 \equiv \langle \mathbf{s}, \mathbf{a}_1 + \mathbf{a}_2 \rangle + (x_1 \wedge x_2) \cdot 2D_r + (x_1 \oplus x_2) D_r + e_1 + e_2 \pmod{r}. \tag{8}$$

Let $\mathbf{u} = (u_0, \ldots, u_{n-1}) = \mathbf{a}_1 + \mathbf{a}_2 \in \mathbb{Z}_r^n$, $u_n = b_1 + b_2 \in \mathbb{Z}_r$, and $e = e_1 + e_2 \in \mathbb{Z}$. Then $|e| < D_r/2$ and the equation (8) becomes

$$u_n \equiv \sum_{i=0}^{n-1} s_i u_i + (x_1 \wedge x_2) \cdot 2D_r + (x_1 \oplus x_2) D_r + e \pmod{r}. \tag{9}$$

Let $w = u_n - \sum_{i=0}^{n-1} s_i u_i$. Since $D_r = m/2 = r/4$, the equation (9) implies the following:

$$\begin{aligned} w - \frac{r}{8} &\equiv (x_1 \wedge x_2) m + \left((x_1 \oplus x_2) - \frac{1}{2} \right) D_r + e \pmod{r}, \\ w + \frac{r}{8} &\equiv (x_1 \vee x_2) m - \left((x_1 \oplus x_2) - \frac{1}{2} \right) D_r + e \pmod{r}. \end{aligned}$$

In the second equation, we used the fact that

$$(x_1 \oplus x_2) = (x_1 \vee x_2) - (x_1 \wedge x_2). \tag{10}$$

Since $|e| < D_r/2$ and $(x_1 \oplus x_2) - 1/2 = \pm 1/2$, we have

$$\left| \pm \left((x_1 \oplus x_2) - \frac{1}{2} \right) D_r + e \right| \le \frac{1}{2} D_r + |e| < \frac{1}{2} D_r + \frac{1}{2} D_r = D_r.$$

Therefore,

$$w - \frac{r}{8} \equiv (x_1 \wedge x_2) m + e_1 \pmod{r}, \tag{11}$$

$$w + \frac{r}{8} \equiv (x_1 \vee x_2) m + e_2 \pmod{r}, \tag{12}$$

for some $e_1, e_2 \in \mathbb{Z}$ with $|e_1| < D_r$ and $|e_2| < D_r$. The two equations (11) and (12) were first observed by Chillotti et al. [39].

Following Ducas and Micciancio (2015 [51]), we use the group homomorphism from the additive subgroup $(\mathbb{Z}_r, +)$ to the following multiplicative group of $R_{m,q} = \mathbb{Z}[x]/(x^m + 1, q)$:

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

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

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

For this $t(x)$, its the 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.

Lemma 3.1. *Suppose $r = 2m$ and m is divisible by 4. Let $T = \{j \in \mathbb{Z} : |j| < D_r\}$ and $t(x) = \sum_{j\in T} x^j \pmod{x^m+1}$, and assume $w \in \mathbb{Z}_r$ satisfies (11) and (12). Let the coefficients of $t(x)x^{-w} \pmod{x^m+1}$ at $x^{3m/4}$ and $x^{m/4}$ be c_1 and c_2, respectively. Then*

$$2(x_1 \wedge x_2) = 1 + c_1, \qquad 2(x_1 \vee x_2) = 1 - c_2.$$

Hence, by (10), we have $2(x_1 \oplus x_2) = -(c_1 + c_2)$.

Proof. Let $z_1 = x_1 \wedge x_2$. By (11), $w - r/8 \equiv z_1 m + e_1 \pmod{r}$ for some $e_1 \in T$, hence

$$x^{w-r/8} \equiv x^{z_1 m + e_1} \equiv (x^m)^{z_1} x^{e_1} \equiv (-1)^{z_1} x^{e_1} \pmod{x^m+1}.$$

Since $x^{-r/8} \equiv -x^{3m/4}$, we have

$$t(x)x^{-w} \equiv -(-1)^{z_1} t(x) x^{-e_1} x^{3m/4}.$$

Since $e_1 \in T$ and $e_1 \pm m \notin T$, $t(x)$ contains the term x^{e_1} with coefficient 1. Hence the coefficient of $x^{3m/4}$ in $t(x)x^{-w}$ is equal to $-(-1)^{z_1}$, which is assumed to be c_1. Therefore, $1 + c_1 = 1 - (-1)^{z_1} = 2z_1$ as claimed. The proof for c_2 is similar. □

For any subset T of $\mathbb{Z}_r$, we define an RLWE cipher for $t(x) = \sum_{j\in T} x^j$ to be of the form

$$\mathrm{RE}_{\mathbf{s}}(t(x)) = \mathbf{v} + t(x)\tilde{D}_q(0,1) \in R^2_{m,Q},$$

where $\mathbf{v} \in R^2_{m,Q}$ is an RLWE sample. Note that we use $\tilde{D}_Q = \lfloor Q/8 \rfloor$ instead of $D_Q = \lfloor Q/4 \rfloor$ due to the factor 2 in Lemma 3.1. Note that, for any $z \in \{0,1\}$ and $u \in \mathbb{Z}$, we have

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

Let $C = \mathrm{GSW}_{\mathbf{s}}(z) \in R^{(2\ell)\times 2}_{m,Q}$ be any GSW cipher. One can map $zu \in \mathbb{Z}_r$ to x^{zu}, then to a GSW cipher: $G + (x^u - 1)C \in R^{(2\ell)\times 2}_{m,Q}$. The following lemma is due to Chillotti et al. [39], but our error bound is simpler.

Lemma 3.2. *Let $z \in \{0,1\}$, $u \in \mathbb{Z}_r$ and $T \subseteq \mathbb{Z}_r$. Suppose $RE_{\mathbf{s}}((t(x)) \in R^2_{m,Q}$ has error size τ and $C = GSW_{\mathbf{s}}(z) \in R^{(2\ell)\times 2}_{m,Q}$ has error size τ_1. Then*

$$RE_{\mathbf{s}}(t(x)) \odot (G + (x^u - 1)C) = RE_{\mathbf{s}}(t(x)x^{zu}) \in R^2_{m,Q},$$

and furthermore, $RE_{\mathbf{s}}(t(x)x^{zu})$ has error size at most $\tau + 4Br\ell\tau_1$.

Proof. Let $C = \mathrm{GSW}_{\mathbf{s}}(z) = A + zG$ where $A \in \mathbb{R}^{(2\ell)\times 2}_{m,Q}$ and

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

with $\mathbf{w}(x) \in R_m$ and $||\mathbf{w}(x)||_\infty \le \tau_1$. Then

$$G + (x^u - 1)C = (x^u - 1)A + (1 + (x^u - 1)z)G = (x^u - 1)A + x^{zu}G = \mathrm{GSW}_{\mathbf{s}}(x^{zu}),$$

and the error polynomial is $(x^u - 1)A(-s(x), 1)^t \equiv (x^u - 1)\mathbf{w}(x)$, with error size

$$||(x^u - 1)\mathbf{w}(x)||_\infty \le 2||\mathbf{w}(x)||_\infty \le 2\tau_1.$$

By Lemma 2.5, we have

$$\mathrm{RE}_{\mathbf{s}}(t(x)) \odot (G + (x^u - 1)C) = \mathrm{RE}_{\mathbf{s}}(t(x)) \odot \mathrm{GSW}_{\mathbf{s}}(x^{zu}) = \mathrm{RE}_{\mathbf{s}}(t(x)x^{zu}),$$

with error at most $\tau||x^{zu}||_\infty + 4Bm\ell(2\tau_1) = \tau + 4Br\ell\tau_1$. □

Our bootstrapping algorithm is described in Figure 1. In Step 4 of the algorithm, we need to extract some coefficients. Let $a(x) = a_0 + a_1 x + \cdots + a_{m-1}x^{m-1} \in \mathbb{R}_m$. We define

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

	Bootstrapping Algorithm : $\mathrm{BT}_{bk}(\mathbf{v}_1, \mathbf{v}_2)$
Input:	$bk = (C_0, \ldots, C_{n-1}) \in R_{m,Q}^{(2\ell)\times 2}$: a bootstrapping key for a user, $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ where $\mathbf{v}_i = E_{\mathbf{s}}(x_i)$ for $x_1, x_2 \in \{0,1\}$.
Output:	$E_{\mathbf{s}}(x_1 \wedge x_2)$, $E_{\mathbf{s}}(x_1 \vee x_2)$, $E_{\mathbf{s}}(x_1 \oplus x_2) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 1.	Compute $\mathbf{u} := \mathbf{v}_1 + \mathbf{v}_2 = (u_0, \ldots, u_{n-1}, u_n) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 2.	Initialization: $t(x) := \sum_{j\in T} x^j$ where $T := \{j \in \mathbb{Z} : -D_r < j < D_r\}$, $A := (0, t(x)x^{-u_n}\tilde{D}_Q) \in R_{m,Q}^2$.
Step 3.	For k from 0 to $n-1$ do (Randomness is involved here) $A := A \odot (G + (x^{u_k} - 1)C_k)$.
Step 4.	Suppose $A = (\sum_{i=0}^{m-1} a_i x^i, \sum_{i=0}^{m-1} b_i x^i) \in R_{m,Q}^2$. Set $\mathbf{a}_1 := (\mathrm{Extract}(a(x), 3m/4), \tilde{D}_Q + b_{3m/4}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$, $\mathbf{a}_2 := (-\mathrm{Extract}(a(x), m/4), \tilde{D}_Q - b_{m/4}) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$, $\mathbf{a}_3 := \mathbf{a}_2 - \mathbf{a}_1 \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$.
Step 5.	Modulus reduction: For i from 1 to 3 do $\mathbf{c}_i := \lfloor r\mathbf{a}_i/Q \rceil \in \mathbb{Z}_r^n \times \mathbb{Z}_r$.
Step 6.	Return $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3$.

FIGURE 1. Homomorphic bit Operations

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

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

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

Theorem 3.3. *Suppose a bootstrapping key bk has error size at most τ_1, r is divisible by 8 and*

$$r \ge 16n, \quad Q \ge \frac{n}{n-3} 16Br^2\ell\tau_1.$$

Then, for any two LWE ciphers $E_{\mathbf{s}}(x_i) = \mathbf{v}_i \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size $< D_r/4$ where $x_i \in \{0,1\}$ for $i = 1, 2$, the bootstrapping algorithm in Figure 1 outputs random LWE ciphers $E_{\mathbf{s}}(x_1 \wedge x_2)$, $E_{\mathbf{s}}(x_1 \vee x_2)$, $E_{\mathbf{s}}(x_1 \oplus x_2) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$, all with error size $< n \le D_r/4$.

Proof. Let $w = u_n - \sum_{i=0}^{n-1} s_i u_i$ where $u = (u_0, \ldots, u_{n-1}, u_n) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ is computed in Step 1. Since $\mathbf{v}_1$ and $\mathbf{v}_2$ have error size $< D_r/4$, the equations (11) and (12) hold. At Step 2, $A = (0, t(x)x^{-u_n}\tilde{D}_Q) = \mathrm{RE}_{\mathbf{s}}(t(x)x^{-u_n}) \in R_{m,Q}^2$, a trivial RLWE cipher for $t(x)x^{-u_n}$ with no error. At Step 3, we apply Lemma 3.2 repeatedly n times. After Step 3, we have $A = \mathrm{RE}_{\mathbf{s}}(t(x)x^{-w})$ with error size at most $n(4Br\ell\tau_1) = 4Bnr\ell\tau_1$.

At Step 4, letting $A = (a(x), b(x)) \in \mathbb{R}_{m,Q}$, we have

$$b(x) \equiv a(x)s(x) + t_1(x)\tilde{D}_Q + v(x) \mod (x^m+1, Q), \tag{13}$$

where $t_1(x) = t(x)x^{-w}$ and and $v(x) \in R_m$ with $||v(x)||_\infty \le 4Bnr\ell\tau_1$. Now consider the coefficients at $x^{3m/4}$ and $x^{m/4}$. Let

$$\mathbf{u}_1 := \mathrm{Extract}(a(x), 3m/4), \qquad \mathbf{u}_2 := \mathrm{Extract}(a(x), m/4).$$

We have

$$\begin{aligned} b_{3m/4} &\equiv \langle \mathbf{s}, \mathbf{u}_1 \rangle + c_1 \tilde{D}_Q + v_{3m/4} \pmod{Q}, \\ b_{m/4} &\equiv \langle \mathbf{s}, \mathbf{u}_2 \rangle + c_2 \tilde{D}_Q + v_{m/4} \pmod{Q}, \end{aligned}$$

where c_1 and c_2 are the coefficients of $t(x)x^{-w}$ at $x^{3m/4}$ and $x^{m/4}$, respectively, and $|v_i| \le 4Bnr\ell\tau_1$ for $i \in \{3m/4, m/4\}$. By Lemma 3.1,

$$\begin{aligned} \tilde{D}_Q + b_{3m/4} &\equiv \langle \mathbf{s}, \mathbf{u}_1 \rangle + (x_1 \wedge x_2) \cdot 2\tilde{D}_Q + v_{3m/4} \pmod{Q}, \\ \tilde{D}_Q - b_{m/4} &\equiv \langle \mathbf{s}, -\mathbf{u}_2 \rangle + (x_1 \vee x_2) \cdot 2\tilde{D}_Q - v_{m/4} \pmod{Q}, \\ -(b_{3m/4} + b_{m/4}) &\equiv \langle \mathbf{s}, -(\mathbf{u}_1 + \mathbf{u}_2) \rangle + (x_1 \oplus x_2) \cdot 2\tilde{D}_Q - (v_{3m/4} + v_{m/4}) \pmod{Q}, \end{aligned}$$

with $|v_{3m/4} + v_{m/4}| \le 8Bnr\ell\tau_1$. Hence $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$ from Step 4 are LWE ciphers for $E_{\mathbf{s}}(x_1 \wedge x_2)$, $E_{\mathbf{s}}(x_1 \vee x_2)$, $E_{\mathbf{s}}(x_1 \oplus x_2)$ with error at most $8Bnr\ell\tau_1$.

Finally, at Step 5, we apply Lemma 2.3 (a) to $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ to get LWE ciphers in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size $< n$. For example, let $\mathbf{c}_1 = (\mathbf{v}, v_n) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$, the lemma tells us that

$$v_n \equiv \langle \mathbf{s}, \mathbf{v} \rangle + (x_1 \wedge x_2) \cdot 2\tilde{D}_r + e \pmod{r}$$

for some $e \in Z$ with $|e| < n$ where $\tilde{D}_r = r/8$, hence $2\tilde{D}_r = r/4 = D_r$. □

4. Fully homomorphic encryption scheme

4.1. **Key generations.** We shall assume that $n \ge 64$ is a power of 2, r is divisible by 8, $m = r/2$, $B = 35r^2n$, and

$$r \ge 16n, \quad q \ge nr, \quad 1220r^4n^2 \le Q < 1225r^4n^2 = B^2. \tag{14}$$

Let $R_n = \mathbb{Z}[x]/(x^n + 1)$, $R_m = \mathbb{Z}[x]/(x^m + 1)$, and

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

$$D_r = \lfloor r/4 \rfloor, \quad D_q = \lfloor q/4 \rfloor, \quad \tilde{D}_Q = \lfloor Q/8 \rfloor, \quad G = \begin{pmatrix} 1 & 0 \\ B & 0 \\ 0 & 1 \\ 0 & B \end{pmatrix}.$$

Each user generates a secret key, a public key and a bootstrapping key as described below.
Secret key. Pick $\mathbf{s} = (s_0, s_1, \ldots, s_{n-1}) \in \{0,1\}^n$ uniform randomly, and let $s(x) = \sum_{i=0}^{n-1} s_i x^i$, representing the n-bit secret key of a user.
Public key. A corresponding public key in $R_{n,q}^2$ is generated as $pk = (k_0(x), k_1(x))$ where $k_0(x) \in R_{n,q}$ is chosen uniform randomly and

$$k_1(x) := k_0(x)s(x) + e(x) \bmod (x^n + 1, q) \tag{15}$$

with $e(x) \in R_n$ being chosen bounded uniform randomly so that

$$||e(x)||_\infty < D_q/(41n),$$

that is, each coefficient of $e(x)$ is chosen uniform randomly and independently between $-D_q/(41n)$ and $D_q/(41n)$.
Bootstrapping key. A corresponding bootstrapping key $bk = (C_0, C_1, \ldots, C_{n-1})$ is generated as follows. For each $0 \le i \le n-1$ do the following:

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

$$||e_{ji}(x)||_\infty \le n, \qquad 1 \le j \le 4,$$

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

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

Note that the error in each C_i is at most n. Let's check the condition in Theorem 3.3 where $\tau_1 = n$, $\ell = 2$, and $B = 35r^2n$:
$$\frac{n}{n-3} 16Br^2\ell\tau_1 = \frac{n}{n-3} 32Br^2n = \frac{n}{n-3} 1120r^4n^2 < 1220r^4n^2,$$
whenever $n \geq 34$. Hence, by the choice of Q in (14), we have
$$\frac{n}{n-3} 16Br^2\ell\tau_1 < Q < B^2 = 1225r^4n^2.$$
This means that the bootstrapping algorithm in Figure 1 will work when r and Q satisfy (14).

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 [94], 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 $P(u,x)$ being nearly uniform random in $R_{n,r}$.

4.2. **Encryption schemes.** Regev [97, 98] introduced an encryption scheme that is semantically secure due to his theorem on the hardness of the LWE problem. In order to improve the efficiency of Regev's scheme, Lyubashevsky, Peikert and Regev [87] introduced an encryption scheme based on the RLWE problem, which they prove is hard assuming the ideal lattice problem in cyclotomic rings is hard. We present two encryption schemes based on the RLWE problem: one using private keys and other using public keys. Our schemes are modified from the scheme in [87], however, we add a step for rounding and modulus reduction. We note that the technique of modulus reduction has been used in [23, 25, 27] (as mentioned earlier), and Brakerski [21] suggested in a comment on using rounding to reduce ciphertext size. We shall combine all these techniques in our schemes, and by carefully choosing error distributions, we significantly reduce the cipher expansion to 6 for encryption with private keys as in Lemma 4.1 and to $10 + \log_2(n)$ for encryption with public keys as in Lemma 4.2.

4.2.1. *Encryption with private keys.* The scheme is presented in Figure 2.

Lemma 4.1. *Let $r = 2^{t+1}$ and $(a(x), b(x)) \in R_{n,r}^2$ be as computed in Figure 2. Then there exists $w_3(x) \in R_n$ with $||w_3(x)||_\infty < D_r/4$ so that*
$$2^{t-4}b(x) - s(x)a(x) \equiv w_3(x) + m(x)D_r \bmod (x^n + 1, r).$$
In particular, when $r = 16n$, $(u, \mathbf{v})$ returned in Step 4 has $6n$ bits and represents an RLWE cipher $RE_{\mathbf{s}}(m(x))$ with error size $< n$.

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

	Encryption with private key : $\mathrm{RE}_{\mathbf{s}}(m(x))$
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 $m_i \in \{0,1\}$, an n-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:	$(u, \mathbf{v}) \in \{0,1\}^n \times \{0,1\}^{5n}$
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 5 bits for each coefficient of $b_1(x)$: $b(x) := \lfloor b_1(x)/2^{t-4} \rfloor$. Let $\mathbf{v} \in \left(\{0,1\}^5\right)^n$ denote the bit representation of $b(x)$.
Step 4.	Return $(u, \mathbf{v})$.

FIGURE 2

Note that $r = 2^{t+1}$ and $D_r = 2^{t-1}$, so

$$|| - b_0(x) + w(x)||_\infty \le ||b_0(x)||_\infty + ||w(x)||_\infty < 2^{t-4} + D_r/8 = 2^{t-3} = D_r/4.$$

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

Remarks for the case when r is not a power of 2. One can modify Step 3 of Figure 2 as

$$b(x) := \lfloor b_1(x)/2^{t-5} \rfloor,$$

that is, taking the highest 6 bits for each coefficient of $b_1(x)$. Then the statement in Lemma 4.1 becomes

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

for some $w_3(x) \in R_n$ with $||w_3(x)||_\infty < D_r/4$, and each ciphertext $(a(x), b(x)) = \mathrm{RE}_{\mathbf{s}}(m(x))$ has $n(10+\lceil \log_2(n) \rceil)$ bits. For the proof, one can similar show that $||w_3(x)||_\infty < 2^{t-5} + D_r/8$. Suppose $r = 2^t + 4r_1 + r_0 > 2^t$ where $r_1 \ge 0$ and $0 \le r_0 < 4$. Then

$$2^{t-5} + \frac{D_r}{8} = 2^{t-5} + 2^{t-5} + \frac{r_1}{8} \le \frac{D_r}{4} = 2^{t-4} + \frac{r_1}{4},$$

hence $||w_3(x)||_\infty < D_r/4$.

4.2.2. *Encryption with public keys.* The scheme is presented in Figure 3.

Lemma 4.2. *Suppose $r = 2^{t+1}$, $r \ge 16n$, $q \ge 4r$ and $n > 164$. Let $(a(x), b(x)) = RE_{pk}(m(x)) \in R_{n,r}^2$ be any ciphertext output by Step 4 of Figure 3. Then*

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

for some $w_3(x) \in R_n$ with $||w_3(x)||_\infty < D_r/4$. In particular, when $r = 16n$, each ciphertext $RE_{pk}(m(x))$ has $n(10 + \log_2(n))$ bits and the error, i.e. each coefficient of $w_3(x)$, is random in $(-n, n)$.

Proof. By Step 3, we have

$$a(x) = ra_1(x)/q + v_1(x), \quad \text{and} \quad 2^{t-5}b(x) = rb_1(x)/q + v_0(x)$$

	Encryption with public key : $\mathrm{RE}_{pk}(m(x))$
Input:	$pk = (k_0(x), k_1(x)) \in R_{n,q}^2$,
	$m(x) = \sum_{i=0}^{n-1} m_i x^i$: an n-bit message where each $m_i \in \{0,1\}$,
	$t := \lceil \log_2(r) \rceil - 1$, hence $2^t < r \le 2^{t+1}$.
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 $\|\|w_1(x)\|\|_\infty \le D_q/(41n)$,
	Pick $w_2(x) \in R_n$ randomly with $\|\|w_2(x)\|\|_\infty \le D_q/82$.
Step 2.	Compute:
	$a_1(x) := k_0(x)u(x) + w_1(x) \bmod (x^n+1, q)$,
	$b_1(x) := k_1(x)u(x) + w_2(x) + m(x)D_q \bmod (x^n+1, q)$.
	(Both $a_1(x)$ and $b_1(x)$ have coefficients in $[0, q-1]$.)
Step 3.	Modulus reduction and rounding:
	$a(x) := \left\lfloor \frac{r}{q} a_1(x) \right\rceil$, $\quad b(x) := \left\lfloor \frac{r}{2^{t-5}q} b_1(x) \right\rceil$.
	(Each coefficient of $b(x)$ is in $[0, 2^6 - 1]$, hence has 6 bits.)
Step 4.	Return $(a(x), b(x))$.

FIGURE 3

where $v_i(x) \in \mathbb{R}[x]$ with degree $< n$ for $i = 0$ and 1, $||v_1(x)||_\infty \le 1/2$ and $|v_0(x)|_\infty < 2^{t-5}$. By (15) and Step 2, both computed modulo (x^n+1, q), there exist polynomials $h_1(x), h_2(x) \in \mathbb{Z}[x]$ so that

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

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

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

where $w_3(x) = \frac{r}{q}w(x) + v_0(x) - s(x)v_1(x) + m(x)\left(\frac{r}{q}D_q - D_r\right)$. Since $q \ge 4r$, we have $|\frac{r}{q}D_q - D_r| < 1$. We need to estimate the coefficient size of other terms in $w_3(x)$ (when reduced modulo x^n+1). Note that, for $e(x) = \sum_{j=0}^{n-1} e_j x^j \in R_n$ and any $0 \le i \le n-1$, we have

$$x^i e(x) \equiv -(e_{n-i}x^0 + \cdots + e_{n-1}x^{i-1}) + \tau x^i + \cdots + e_{n-i-1}x^{n-1} \pmod{x^n+1}.$$

Hence $||x^i e(x)||_\infty = ||e(x)||_\infty$. By choice, $u(x) = \sum_{i=0}^{n-1} u_i x^i$ where $u_i \in \{-1, 0, 1\}$. Then

$$||u(x)e(x)||_\infty \le \sum_{i=0}^{n-1} ||u_i x^i e(x)||_\infty = \sum_{i=0}^{n-1} ||e(x)||_\infty = n||e(x)||_\infty \le n\frac{D_q}{41n} = \frac{D_q}{41}.$$

Similarly, since $s_i \in \{0, 1\}$, we have

$$||s(x)w_1(x)||_\infty \le n\,||w_1(x)||_\infty \le n\frac{D_q}{41n} = \frac{D_q}{41}.$$

Therefore,

$$||w(x)||_\infty \le ||u(x)e(x)||_\infty + ||w_2(x)||_\infty + ||s(x)w_1(x)||_\infty \le 2\,\frac{D_q}{41} + \frac{D_q}{82} = \frac{5}{82}D_q.$$

Also,

$$||v_0(x) - s(x)v_1(x)||_\infty \le ||v_0(x)||_\infty + ||s(x)v_1(x)||_\infty \le 2^{t-5} + n/2.$$

It follows that

$$\begin{aligned} ||w_3(x)||_\infty &\le \frac{r}{q}||w(x)||_\infty + ||v_0(x) - s(x)v_1(x)||_\infty + ||m(x)||_\infty \\ &\le \frac{5r}{82q}D_q + 2^{t-5} + \frac{n}{2} + 1 \\ &\le \frac{5}{82}D_r + \frac{r}{2^6} + \frac{n}{2} + 1 < \frac{D_r}{4}, \end{aligned}$$

where the last inequality holds provided $r \ge 16n$ and $n > 164$. □

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

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

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. **Suggested parameters.** In Figures 4, we list some parameters that satisfy the conditions in Theorem 3.3, Lemma 4.1 and Lemma 4.2. We choose primes q and Q so that

$$r|(q-1), \qquad r|(Q-1),$$

hence FFT can be used in computing products of polynomials in $R_{n,q}$ and $R_{m,Q}$. 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 second row under q and Q indicates the bit size of q and Q, respectively. The row for bk indicates the bit size of bootstrapping keys. It is also possible to pick Q as a product of two primes so that FFT can be used, but the details are omitted here.

n	$2^9 = 512$	$2^{10} = 1024$	$2^{11} = 2048$	$2^{12} = 4096$
r	16n	16n	16n	16n
$q-1$	$r(41n+20)$ 27 bits	$r(41n+9)$ 29 bits	$r(41n+11)$ 31 bits	$r(41n+25)$ 33 bits
$Q-1$	$r(\rho r^3 n^2 + 19)$ 81 bits	$r(\rho r^3 n^2 + 85)$ 87 bits	$r(\rho r^3 n^2 + 89)$ 93 bits	$r(\rho r^3 n^2 + 31)$ 99 bits
c_s	6	6	6	6
c_{pk}	19	20	21	22
bk	162 MB	696 MB	2976 MB	12672 MB

FIGURE 4. Suggested Parameters for private-key and public-key encryptions and for bootstrapping, where $\rho = 1220$. The row of c_s gives the ciphertext expansion under private-key encryption, the row of c_{pk} is for ciphertext expansion for public-key encryption, and the row of bk is for the size of bootstrapping keys.

4.4. **Efficiency of bootstrapping.** Figure 5 below lists the parameters used by the current paper and by Ducas and Micciancio [51] and Chillotti et al. [39, 40] when $n = 512$. In [39, 40, 51], they use $n = 500$, however, their bootstrapping algorithms work well for $n = 512$ (with other parameters the same). The input LWE ciphers are assumed to be in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ where r is listed in the table (after rescaling in [39, 40, 51]. For a bootstrapping key $bk = (C_0, C_1, \ldots, C_{n-1})$, in [51, 39] each C_i is a 6×2 matrix over $R_{m,Q}$ and $\mathbb{R}_m$ (respectively), in [40] each C_i is a 4×2 matrix over $\mathbb{R}_m$ with real numbers of 32 bits as coefficients, and in the current paper each C_i is a 4×2 matrix over $R_{m,Q}$. The column under FFT lists the number of FFTs of length m that need to be computed in the whole bootstrapping algorithm, and the column under Failure is for the failure probability, that is, the probability that the computed new ciphertext is not a valid LWE cipher (to participate in further homomorphic computing. For our bootstrapping algorithm, it always outputs new LWE ciphers with error at most $D_r/4$, hence there is no failure at all, however, in [51], it has a probability of at most 2^{-31} that the new LWE cipher has error bigger than $D_r/4$, similarly in [39, 40], the failure probability is at most 2^{-33} (more precisely $2^{-33.56}$). The probability 2^{-33} is very small, hence useful for computing many functions that need fewer than $2^{26} = 67,108,864$ bit operations (with failure probability at most $1/128 < 0.008$). However, their bootstrapping algorithm (for $n = 512$) can not be used to compute functions that need more than 2^{33} bit operations.

As pointed out in [39], 90% of the time of the bootstrapping algorithm is on computing FFTs. Our FFT has length 4 times that of [51, 39, 40], and we use the same number of FFTs as in [40]. Note that, in [40], FFT is computed over complex numbers which have about 53 bits in binary floating points, while our FFT is computed modulo Q which has 81 bits (for $n = 512$), we use the rough estimate that each operation modulo Q is 2.5 times as expensive as the corresponding complex number operation with 53 bits.[1] This means that our bootstrapping algorithm can be implemented with running time about 10 times of [40]. By [40], each homomorphic bit operation can be computed in 13ms. We estimate that our bootstrapping algorithm can compute three LWE ciphers $E_{\mathbf{s}}(x_1 \wedge x_2)$, $E_{\mathbf{s}}(x_1 \vee x_2)$ and $E_{\mathbf{s}}(x_1 \oplus x_2)$ in about 130ms. Of course, this needs to be verified by careful implementation, and further optimization should be investigated. We hope to do a careful hardware implementation in ASIC (application-specific integrated circuit) in the near future.

4.5. **Homomorphic computing of arbitrary functions.** Let $f : \{0,1\}^N \to \{0,1\}^M$ be an arbitrary function with input length N and output length M. We suppose that f is given as a Boolean circuit, or a pseudocode (with restricted for-loops and while-loops, see below). In practice, f can be any search function on a large data base $\mathbf{x} \in \{0,1\}^N$, hence N is usually large (say $N = 8 \cdot 2^{40} = 8,796,093,022,208$ if the data base has size 1TB). Without loss of generality, we assume that $N = nd$ where $n = 512$ (or any of 1024 and 2048). The data $\mathbf{x}$ is divided into d blocks of length n, that is, $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_d)$ where $\mathbf{x}_k = (x_{k,0}, x_{k,1}, \ldots, x_{k,n-1}) \in \{0,1\}^n$, and each block is converted into a polynomial $\sum_{i=0}^{n-1} x_{k,i}x^i \in R_n$, then encrypted using the private-key encryption scheme or the public-key encryption scheme described in the previous section:

$$\mathbf{c}_k = \mathrm{RE}_{\mathbf{s}}(\mathbf{x}_k), \text{ or } \mathrm{RE}_{pk}(\mathbf{x}_k), \quad 1 \le k \le d.$$

[1]The estimation is based on the following argument. We can pick $Q = Q_1Q_2$ where Q_1 and Q_2 are two distinct primes with about 41 to 50 bits for n between 512 and 4096. By the Chinese remainder theorem, each operation modulo Q is equivalent to the corresponding operation modulo Q_1 and Q_2, separately. So each multiplication modulo Q is equivalent to two multiplications modulo a prime number with at most 50 bits. On the other hand, for complex numbers of the form $a+jb$ where $j = \sqrt{-1}$ and a and b are real numbers with 53 bits, one multiplication needs four multiplications and two additions (or three multiplications and 7 additions) of real numbers with 53 bits. Hence it is an overestimate that each operation modulo Q, where Q has 80 to 100 bits, costs about 2.5 times that of complex numbers with 53 bits.

	r	m	Q	bk	#FFT	Failure
[51]	2048	1024	32 bits	1032 (+314) MB	48000	2^{-31}
[39]	2048	1024	$\mathbb{C}53$	23 (+29) MB	4006	2^{-33}
[40]	2048	1024	$\mathbb{C}53$	16 (+29) MB	3072	2^{-33}
Ours	8192	4096	81 bits	162 MB	3072	0

FIGURE 5. Parameter comparison for Bootstrapping with $n = 512$ where $\mathbb{C}53$ means operations on complex numbers $a + bj$ where $j = \sqrt{-1}$ and a and b are real numbers with 53 bits in binary floating points. In all four cases, FFTs used in computing the external product have length m, and the number of FFTs used is indicated in the column under #FFT. The column under bk lists the size of bootstrapping keys where the numbers in the parenthesis denote the size of key-switch keys used in their papers. The last column is for the failure probability.

Note that the total bit size of all $\mathbf{c}_k$'s is about $6N$ bits with private-key encryption (and using a pseudo random number generator), or $20N$ bits with public-key encryption. The ciphertexts $\mathbf{c}_k$, $1 \le k \le d$, are computed once and stored on a cloud server (or any where in the world) that is reliable (that is, the encrypted data is available and is error free any time), but not necessarily trustful.

Suppose the bootstrapping key $bk = (C_0, C_1, \dots, C_{n-1})$ is public. The cloud server (in fact anyone, including hackers) can perform homomorphic computing in three steps:

(a) unpacking the ciphertexts $\mathbf{c}_1, \cdots, \mathbf{c}_d$ to get LWE ciphers in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ for the bits of $\mathbf{x}$,
(b) homomorphic computing of f on the LWE ciphers, using the bootstrapping algorithm,
(c) packing the resulted LWE ciphers into RLWE ciphers in $R_{n,r}^2$.

We describe these steps in details below.

4.5.1. *Unpacking.* First extract LWE ciphers for the bits of $\mathbf{x}_k$ from $\mathbf{c}_k$ for $1 \le k \le d$. Suppose $\mathbf{c}_k$ is from private-key encryption. Then $\mathbf{c}_k$ is of the form $\mathbf{c}_k = (u_k, v_k)$ where $u_i \in \{0,1\}^n$ and $v_i \in \left(\{0,1\}^5\right)^n$. Apply the pseudo random number generator P to u_k to get a polynomial $a_i(x) = P(u_k, x) \in R_{n,r}$, and convert v_i into a polynomial $b_i(x) \in R_{n,r}$. By Lemma 4.1, we have

$$2^{t-4} b_k(x) \equiv a_k(x)s(x) + \left(\sum_{i=0}^{n-1} x_{k,i} x^i\right) D_r + w_k(x) \bmod (x^n + 1, r),$$

where $\|w_k(x)\|_\infty < D_r/4$. For $0 \le i \le n-1$, an LWE cipher for $x_{k,i}$ is

$$E_{\mathbf{s}}(x_{k,i}) = (\text{Extract}(a_k(x), i), 2^{t-4} b_{k,i}) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$$

with error size $< D_r/4$, where $b_{k,i}$ is the coefficient of x^i in $b_k(x)$.

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

$$2^{t-5} b_k(x) \equiv a_k(x)s(x) + \left(\sum_{i=0}^{n-1} x_{k,i} x^i\right) D_r + w_k(x) \bmod (x^n + 1, r),$$

where $\|w_k(x)\|_\infty < D_r/4$. Hence, for $0 \le i \le n-1$, an LWE cipher for $x_{k,i}$ is

$$E_{\mathbf{s}}(x_{k,i}) = (\text{Extract}(a_k(x), i), 2^{t-5} b_{k,i}) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$$

with error size $< D_r/4$, where $b_{k,i}$ is the coefficient of x^i in $b_k(x)$.

4.5.2. *Homomorphic computing.* Let $f(\mathbf{x}) = \mathbf{y} = (y_1, \ldots, y_M) \in \{0,1\}^M$ an arbitrary function. It is desired to compute a valid ciphertext for $\mathbf{y}$ from the ciphertexts $\mathbf{c}_k$, $1 \le k \le d$, using the bootstrapping algorithm from the previous section. Suppose f is given as a Boolean circuit with N input bits and M output bits. The cloud server can use our bootstrapping algorithm to compute an LWE cipher for each gate of the circuit starting from the input to output, obtaining M LWE ciphers for $y_1, \ldots, y_M$, one for each. These LWE ciphers will be the ciphertext for $f(\mathbf{x})$. Since our bootstrapping algorithm does not produce invalid LWE ciphers at all, one can always get the correct answer $\mathbf{y} = f(\mathbf{x})$ by decoding the M LWE ciphers. Note that the computed ciphertext for $f(\mathbf{x})$ has a larger ciphertext expansion factor. This should not be a problem in practice since an answer is usually small (say from a few bits to a few thousand bits). However, when M is large, we need to pack these LWE ciphers into RLWE ciphers; see next subsection for more details.

Now suppose f is given as a pseudocode which may contain if-statements, for-loops and while-loops, among other statements. We show below how to homomorphically compute f, demonstrated by a few examples.

Homomorphic if-statements. We show how to homomorphically compute if-statements. We demonstrate by a simple case:

$$\text{“if } \mathbf{a} \ne \mathbf{b} \text{ then } \mathbf{u} \text{ else } \mathbf{v}\text{”}$$

where $\mathbf{a} = (a_1, \ldots, a_m) \in \{0,1\}^m$, $\mathbf{b} = (b_1, \ldots, b_m) \in \{0,1\}^m$ and $\mathbf{u} = (u_1, \ldots, u_k) \in \{0,1\}^k$, $\mathbf{v} = (v_1, \ldots, v_\ell) \in \{0,1\}^\ell$. Suppose we have computed an LWE cipher for each bit of $\mathbf{a}$ and $\mathbf{b}$. How do we compute the output of this if-statement? Let

$$z = \bigvee_{i=1}^{m} (a_i \oplus b_i).$$

Then $z = 1$ iff $\mathbf{a} \ne \mathbf{b}$. We can compute an LWE cipher $E_{\mathbf{s}}(z) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ using $2m-1$ BT operations, that is, calling the bootstrapping algorithm $2m-1$ times. If $k \ne \ell$, we need to pad $\mathbf{u}$ or $\mathbf{v}$ by 0's so that they have the same length, hence we may assume that $\ell = k$. Let

$$\mathbf{w} = z\mathbf{u} \oplus (1-z)\mathbf{v} = (z \wedge u_1 \oplus (1-z) \wedge v_1, \ldots, z \wedge u_k \oplus (1-z) \wedge v_k).$$

Then $\mathbf{w} = \mathbf{u}$ if $z = 1$ and $\mathbf{w} = \mathbf{v}$ if $z = 0$, as required by the if-statement. If $\mathbf{u}$ and $\mathbf{v}$ are in plaintext, an LWE cipher for each bit $z \wedge u_i \oplus (1-z) \wedge v_i$ is simply $u_i E_{\mathbf{s}}(z) + v_i(1 - E_{\mathbf{s}}(z))$, which still has error size $< D_r/4$. If the bits of $\mathbf{u}$ and $\mathbf{v}$ are given as LWE ciphers with error size $< D_r/4$, then an LWE cipher for each bit $z \wedge u_i \oplus (1-z) \wedge v_i$ can be computed with three BTs. Hence this if-statement can be computed using $(2m-1) + 3\min\{\ell, k\}$ BTs.

Homomorphic for-loop: Integer addition. Let $a = (a_0, \ldots, a_{m-1})$ and $b = (b_0, \ldots, b_{m-1})$ be two integers in binary representation where a_0 and b_0 are the least significant bits. Suppose we are given LWE ciphers for all the bits of a and b. We want to compute an LWE cipher for each bit of their sum. Let $c = (c_0, \ldots, c_m)$ represent their sum. We have the following pseudocode for computing the bits of c:

Integer addition:
$c_0 := a_0 \oplus b_0$; $\quad z := a_0 \wedge b_0$;
for i from 1 to $m-1$ do
    $t_1 := a_i \oplus b_i$; $\quad t_2 := a_i \wedge b_i$;
    $c_i := t_1 \oplus z$; $\quad t_3 := t_1 \wedge z$;
    $z := t_2 \oplus t_3$;
end for
$c_m := z$;
Return $(c_0, c_1, \ldots, c_m)$.

In the first line, z represents the carry of $a_0 + b_0$. The three lines in the for-loop compute the sum $a_i + b_i + z$, certainly $c_i = (a_i \oplus b_i) \oplus z$ is the least significant bit, and the carry bit is

$$t_2 \oplus t_3 = (a_i \wedge b_i) \oplus (a_i \oplus b_i) \wedge z = (a_i \wedge b_i) \oplus (a_i \wedge z) \oplus (b_i \wedge z),$$

which is 1 if and only if two or three of a_i, b_i, z are equal to 1. Note that each line of the pseudocode can be computed by one BT, namely, using our bootstrapping algorithm once. Hence $a + b$ can be computed by $3m - 2$ BTs.

Homomorphic while-loop: Integer comparison. Let $a = (a_0, \ldots, a_{m-1})$ and $b = (b_0, \ldots, b_{m-1})$ be two integers as above. We show how to homomorphically test if $a \geq b$ (as integers), that is, compute $z \in \{0, 1\}$ so that $z = 1$ iff $a \geq b$. A pseudocode could be

Integer comparison: $z := (a \geq b)$
$z := 1$; $i := m - 1$;
while $i \geq 1$ and $a_i = b_i$ do $i := i - 1$ end while
if $a_i < b_i$ then $z := 0$ end if
Return z.

The while-loop is hard to implement homomorphically, since there is no priori bound on the largest i with $a_i \neq b_i$. We introduce a trick to convert the while-loop into a for-loop as indicated in the following pseudocode:

Homomorphic comparison: $z := (a \geq b)$
$z := 1$; $v := 1$; ($v = 0$ indicates when the above while-loop finishes)
for i from $m - 1$ down to 0 do
  $t := v \wedge (a_i \oplus b_i)$;
  $v := (1 - t) \wedge v$;
  $z := [t \wedge (a_i \vee (1 - b_i))] \oplus [(1 - t) \wedge z]$;
end for
Return z.

Note that $a_i \oplus b_i = 1$ iff $a_i \neq b_i$, and $a_i \vee (1 - b_i) = 1$ iff $a_i \geq b_i$. Hence v changes from 1 to 0 at the largest i with $a_i \neq b_i$, and it never change back to 1 after that; also, when $v = 0$, all the future t will always be 0, hence z will always stay the same. This pseudocode can be directly implemented homomorphically. (Note that LWE ciphers for $a_i \oplus b_i$ and $a_i \vee (1 - b_i)$ can be computed by one BT, since $1 - (a_i \oplus b_i) = a_i \oplus (1 - b_i)$, and one BT can compute both $a_i \oplus (1 - b_i)$ and $a_i \vee (1 - b_i)$.)

In summary, we see from these examples that one can homomorphically compute any pseudocode that contains if-statements, for-loops and while-loops, the only restriction is that the number of iterations executed in the for-loops and while-loops must be upper bounded by numbers that do not depend on encrypted data. The example for integer comparison shows a trick on how to convert a while-loop for which the number of executed times depends on encrypted data into a for-loop for which the number of executed times is independent of encrypted data. Under these restrictions on the for-loops and while-loops, any pseudo code can be computed homomorphically. Certainly, more careful study is needed on how to convert an arbitrary algorithm into restricted pseudocode and on how to compute them more efficiently.

4.5.3. *Packing of LWE ciphers.* Suppose the output y of a function $y = f(x)$ has M bits. Then the ciphertext computed for y will be a sequence of M LWE ciphers. We show below how we can pack any n LWE ciphers in $\mathbb{Z}_r^n \times \mathbb{Z}_r$ into one RLWE cipher in $R_{m,r}^2$ where $m = 8n$, hence the size of the ciphertext of y will be reduced by a factor of $n/16$.

Lemma 4.3. *Suppose $y_0, \ldots, y_{n-1} \in \{0,1\}$ and we have n LWE ciphers $E_s(y_i) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$ with error size $< D_r/4$ for $0 \le i \le n-1$. With the bootstrapping key bk, one can compute an RLWE cipher $(w(x), v(x)) \in R_{m,r}^2$ for $m(x) = \sum_{i=0}^{n-1} y_i x^i$ with error $< D_r/4$, that is,*

$$v(x) \equiv w(x)s(x) + z(x) + m(x)D_r \pmod{x^m+1, r},$$

where $z(x) \in \mathbb{R}_m$ with $\|z(x)\|_\infty < n \le D_r/4$.

Proof. Set $E_s(1) = (0, D_r) \in \mathbb{Z}_r^n \times \mathbb{Z}_r$, the trivial LWE cipher with no error. For each $0 \le i \le n-1$, we compute $1 \cdot y_i$ homomorphically using the bootstrapping algorithm in Section 3, where the Step 5 will be skipped (and ignore $\mathbf{a}_2$ and $\mathbf{a}_3$ there), giving us LWE ciphers $(\mathbf{a}_i, b_i) \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$ so that

$$b_i \equiv \langle \mathbf{a}_i, s\rangle + y_i D_Q + \epsilon_i \pmod{Q}$$

where $\epsilon_i \in \mathbb{Z}$ and $|\epsilon_i| < 4Bnr\ell\tau_1 = 8Bn^2r$ (since $\ell = 2$ and $\tau_1 = n$ in the the bootstrapping key C_i) for $0 \le i \le n-1$. Now form

$$\mathbf{a}(x) = \sum_{i=0}^{n-1} \mathbf{a}_i x^{i-1} = (a_0(x), \ldots, a_{n-1}(x)) \in R_{m,Q}^n, \quad \text{and} \quad b(x) = \sum_{i=0}^{n-1} b_i x^i.$$

Then

$$b(x) \equiv \sum_{i=0}^{n-1} a_i(x)s_i + m(x)D_Q + \epsilon(x) \pmod{Q}, \tag{16}$$

where $\epsilon(x) = \sum_{i=0}^{n-1} \epsilon_i x^i$ and $\|\epsilon(x)\|_\infty \le 8Bn^2r$.

Next, we need to compute the sum $\sum_{i=0}^{n-1} a_i(x)s_i$ homomorphically. For each s_i, we take the last two rows of the bootstrapping key $C_i = \mathrm{GSW}(s_i)$ (as in Section 4.1), which gives us $a_{ji}(x)$ and $b_{ji}(x) \in \mathbb{R}_{m,Q}$ so that

$$\begin{pmatrix} b_{3i}(x) \\ b_{4i}(x) \end{pmatrix} \equiv \begin{pmatrix} a_{3i}(x) \\ a_{4i}(x) \end{pmatrix} s(x) + s_i \begin{pmatrix} 1 \\ B \end{pmatrix} + \begin{pmatrix} e_{3i}(x) \\ e_{4i}(x) \end{pmatrix} \pmod{x^m+1, Q}, \tag{17}$$

where $\|e_{ji}(x)\|_\infty < n$. We define a shortened external product: for any $a(x)$, $b_1(x)$ and $b_2(x)$ from $R_{m,Q}$, define

$$a(x) \odot \begin{pmatrix} b_1(x) \\ b_2(x) \end{pmatrix} = (u_1(x), u_2(x)) \begin{pmatrix} b_1(x) \\ b_2(x) \end{pmatrix} = u_1(x)b_1(x) + u_2(x)b_2(x) \in R_{m,Q},$$

where $u_1(x)$ and $u_2(x)$ are random so that $a(x) = u_1(x) + u_2(x)B$ with $\|u_i(x)\|_\infty \le 2B$. Let's apply to the bootstrapping key in (17):

$$\begin{aligned} v_i(x) := a(x) \odot \begin{pmatrix} b_{3i}(x) \\ b_{4i}(x) \end{pmatrix} &= (u_1(x), u_2(x)) \begin{pmatrix} b_{3i}(x) \\ b_{4i}(x) \end{pmatrix} \\ &\equiv (u_1(x), u_2(x)) \left[\begin{pmatrix} a_{3i}(x) \\ a_{4i}(x) \end{pmatrix} s(x) + s_i \begin{pmatrix} 1 \\ B \end{pmatrix} + \begin{pmatrix} e_{3i}(x) \\ e_{4i}(x) \end{pmatrix} \right], \\ &\equiv w_i(x)s(x) + a(x)s_i + z_i(x), \end{aligned}$$

where

$$w_i(x) = u_1(x)a_{3i}(x) + u_2(x)a_{4i}(x) = a(x) \odot \begin{pmatrix} a_{3i}(x) \\ a_{4i}(x) \end{pmatrix} \in R_{m,Q}$$

and $z_i(x) = u_1(x)e_{3i}(x) + u_2(x)e_{4i}(x)$ with $\|z_i(x)\|_\infty \le 2 \cdot 2Bnm = 2Bnr$.

Now back to (16). We compute

$$(\tilde{w}(x), \tilde{v}(x)) := \sum_{i=0}^{n-1} a_i(x) \odot \begin{pmatrix} a_{3i}(x) & b_{3i}(x) \\ a_{4i}(x) & b_{4i}(x) \end{pmatrix} = \sum_{i=0}^{n-1} (w_i(x), v_i(x)).$$

Then

$$\tilde{v}(x) \equiv \tilde{w}(x)s(x) + \sum_{i=0}^{n-1} a_i(x)s_i + \tilde{z}(x), \tag{18}$$

where $\tilde{z}(x) = \sum_{i=0}^{n-1} z_i(x)$ with $\|\tilde{z}(x)\| \le 2Bn^2r$. By (16) and (18), we have

$$b(x) - \tilde{v}(x) \equiv -\tilde{w}(x)s(x) + m(x)D_Q + (\epsilon(x) - \tilde{z}(x)) \pmod{x^m + 1, Q}. \tag{19}$$

Let

$$\tilde{w}_1(x) = -w(x), \quad \tilde{v}_1(x) = b(x) - \tilde{v}(x), \quad \tilde{z}_1(x) = \epsilon(x) - \tilde{z}(x).$$

Then $\|\tilde{z}_1(x)\|_\infty \le 8Bn^2r + 2Bn^2r \le 10Bn^2r$ and (19) becomes

$$\tilde{v}_1(x) \equiv \tilde{w}_1(x)s(x) + m(x)D_Q + \tilde{z}_1(x) \pmod{x^m + 1, Q}, \tag{20}$$

where $\|\tilde{z}_1(x)\|_\infty \le 10Bn^2r$. Finally, do modulus reduction:

$$w(x) = \left\lfloor \frac{r}{Q}\tilde{w}_1(x) \right\rceil, \quad v(x) = \left\lfloor \frac{r}{Q}\tilde{v}_1(x) \right\rceil,$$

both are in $R_{m,r}$. Then (20) becomes

$$v(x) \equiv w(x)s(x) + m(x)D_r + z(x) \pmod{x^m + 1, r}, \tag{21}$$

for some $z(x) \in R_m$.

We need to estimate the coefficient size of $z(x)$. By our choice,

$$w(x) = \frac{r}{Q}\tilde{w}_1(x) + h_1(x), \quad v(x) = \frac{r}{Q}\tilde{v}_1(x) + h_2(x),$$

where $h_i(x) \in \mathbb{R}_m[x]$ and $\|h_i(x)\|_\infty \le 1/2$ for $i = 1, 2$. Then (20) becomes

$$v(x) \equiv w(x)s(x) - h_1(x)s(x) + h_2(x) + m(x)\frac{r}{Q}D_Q + \frac{r}{Q}\tilde{z}_1(x),$$

Hence, in (21), we must have

$$z(x) = h_2(x) - h_1(x)s(x) + m(x)\left(\frac{r}{Q}D_Q - D_r\right) + \frac{r}{Q}\tilde{z}_1(x).$$

Thus

$$\begin{aligned} \|z(x)\|_\infty &\le \frac{1}{2}n + \frac{1}{2} + 1 + \frac{r}{Q}10Bn^2r \\ &\le \frac{1}{2}(n+3) + \frac{10r}{1220r^4n^2}(35r^2n)n^2r \\ &\le \frac{1}{2}(n+3) + \frac{35}{122}n < n. \end{aligned}$$

This completes the proof. □

5. Security analysis

In this section, we give a brief analysis on the security of our homomorphic encryption scheme. The security model is IND-CPA, that is, our scheme is secure under chosen plaintext attack, which could be interactive. This means that an adversary can choose plaintexts and get their ciphertexts, as many such plaintext and ciphertext pairs as desired (but bounded by a polynomial in n), then try to distinguish (say with success probability $1/10$) whether a given ciphertext is an encryption of 0 or 1, or to distinguish a given valid ciphertext from uniform random bit strings (of the the same format). This is equivalent to solving hard lattice problems by Regev's theorem. However, in the reduction proof of Regev's theorem, there is a tightness gap that may be very large as explained in [32]. Hence we need to estimate the concrete complexity of all current attacks on our scheme with the proposed parameters.

In our scheme, according to Figure 4, n is a power of 2 and we have RLWE ciphertexts over $\mathbb{Z}_q$ for three choices of q:

Type 1. $q = r = 16n$ and the error size is bounded by n: corresponding to ciphertexts in $R^2_{n,q}$ from private-key and public-key encryptions of the original data (see Lemmas 4.1 and 4.2);

Type 2. $q \approx 41rn = 656n^2$ and the error size is bounded by $D_q/(41n) \approx 4n$: corresponding to the public key $pk = (k_0(x), k_1(x)) \in R^2_{n,q}$;

Type 3. $q = Q \approx 1220r^4n^2 \approx 2^{27}n^6$ and the error size is bounded by n: corresponding to bootstrapping key $C_i \in R^2_{m,Q}$, $1 \le i \le n$, of the bootstrapping key where $m = 8n$.

Hence finding the secret key $s(x)$ is equivalent to solving the RLWE problem in the following cyclotomic rings:

$$R_{n,q} = \mathbb{Z}[x]/(x^n + 1, q), \quad R_{m,Q} = \mathbb{Z}[x]/(x^{8n} + 1, Q)$$

where q is either $16n$ (a power of 2) or a prime, and Q is a prime (or a product of two primes in practical implementation).

5.1. **Number theoretic attack and circular security issue.** 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 [52, 53, 31, 34, 35, 36] 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, the attack is likely to work provided the order $\le \log_2(q)$ (and $\tau\sqrt{n}$ is not too large compare to q where τ is the error size). For our two rings, when n is a power of 2, $x^n + 1 \equiv (x-1)^n \bmod 2$, but $x^n + 1$ is irreducible modulo 2^k for all $k > 1$.[2] When $(2n)|(q-1)$ and q is a prime, the polynomial $x^n + 1$ factors completely over $\mathbb{Z}_q$, and each root of $x^n + 1$ has multiplicative order exactly $2n$. Hence $x^n + 1$ modulo q has no factor of small degrees whose roots have small orders. Similarly for $X^{8n} + 1$ over $\mathbb{Z}_q$ where $(16n)|(q-1)$ (where q may be much smaller than Q after a modulus reduction). Hence the number theoretic attack can not be applied to our rings.

The bootstrapping key is an encoding of the bits of secret key by the secret key itself. This is called circular encoding. There is a concern whether this kind of circular ciphers could be easier to break. That remains an open question. For the positive side, the bootstrapping key enable any third party to compute new LWE ciphers from any given LWE ciphers in $\mathbb{Z}^n_r \times \mathbb{Z}_r$ with the error size all bounded by n, hence yielding a fully homomorphic encryption scheme. There is another benefit: by Lemma 4.3, LWE ciphers in $\mathbb{Z}^n_r \times \mathbb{Z}_r$ can be packed into RLWE ciphers in $R^2_{m,r}$, hence we have the following interesting result.

Corollary 5.1. *Let the parameters be as in Section 4.1 with $r = 16n$. Given the bootstrapping key bk, the LWE problem in $\mathbb{Z}^n_r \times \mathbb{Z}_r$ is equivalent to the RLWE problem in $R^2_{m,r}$ where $m = 8n$ and the error size in both cases is bounded by n.*

The negative side of the bootstrapping key, however, is that our encryption scheme is not secure under chosen ciphertext attack. In fact, for any Boolean function $\omega(s_0, s_1, \ldots, s_{n-1})$ (say the sum of a random subset of the s_i's modulo 2), any third party can use the bootstrapping key to compute a random LWE cipher

$$y = E_{\mathbf{s}}(\omega(s_0, s_1, \ldots, s_{n-1})) \in \mathbb{Z}^n_r \times \mathbb{Z}_r.$$

Then send y to the owner of the secret key $\mathbf{s}$ to decode, and the owner releases the value $\omega(s_0, s_1, \ldots, s_{n-1})$. One just needs $n + O(1)$ such values to completely determine $\mathbf{s}$. Hence

[2] Thanks to Professor Daqing Wan for his proof, personal communications.

the moral is that, for homomorphic encryption scheme with bootstrapping key, *never decode for anyone but yourself!*

5.2. **Main attacks on the LWE problem.** Except the weak instances of rings mentioned above, it is not known how to use the ring structure to solve the RLWE problem over our two rings. Currently the best approach is to extract the LWE ciphers from RLWE ciphers and solve the LWE problem. The main approaches for solving the LWE problem are briefly described below.

Meet-in-the-middle attack. Since our polynomial $s(x)$ has binary coefficients, there are 2^n choices for $s(x)$. Exhaustive search requires 2^n operations in $R_{n,q}$. A better approach is the modified meet-in-the-middle attack by [76, 13, 9, 29], which uses about $2^{0.337n}$ operations in $R_{n,q}$(ignoring some log factor). Since $n \geq 512$, this requires at least 2^{170} operations in $R_{n,q}$.

Gröbner basis attack. When the coefficients of error polynomials are small, one can solve $s(x)$ by computing certain Gröbner basis [8, 56]. However, our error size is n or $4n$, this approach requires at least $n^{2n} = 2^{2n \log_2 n}$ operations in $\mathbb{Z}_q$, which is worst than the meet-in-the-middle attack.

Lattice basis reduction attacks. A much more powerful attack is to use the lattice basis reduction algorithm (LLL) due to Lenstra, Lenstra and Lovasz (1982, [82]); see [93] for its practical performance and [91, 92] for its recent improvements. There is also a BKZ variation [100, 37], which uses SVP oracles [90, 74, 104, 105, 80, 81, 14]. There are several approaches that can reduce LWE problems over $\mathbb{Z}_q$ to lattice problems over $\mathbb{Z}$, including the SIS method [1, 89, 84], the BKW method [17, 3, 4, 5, 50, 69, 79], the bounded distance decoding (BBD) method [77, 84, 85, 9].

The paper by Albrecht, Player and Scott [6] give a nice survey on these methods and give concrete complexity analysis, including the case when the secret vector $\mathbf{s}$ is binary (as we use in our scheme, see also [2]). We shall use their analysis directly, particularly Table 5 in [6], which assumes that $q \approx n^2$, the secret vector is binary and the error size is about $q\alpha \approx n^{1.5}/(\log_2 n)^2$. Note that, for $2^{10} \leq n \leq 2^{16}$, we have

$$n^{1.5}/(\log_2 n)^2 \leq n.$$

These parameters match those of Type 2 ciphertexts above. Also, by modulus reduction, Type 3 ciphertexts can be reduced to Type 2 ciphertexts with error size $< n$. By Table 5, we see that, for $n = 512$, the best approach is the sieve method in [9], which needs at least 2^{168} bit operations; for $n = 1024$, the best approach is the BKW method, which requires at least 2^{481} bit operations. For Type 1 ciphertexts, since the error is as big as $q/16$, the complexity can only be bigger. We hope to add more detail of analysis for Type 1 and for larger values of n in the full version of the paper.

6. Use cases

Fully homomorphic encryption scheme is a powerful tool that can solve many problems in IT computing, for examples, outsourcing computation [42, 59, 49, 67], verifiable computing [58, 15, 96, 12], secure multi-party computation [46, 47, 45, 47, 83, 24, 71, 47, 83, 24, 71, 66] maliciously circuit-private FHE [95, 41], Functional encryption and program obfuscation [18, 11, 19, 57], zero-knowledge proof and homomorphic signature [16, 66, 68]. Below we describe a protocol for secure two-party computation and zero knowledge proof of any NP language.

Secure two-party computation and zero knowledge proof. For zero knowledge proof of any language in NP, Barak and Brakerski [10] proposed a protocol in their blog [10], see also [70]. Here we present a different protocol where the prover performs heavy computing but the verifier does only light computing, which is often desired in verifiable

computing (especially on blockchains). Let L be any language in NP and let $R_L(x, y)$, whose value is either 1 or 0, be a relation defining L:

$$L = \{x \in \{0,1\}^* : \exists y \in \{0,1\}^* \text{ so that } R_L(x, y) = 1\}.$$

Alice and Bob both know the relation $R_L(x, y)$ and assume $R_L(x, y)$ can be computed in polynomial time. Suppose Alice has a secret string x for which Bob does not know, Bob claims that he knows a secret y so that $R_L(x, y) = 1$, hence $x \in L$. Neither Alice nor Bob wants to leak any information on x or y to the other party. This means that they want to compute the function $R_L(x, y)$ without leaking any information of x or y.

Using our FHE scheme with $n \geq 512$, Bob (Prover) can prove to Alice (Verifier) by the following protocol. First, Alice picks a random secret key $\mathbf{s} \in \{0,1\}^n$ and publishes a corresponding bootstrapping key bk (and a public key, but not needed here), so Bob knows bk. Fix t to be an integer ≥ 500.

Protocol for Zero Knowledge Proof of any Language in NP

Step 1. For $1 \leq i \leq t$, Alice picks $r_i \in \{0,1\}$ uniform randomly, $x_i \notin L$ randomly (with the same length as x)[3], both independently, let

$$u_i := r_i x \oplus (1 - r_i)x_i = \begin{cases} x, & \text{if } r_i = 1, \\ x_i, & \text{if } r_i = 0, \end{cases}$$

and computes a ciphertext $c_i := E_{\mathbf{s}}(u_i)$. Then send $c_1, \ldots, c_t$ to Bob.

Step 2. For $1 \leq i \leq t$, Bob picks $w_i \in \{0,1\}$ uniform randomly and $y_i \in \{0,1\}^*$ randomly (with the same length as y), let

$$v_i := w_i y \oplus (1 - w_i)y_i = \begin{cases} y, & \text{if } w_i = 1, \\ y_i, & \text{if } w_i = 0, \end{cases}$$

and uses the bootstrapping key bk to compute $R_L(u_i, v_i)$ homomorphically from c_i and the trivial encryption of v_i to get a ciphertext

$$b_i := E_{\mathbf{s}}(R_L(u_i, v_i)).$$

Then send $b_1, \ldots, b_t$ to Alice.

Step 3. Alice decodes b_i to get $z_i \in \{0,1\}$ for $1 \leq i \leq t$. Alice accepts only if ($r_i \geq z_i$ for all i) and ($r_i = z_i = 1$ for at least $t/5$ values of i).

Completeness. Suppose Bob knows a y so that $R_L(x, y) = 1$. Note that, for each $1 \leq i \leq t$, there are three cases in the above protocol:

(a) $r_i = 0$: $z_i = R_L(u_i, v_i) = R_L(x_i, v_i) = 0$, since $x_i \notin L$;
(b) $r_i = 1$ and $w_i = 0$: $z_i = R_L(u_i, v_i) = R_L(x, y_i)$, hence z_i can be 0 or 1;
(c) $r_i = 1$ and $w_i = 1$: $z_i = R_L(u_i, v_i) = R_L(x, y) = 1$.

Note that the case $r_i = 0$ and $z_i = 1$ never happen, hence Bob must be cheating if this happens. Also, the probability that $r_i = w_i = 1$ is $1/4$, thus it is expected to have $t/4$ values of i so that $r_i = z_i = 1$, hence with high probability the number of such i is at least $t/5$ (In fact, this probability is exponentially close to 1 by the laws of large numbers).

Soundness. Now suppose Bob does not know any y so that $R_L(x, y) = 1$. Under the security assumption of our FHE scheme, Bob has no way to distinguish $E_{\mathbf{s}}(x)$ from $E_{\mathbf{s}}(x_i)$, and $E_{\mathbf{s}}(x)$ is independent random for each instance of encryption. Bob may try to guess z_i and send its encryption to Alice. The probability that ($r_i \geq z_i$ for all i) and ($r_i = z_i = 1$ for at least $t/5$ values of i) is exponentially small. Or Bob may guess a y and use a different function R' and compute $z_i := R'(u_i, y)$ homomorphically from c_i and the trivial encryption of y. For Alice to accept, Bob must find R' and y so that $R'(x, y) = 1$ and ($R'(x_i, y) = 0$

[3]Note that we only need x_i so that, for random y, $R_L(x_i, y) = 0$ holds with probability exponentially close to 1.

for all i with $r_i = 0$), where $x, x_1, \ldots, x_t$ are unknown to Bob. Note that it is expected that there are $t/2$ values of i with $r_i = 0$. It can be shown that, for any function R' and for random $x, y, x_1, \ldots, x_{t/2}$, the probability that $R'(x, y) = 1$ and $R'(x_i, y) = 0$ for $1 \le i \le t/2$ is at most $1/t$. Hence, for large t, Bob has only a small probability of success. (Ideally, we would want the probability to be exponentially small in t. We leave this as an open problem.)

Zero Knowledge. First suppose Alice is not honest and tries to get information on Bob's secret input y. Alice would pick $x_1, \ldots x_t$ in certain pattern and hope to get the values $R(x_i, y)$ for many different x_i's, then try to reconstruct y. Since Bob returns the values of $R_L(x_i, v_i)$, which is either $R_L(x_i, y)$ or $R_L(x_i, y_i)$ for some random y_i, each with probability $1/2$, Alice has no way to tell which case it is, hence can not deduce any information on y. Next suppose Bob is not honest and tries to get information on Alice's secret input x. Bob only sees the ciphertexts of x and $x_1, \ldots x_t$ for some random x_i's. Since the FHE scheme used is assumed to be semantically secure, Bob can not get any information on x.

7. Conclusions

We presented a compact fully homomorphic encryption scheme with a small cipher expansion that is suitable for practical applications in distributed networks of computers, including IoTs, blockchains and cloud servers. The scheme can protect function privacy and can be used in many applications including outsourced computing, multi-party secure computation, verifiable computing, zero knowledge proof, etc. The bottleneck is that our bootstrapping for homomorphic bit operations is still slow, however, it is suitable for hardware implementation, especially on a cluster of special designed circuits that can perform homomorphic bit operations in parallel. On the theoretical side, there are still many open problems, including solving LWE problems and lattice basis problems, and there is a great need for more careful studying of attacks based on lattice basis-reduction algorithms.

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 R. Albrecht, *On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL*, Cryptology ePrint Archive, Report 2017/047, 2017, https://eprint.iacr.org/2017/047.
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. Sanjeev Arora and Rong Ge, *New algorithms for learning in presence of errors*, Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I, 2011, pp. 403–415.
9. Shi Bai and Steven D. Galbraith, *Lattice decoding attacks on binary LWE*, pp. 322–337, Springer International Publishing, Cham, 2014.
10. Boaz Barak and Zvika Brakerski, *The swiss army knife of cryptography*, Blog document, 2012, http://windowsontheory.org/2012/05/01/the-swiss-army-knife-of-cryptography.

11. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang, *On the (im)possibility of obfuscating programs*, J. ACM **59** (2012), no. 2, 6:1–6:48.
12. Carsten Baum, Ivan Damgård, and Claudio Orlandi, *Publicly auditable secure multi-party computation*, Security and Cryptography for Networks (Cham) (Michel Abdalla and Roberto De Prisco, eds.), Springer International Publishing, 2014, pp. 175–196.
13. Anja Becker, Jean-Sébastien Coron, and Antoine Joux, *Improved generic algorithms for hard knapsacks*, Advances in Cryptology – EUROCRYPT 2011 (Berlin, Heidelberg) (Kenneth G. Paterson, ed.), Springer Berlin Heidelberg, 2011, pp. 364–385.
14. 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.
15. Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza, *SNARKs for C: Verifying program executions succinctly and in zero knowledge*, Advances in Cryptology – CRYPTO 2013 (Berlin, Heidelberg) (Ran Canetti and Juan A. Garay, eds.), Springer Berlin Heidelberg, 2013, pp. 90–108.
16. Rikke Bendlin and Ivan Damgård, *Threshold decryption and zero-knowledge proofs for lattice-based cryptosystems*, Theory of Cryptography (Berlin, Heidelberg) (Daniele Micciancio, ed.), Springer Berlin Heidelberg, 2010, pp. 201–218.
17. 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
18. Dan Boneh, Amit Sahai, and Brent Waters, *Functional encryption: Definitions and challenges*, Theory of Cryptography (Berlin, Heidelberg) (Yuval Ishai, ed.), Springer Berlin Heidelberg, 2011, pp. 253–273.
19. ———, *Functional encryption: A new vision for public-key cryptography*, Commun. ACM **55** (2012), no. 11, 56–64.
20. Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig, *Improved security for a ring-based fully homomorphic encryption scheme*, Cryptography and Coding (Berlin, Heidelberg) (Martijn Stam, ed.), Springer Berlin Heidelberg, 2013, pp. 45–64.
21. 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.
22. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan, *Fully homomorphic encryption without bootstrapping*, Cryptology ePrint Archive, Report 2011/277, 2011, https://eprint.iacr.org/2011/277.
23. 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
24. Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou, *Four round secure computation without setup*, Theory of Cryptography (Cham) (Yael Kalai and Leonid Reyzin, eds.), Springer International Publishing, 2017, pp. 645–677.
25. 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
26. Zvika Brakerski and Vinod Vaikuntanathan, *Fully homomorphic encryption from ring-LWE and security for key dependent messages*, Advances in Cryptology – CRYPTO 2011 (Berlin, Heidelberg) (Phillip Rogaway, ed.), Springer Berlin Heidelberg, 2011, pp. 505–524.
27. ———, *Efficient fully homomorphic encryption from (standard) LWE*, SIAM Journal on Computing **43** (2014), no. 2, 831–871.
28. ———, *Lattice-based FHE as secure as PKE*, Proceedings of the 5th Conference on Innovations in Theoretical Computer Science (New York, NY, USA), ITCS '14, ACM, 2014, pp. 1–12.
29. Johannes Buchmann, Florian Göpfert, Rachel Player, and Thomas Wunderer, *On the hardness of LWE with binary error: Revisiting the hybrid lattice-reduction and meet-in-the-middle attack*, Proceedings of the 8th International Conference on Progress in Cryptology — AFRICACRYPT 2016 - Volume 9646 (New York, NY, USA), Springer-Verlag New York, Inc., 2016, pp. 24–43.
30. Daniel Cabarcas, Florian Göpfert, and Patrick Weiden, *Provably secure LWE encryption with smallish uniform noise and secret*, Proceedings of the 2Nd ACM Workshop on ASIA Public-key Cryptography (New York, NY, USA), ASIAPKC '14, ACM, 2014, pp. 33–42.
31. 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.
32. Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar, *Another look at tightness II: Practical issues in cryptography*, Paradigms in Cryptology – Mycrypt 2016. Malicious and Exploratory

Cryptology (Cham) (Raphaël C.-W. Phan and Moti Yung, eds.), Springer International Publishing, 2017, https://eprint.iacr.org/2016/360, pp. 21–55.
33. Hao Chen, Kim Laine, and Rachel Player, *Simple encrypted arithmetic library - SEAL v2.1*, Cryptology ePrint Archive, Report 2017/224, 2017, https://eprint.iacr.org/2017/224.
34. 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.
35. 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.
36. Yao Chen, Benjamin Case, Shuhong Gao, and Guang Gong, *Error analysis of weak Poly-LWE instances*, 2017, http://cacr.uwaterloo.ca/techreports/2017/cacr2017-07.pdf.
37. 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
38. Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrède Lepoint, Mehdi Tibouchi, and Aaram Yun, *Batch fully homomorphic encryption over the integers*, Advances in Cryptology – EUROCRYPT 2013 (Berlin, Heidelberg) (Thomas Johansson and Phong Q. Nguyen, eds.), Springer Berlin Heidelberg, 2013, pp. 315–335.
39. 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.
40. 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.
41. Wutichai Chongchitmate and Rafail Ostrovsky, *Circuit-private multi-key FHE*, Proceedings, Part II, of the 20th IACR International Conference on Public-Key Cryptography — PKC 2017 - Volume 10175 (New York, NY, USA), Springer-Verlag New York, Inc., 2017, pp. 241–270.
42. Kai-Min Chung, Yael Kalai, and Salil Vadhan, *Improved delegation of computation using fully homomorphic encryption*, Advances in Cryptology – CRYPTO 2010 (Berlin, Heidelberg) (Tal Rabin, ed.), Springer Berlin Heidelberg, 2010, pp. 483–501.
43. Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi, *Fully homomorphic encryption over the integers with shorter public keys*, Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings, 2011, pp. 487–504.
44. Thomas M. Cover and Joy A. Thomas, *Elements of Information Theory*, 2nd Edition, John Wiley and Sons, 2006.
45. Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P. Smart, *Practical covertly secure mpc for dishonest majority – or: Breaking the spdz limits*, Computer Security – ESORICS 2013 (Berlin, Heidelberg) (Jason Crampton, Sushil Jajodia, and Keith Mayes, eds.), Springer Berlin Heidelberg, 2013, pp. 1–18.
46. Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias, *Multiparty computation from somewhat homomorphic encryption*, Advances in Cryptology – CRYPTO 2012 (Berlin, Heidelberg) (Reihaneh Safavi-Naini and Ran Canetti, eds.), Springer Berlin Heidelberg, 2012, pp. 643–662.
47. Ivan Damgård, Antigoni Polychroniadou, and Vanishree Rao, *Adaptively secure multi-party computation from LWE (via equivocal FHE)*, Public-Key Cryptography – PKC 2016 (Berlin, Heidelberg) (Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, eds.), Springer Berlin Heidelberg, 2016, pp. 208–233.
48. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan, *Fully homomorphic encryption over the integers*, IACR Cryptology ePrint Archive **2009** (2009), 616.
49. Marten van Dijk and Ari Juels, *On the impossibility of cryptography alone for privacy-preserving cloud computing*, Proceedings of the 5th USENIX Conference on Hot Topics in Security (Berkeley, CA, USA), HotSec'10, USENIX Association, 2010, pp. 1–8.
50. 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.
51. 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
52. 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.

53. 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.
54. Junfeng Fan and Frederik Vercauteren, *Somewhat practical fully homomorphic encryption*, Cryptology ePrint Archive, Report 2012/144, 2012, https://eprint.iacr.org/2012/144.
55. Steven D. Galbraith, *Mathematics of public key cryptography*, 1st ed., Cambridge University Press, New York, NY, USA, 2012.
56. Shuhong Gao, Frank Volny IV, and Mingsheng Wang, *A new framework for computing Grobner bases*, Math. Comput. **85** (2016), no. 297, 449–465.
57. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters, *Candidate indistinguishability obfuscation and functional encryption for all circuits*, Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science (Washington, DC, USA), FOCS '13, IEEE Computer Society, 2013, pp. 40–49.
58. Rosario Gennaro, Craig Gentry, and Bryan Parno, *Non-interactive verifiable computing: Outsourcing computation to untrusted workers*, Advances in Cryptology – CRYPTO 2010 (Berlin, Heidelberg) (Tal Rabin, ed.), Springer Berlin Heidelberg, 2010, pp. 465–482.
59. Rosario Gennaro and Daniel Wichs, *Fully homomorphic message authenticators*, Advances in Cryptology - ASIACRYPT 2013 (Berlin, Heidelberg) (Kazue Sako and Palash Sarkar, eds.), Springer Berlin Heidelberg, 2013, pp. 301–320.
60. 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.
61. Craig Gentry and Shai Halevi, *Fully homomorphic encryption without squashing using depth-3 arithmetic circuits*, 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science—FOCS 2011, IEEE Computer Soc., Los Alamitos, CA, 2011, pp. 107–116. MR 2932685
62. ———, *Implementing Gentry's fully-homomorphic encryption scheme*, Advances in cryptology—EUROCRYPT 2011, Lecture Notes in Comput. Sci., vol. 6632, Springer, Heidelberg, 2011, pp. 129–148. MR 2813639
63. Craig Gentry, Shai Halevi, and Nigel P. Smart, *Better bootstrapping in fully homomorphic encryption*, Public key cryptography—PKC 2012, Lecture Notes in Comput. Sci., vol. 7293, Springer, Heidelberg, 2012, pp. 1–16. MR 2980588
64. ———, *Fully homomorphic encryption with polylog overhead*, Advances in cryptology—EUROCRYPT 2012, Lecture Notes in Comput. Sci., vol. 7237, Springer, Heidelberg, 2012, pp. 465–482. MR 2972914
65. 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.
66. O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game*, Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '87, ACM, 1987, pp. 218–229.
67. Shafi Goldwasser, Yael Tauman Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich, *How to run turing machines on encrypted data*, Advances in Cryptology – CRYPTO 2013 (Berlin, Heidelberg) (Ran Canetti and Juan A. Garay, eds.), Springer Berlin Heidelberg, 2013, pp. 536–553.
68. Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs, *Leveled fully homomorphic signatures from standard lattices*, Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '15, ACM, 2015, pp. 469–477.
69. Qian Guo, Thomas Johansson, and Paul Stankovski, *Coded-BKW: Solving LWE using lattice codes*, pp. 23–42, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.
70. Shai Halevi, *Tutorial on homomorphic encryption*, https://shaih.github.io/pubs/he-chapter.pdf (2017).
71. Shai Halevi, Carmit Hazay, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam, *Round-optimal secure multi-party computation*, Cryptology ePrint Archive, Report 2017/1056, 2017, https://eprint.iacr.org/2017/1056.
72. Shai Halevi and Victor Shoup, *Algorithms in HElib*, Advances in cryptology—CRYPTO 2014. Part I, Lecture Notes in Comput. Sci., vol. 8616, Springer, Heidelberg, 2014, pp. 554–571. MR 3239456
73. ———, *Bootstrapping for* `helib`, Advances in cryptology—EUROCRYPT 2015. Part I, Lecture Notes in Comput. Sci., vol. 9056, Springer, Heidelberg, 2015, pp. 641–670. MR 3344941
74. 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.

75. Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman, *Ntru: A ring-based public key cryptosystem*, Algorithmic Number Theory (Berlin, Heidelberg) (Joe P. Buhler, ed.), Springer Berlin Heidelberg, 1998, pp. 267–288.
76. Nick Howgrave-Graham and Antoine Joux, *New generic algorithms for hard knapsacks*, Advances in Cryptology – EUROCRYPT 2010 (Berlin, Heidelberg) (Henri Gilbert, ed.), Springer Berlin Heidelberg, 2010, pp. 235–256.
77. Ravi Kannan, *Minkowski's convex body theorem and integer programming*, Math. Oper. Res. **12** (1987), no. 3, 415–440.
78. Miran Kim and Kristin Lauter, *Private genome analysis through homomorphic encryption*, BMC Medical Informatics and Decision Making **15(Suppl 5)** (2015), no. S3, http://doi.org/10.1186/1472-6947-15-S5-S3.
79. 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.
80. 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.
81. 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.
82. A. K. Lenstra, H. W. Lenstra, and L. Lovász, *Factoring polynomials with rational coefficients*, Mathematische Annalen **261** (1982), no. 4, 515–534.
83. Yehuda Lindell, Nigel P. Smart, and Eduardo Soria-Vazquez, *More efficient constant-round multi-party computation from BMR and SHE*, Theory of Cryptography (Berlin, Heidelberg) (Martin Hirt and Adam Smith, eds.), Springer Berlin Heidelberg, 2016, pp. 554–581.
84. 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.
85. 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
86. Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan, *On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption*, Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '12, ACM, 2012, pp. 1219–1234.
87. 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
88. Daniele Micciancio and Chris Peikert, *Hardness of SIS and LWE with small parameters*, Advances in Cryptology – CRYPTO 2013 (Berlin, Heidelberg) (Ran Canetti and Juan A. Garay, eds.), Springer Berlin Heidelberg, 2013, pp. 21–39.
89. Daniele Micciancio and Oded Regev, *Lattice-based cryptography*, pp. 147–191, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
90. 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.
91. 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
92. 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.
93. 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.
94. National Institute of Standards and Technology, *FIPS PUB 202 SHA-3 standard: Permutation-based hash and extendable-output functions*, 2015.
95. Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky, *Maliciously circuit-private FHE*, Advances in Cryptology – CRYPTO 2014 (Berlin, Heidelberg) (Juan A. Garay and Rosario Gennaro, eds.), Springer Berlin Heidelberg, 2014, https://eprint.iacr.org/2013/307, pp. 536–553.
96. Bryan Parno, Mariana Raykova, and Vinod Vaikuntanathan, *How to delegate and verify in public: Verifiable computation from attribute-based encryption*, Theory of Cryptography (Berlin, Heidelberg) (Ronald Cramer, ed.), Springer Berlin Heidelberg, 2012, pp. 422–439.

97. 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.
98. Oded Regev, *On lattices, learning with errors, random linear codes, and cryptography*, J. ACM **56** (2009), no. 6, 34:1–34:40.
99. R L Rivest, L Adleman, and M L Dertouzos, *On data banks and privacy homomorphisms*, Foundations of Secure Computation, Academia Press (1978), 169–179.
100. 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.
101. N. P. Smart and F. Vercauteren, *Fully homomorphic encryption with relatively small key and ciphertext sizes*, Public Key Cryptography – PKC 2010 (Berlin, Heidelberg) (Phong Q. Nguyen and David Pointcheval, eds.), Springer Berlin Heidelberg, 2010, pp. 420–443.
102. ———, *Fully homomorphic simd operations*, Designs, Codes and Cryptography **71** (2014), no. 1, 57–81.
103. Damien Stehlé and Ron Steinfeld, *Making NTRU as secure as worst-case problems over ideal lattices*, Advances in Cryptology – EUROCRYPT 2011 (Berlin, Heidelberg) (Kenneth G. Paterson, ed.), Springer Berlin Heidelberg, 2011, pp. 27–47.
104. 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.
105. 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.

Department of Mathematical Sciences, Clemson University, Clemson, SC 29634-0975 USA
sgao@clemson.edu